

Ensuring Data Security and Annotators Anonymity Through a Secure and Anonymous Multiparty Annotation System

Dany Rimez(✉), Axel Legay, and Benoît Macq

ICTEAM, UCLouvain, Ottignies-Louvain-la-Neuve, Belgium
dany.rimez@uclouvain.be

Abstract. Healthcare is a domain characterized by the continuous influx of unannotated data generated by clinical workflows. The potential benefits of integrating artificial intelligence in support of clinical decisions are therefore substantial. An efficient machine learning approach should rely on a continual finetuning of the model with human expert annotations. In order to reduce the workload of the experts, Active Learning strategies are required. They involve the active participation of medical experts to provide the most informative annotations for model finetuning. The main goal of this paper is to propose an architecture which builds trust among the annotators in the process, thereby encouraging their engagement and motivation for ongoing participation in the annotation process. We provide a Secure and Anonymous Multiparty Annotation System (SAMAS) that could be used for securely annotating data in both Active Learning and Continuous annotation workflow. This architecture is based on conditional access mechanisms and communication scheme implying a Trusted Third Party to ensure both security of communications and annotators anonymity. We verified our implementation, and its compliance to a set of predefined properties, in a simplified example of distributed annotation using Spin and Linear Temporal Logic.

Keywords: Active Learning · Security Architecture · Trusted annotation · Privacy preserving annotation · formal verification · Promela

1 Introduction

In the domain of healthcare, large unannotated medical datasets are continuously generated through the workflows of diverse hospitals. These datasets hold great potential for training deep learning models but require sufficient annotations to be useful. Trusted and continuous fine-tuning of AI models necessitates a secure flow of data and annotations. This secure flow is crucial for training the models and keeping them up-to-date with the evolving data, such as new equipment,

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024
P. Mylonas et al. (Eds.): NiDS 2024, LNNS 1170, pp. 620–631, 2024.
https://doi.org/10.1007/978-3-031-73344-4_54

changes in medical procedures, and epidemics. However, maintaining this process relies on a budget, which supports the infrastructure and resources needed for data security, annotation, and ongoing model updates.

Following Active Learning (AL) strategies, we can reduce the budget by optimising the selection of data to annotate. For instance, we can ask annotations to human experts for the most uncertain cases, generally the most difficult and the most informative. This way, AL allows to both increase the quality of the dataset and of the model trained on it, and reduce the time human experts have to spend time on annotating data. However, this process is still bounded by the availability of the annotators.

On the other hand, Federated Learning (FL) [9] allows to train a model on several datasets simultaneously, augmenting the amount of data and annotators available while ensuring the security of data.

As depicted on Fig. 1, FL allows to train a global model from multiple annotated data silos belonging to each client (hospital for instance). First, a local model is trained in each hospital on the local data annotated by the medical experts of this hospital. The knowledge of these local models is then aggregated to build a global model.

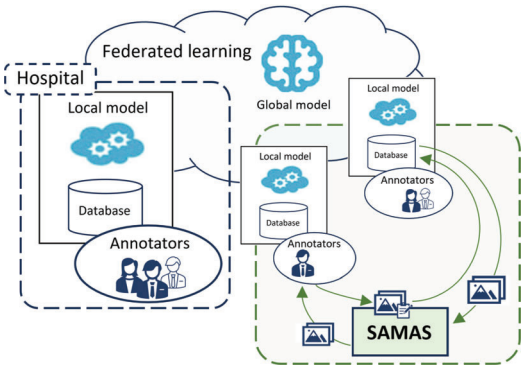


Fig. 1. Presentation of SAMAS in the context of distributed learning in a coalition of Hospitals.

By combining AL and FL, we can both increase the availability of annotators and quality of annotations, while keeping medical data secured. In such a learning architecture, difficult cases can be screened by several experts with different opinions, and enrich the shared knowledge of the model. This process will lead to a continual training and alignement of the pool of experts on the most difficult cases and will build a global consensus. However, new problems arises in a coalition of hospitals as both the security of the annotation itself and the anonymity of annotators has to be valued.

Traditional distributed learning architectures, such as vanilla FL, incorporating human annotators, often inadequately address imperative considerations of

security and anonymity. Even though many papers have brought improvements concerning security of data and models [2,8,10,12,13], client anonymity [3], and annotators evaluation [5,11] that could be used to identify spurious/malicious annotators, no work have been done yet to secure annotators identity throughout the annotation task.

We then propose a new security architecture to annotate data in such an FL setting. This way, confidential data from one hospital can not only be annotated by one of its local expert, but also by an expert of another participating hospital. By promoting the anonymity of annotators, our Secure and Anonymous Multiparty Annotation System (SAMAS) also allow an objective and incentivizing peer-reviewing process. SAMAS is meticulously engineered to tackle the following security concerns in distributed learning:

- accountability of the external experts with regards on the confidentiality, anonymity and non-divulgation of the data to be annotated
- anonymity of the experts may be wished for fair comparisons in multi-experts labels and for avoiding accountability issues in the label
- compensating the contributions of the experts of the pool by setting a tokenisation procedure

It is based on conditional access mechanisms and communication scheme implying a Trusted Third Party (TTP). We verified the compliance of its design to a set of properties in a simulated coalition with a formal verification tool. This allows an exhaustive search of our design's failures, which could not be achieved with a real-world prototyping, on top of beeing much faster and cheaper to achieve.

Spin uses the modeling language Promela (Process Meta Language) to represent concurrent processes communicating and interacting with each other. The model of the protocol can then be executed, and each possible configuration of the model is called a state. A state is defined by the values of variables and the control location of each process at a given point in the execution.

Spin verifies a model by exploring all its possible states with the "explicit state model checking" algorithm to check for the presence of errors, such as deadlocks or violations of specified properties. These properties often involve reaching a certain desirable state or satisfying specific conditions, and can be translated into statements using Linear Temporal Logic (LTL) [7], so that Spin can check these statements are True at every steps of the execution of the model. It is particularly valuable for systems with complex concurrency and synchronization issues, as it exhaustively explores the state space, providing a high level of confidence in the correctness of the model.

The key steps in using Spin for verification of a protocol with concurrent processes include:

1. Modeling: Define the processes taking part in the protocol, their interactions. Implement these processes in Promela. Define the properties to verify and implement them with LTL.
2. Verification: Run spin to check for errors or violations of these properties.

3. While verifying a property, Spin will search for a counterexample where the property is violated, identifying a sequence of events leading to the issue.
4. If a counterexample is found, we fix the implementation of the protocol to cope with this particular case.

We will begin by developing the requirements of SAMAS. Starting from these requirements, we will then present the activity diagram of SAMAS and define its key properties. Formal verification with Spin will allow us to prove that the activity diagram adheres to these properties.

2 Definition of the Requirements

Let's consider a coalition of hospitals with a need to annotate medical images. We have access to a pool of medical expert annotators and an active learning oracle (the sampler), which can select images that require annotation.

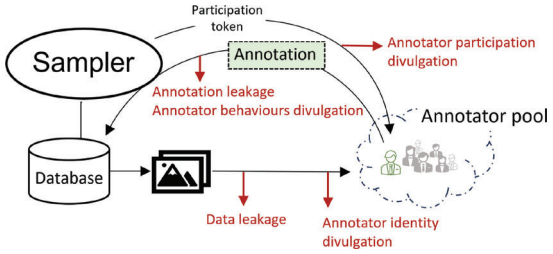


Fig. 2. Communication diagram depicting the threats of an annotation workflow without SAMAS.

As sketched on Fig. 2, such an annotation workflow is exposed to several threats that should be dealt with:

- **Data leakage.** Leakage of sensitive data (i.e. medical images).
- **Annotation leakage.** Leakage of the annotation (ie segmentation mask of a tumor by a doctor)
- **Participation divulgation.** Leakage of token transfers, hence divulgation of the identity of annotators that performed an annotation task.
- **Identity divulgation.** Leakage of the notification sent by the central server to the annotators, hence divulgation of the identities of annotators queried for this annotation task.

Apart from the common data leakage issues (threats 1 and 2), our work puts a specific concern on protecting each annotator against wrong usage of the annotation history. Indeed, one could also intercept annotations to extract annotations behaviours of specific annotators, and infer both participation and performance statistics for each annotator.

Starting from those elements, a communication protocol should answer to some constraints:

Secure. We need to annotate images that are medical data. Those images are then of sensitive nature and it is essential to avoid any data leakage (threats 1 and 2).

Anonymous. Annotators are medical practitioners coming from any of the hospital (hospital) member of the coalition. As the annotation performance of an annotator could be used to compare him to others or to compare hospital, we value annotators anonymity. Only the central server should know who annotated what, and this information should not be accessible otherwise (threats 3 and 4).

To these constraints, we could add more general features linked to the multi-annotator annotation framework:

Incentive. As the time of annotators is precious, we should reward them for their contribution to the coalition. Moreover, annotators should be notified when they have the possibility to contribute to the coalition (i.e., when they have an image to annotate) in a way that still guarantee both fairness amongst annotators and their anonymity.

Multiple Annotators. From the latter point, we come up with a new constraint on the design. The sampler will select several annotators to increase the security of the communication protocol. Indeed, if a notification is sent to 5 annotators to annotate an image, it is less likely that all notifications would be intercepted and no annotators would receive a notification. This design also allow some optimisation of the annotation time as we will only have to wait for the first annotator.

2.1 Key Features

Following the aforementioned specifications, we define key elements of the design of SAMAS:

Encryption. Images will be transmitted using conditional access mechanisms based on entitlements and hybrid encryption, where each annotator own a private decryption key. The central server generates at random a public key to encrypt the image and encrypt the key with the private encryption keys of all annotators allowed to annotate this image.

TTP. The presence of a Trusted Third Party (TTP) is essential in our design as it will play a crucial role in establishing trust contracts between medical experts and the central server. By providing a guarantee on the link between the annotators' identity and their private key, it will ensure access transactions to the data (and any potential payment) without the need for direct contact between the annotators and the central server.

Data Cache. A data cache will is needed for transient storage of images and annotations during the annotation process, so that images can be queried by

any annotator of any coalition center if he has one of the right decryption keys. This cache will also allow to keep secret the id of the image annotated and preserve the anonymity of annotators. Indeed, for a given annotator, the most complete information available for the TTP would be that this annotator annotated something and when, but not which image within the whole aggregated dataset was annotated.

Annotation Tag. A tag will be used in the conditional access mechanism to indicate when an image has been annotated, so the central server can retrieve it. This tag will also be encrypted with the private key owned by the annotator, and appended to the encrypted annotated image. This way, the central server can decrypt it with the corresponding private decryption key and infer the identity of the annotator that provided the annotation.

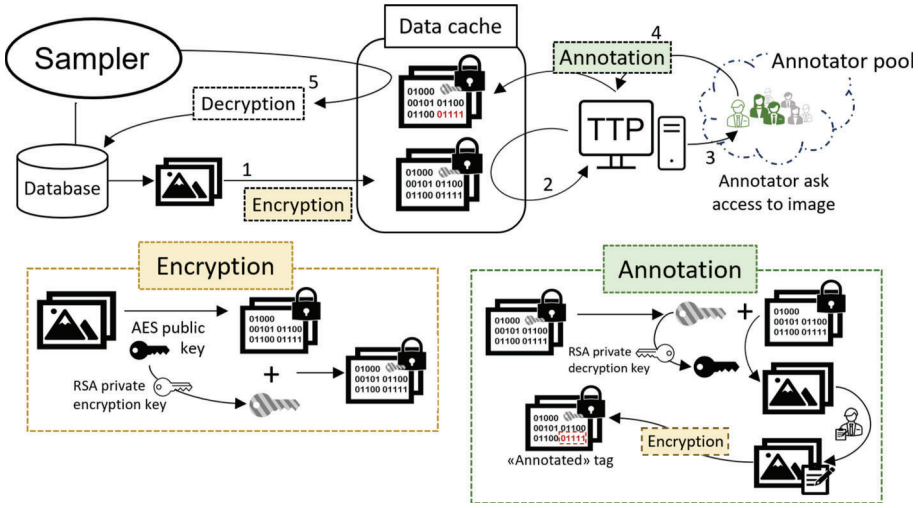


Fig. 3. Scheme of the proposed communication protocol.

3 Design Presentation

The proposed protocol is represented on Fig. 3. After beeing selected by the sampler of the central server, an image to be annotated is first encrypted following an hybrid encryption protocol [4]. It will be encrypted by a randomly-generated AES public key [1], itself encrypted with a private RSA key [14] and added to the cyphertext. The image is encrypted and sent to the data cache (1), waiting to be annotated, and the identity of the annotators with access to this image is communicated to the TTP.

The AES key is randomly generated for each image transmission, and the RSA public key controls access to a limited number of annotators via the TTP.

Indeed, when an annotator wants to annotate an image, its authentication with the TTP determines its access rights to this image. Once it is shared to the annotator, it is removed from the pool so that only one annotation will be provided by the pool (2).

The image can now be locally decrypted with the RSA private key of the TTP (3). The annotator can now modify it to provide the annotation, and the annotated image is encrypted following the same encryption process to be sent back to the data cache (4). The only difference is the addition of a tag to the cyphertext to indicate the image has been annotated.

Once the annotated image in the cache, it is recovered and removed from the cache by the sampler that regularly checks for annotated data (5). This retrieval is allowed by the presence of a tag in annotated data. Finally, annotators regularly receive "participation tokens" from the central server depending on their contribution to the coalition within a given time window.

*T*he properties we will verify in this work are the following:

1. Only one annotator, amongst the ones queried to annotate a given image, gets access to that image, and only once.
2. An image can only be removed from the cache to be accessed by the annotator.
3. Once an image is annotated, it is sent back to the cache.
4. Once an annotator sent an annotation, its access to the corresponding image is revoked.
5. An annotator gets a token if and only if it has performed an annotation: if an annotation is performed, a token is received, and reciprocally.
6. No annotator can have access to several images simultaneously.
7. When an annotated image is present in the cache, it is removed from the pool and retrieved by the central server.
8. All images in the cache are annotated.

4 Design Proof

This protocol was implemented in Promela and verified with Spin [6] for compliance to a set of key properties defined with Linear Temporal Logic [6]. This verification process followed the steps presented earlier in Sect. 1.

4.1 Abstraction Model

In order to implement an abstraction model of SAMAS in Promela, we need to build a simplified environment in which we can define the processes, their interactions, and the properties to check.

This environment is the following: The central server will send 2 images into the data cache and 4 annotators will be selected to annotate those images. Image 1 was assigned to annotators 0 and 1, and image 2 was assigned to annotators 1, 2 and 3. Several annotators are then queried to annotate the same image, and an annotator was queried to annotate several images. We can now define the processes of the model:

- **User (Annotator):** Notified for the annotation task, users will try to annotate their dedicated image(s) if possible.
- **Central Server:** Send images to data cache through the *cent2ttp*, notifies the annotators with the channel *cent2user*, and retrieve annotated data through the channel *ttp2cent*.
- **TTP:** Interact with the annotators using the *user2ttp* and *ttp2user* channels, and with the data cache using the *cent2ttp* and *ttp2cent* channels. The TTP will play the intermediate between the data cache and the annotators to allow them to annotate images if doing so respect the desired properties.

The data cache is modeled with the communication channels between the central server and TTP. The image can wait within *cent2ttp* to be downloaded by the TTP, or the annotated image within *ttp2cent* to be retrieved by the central server.

As all annotators can have access to the TTP simultaneously, one "TTP process" was created for each annotator to model their interaction with the TTP. As this interaction implies the annotator and its TTP process are synchronised with one another, *rendez-vous* communications were used between the TTP and user processes to simplify the model. If it sends a message, the annotator is then unable to continue its execution while the TTP has not received it. This synchronization also prevent an annotator to try accessing a second time to the TTP while waiting for its response, implicitly ensuring property 6.

The last 2 properties can easily be verified by the full completion of the model execution. In this case, all images were annotated and removed from the cache and properties 7 and 8 holds if the execution always finish. The other properties (1 to 5) cannot be verified by simply checking for deadlocks, it is needed to implement them with LTL. For instance, the property 1 can be implemented as the following:

```
#define img1 (len(cache2ttp[0])) // Image 1 is still in the cache
#define img2 (len(cache2ttp[1])) // Image 2 is still in the cache
#define rec0 (rec[0]==1 & img1+img2==1) // Annotator 0 received something
⋮
#define rec3 (rec[3]==1 & img1==1 & img2==0) // Annotator 3 received image 2
#define intermediate (rec0 ||...|| rec3) // 1 image has be received by an annotator
ltl p1 { always (
                    (rec[0]==0 & rec[1]==1 & rec[2]+rec[3]==1)
                 || (rec[0]==2 & rec[1]+rec[2]+rec[3]==0)
                 || (rec[0]==1 & rec[1]==1 & rec[2]+rec[3]==0)
                 || (rec[0]==1 & rec[1]==0 & rec[2]+rec[3]==1)
                 || intermediate
                 || img1+img2==0
               )
}
```

Where rec is an array that allows the TTP to count the number of images received by each user from the beginning of model execution.

```
if
  :: len(cache2ttp[1])==1      // image 1 is still in the cache
     → ttp2user[n]!to_annot // image 1 is sent to annotator n
       rec[n] = rec[n] + 1   // annotator n received 1 more image
       cache2ttp[1]?data // image 1 is removed from the cache
  :: len(cache2ttp[1])==0 // image 1 is not in the cache
     → annotators[n] = 0 // access to image 1 is revoked for annotator n
fi
```

(a) Naive implementation

```
if
  :: cache2ttp[1]?data        // image 1 is removed from the cache if possible
     → ttp2user[n]!to_annot // image 1 is sent to annotator n
       rec[n] = rec[n] + 1   // annotator n received 1 more image
  :: len(cache2ttp[1])==0   // image 1 is not in the cache
     → annotators[n] = 0 // access to image 1 is revoked for annotator n
fi
```

(b) Corrected implementation

Fig. 4. Example of modification of our implementation to ensure commitment to property 1. Red lines in naive implementation (a) were replaced by green line in corrected implementation (b).

4.2 Verification

Now that the model is implemented, we can use Spin to verify its compliance to the defined properties. Correctness of LTL formulation of the properties was checked by both changing the evaluation design, and tweaking the Promala model implementation to provoke violations of the property.

For example, a naive implementation of the Property 1 could ignore the *intermediate* statements in the LTL definition, only considering the valid terminal states of the model provided by the "rec" array. However, this would allow a particular case of violation of the property as "rec" only stores the amount of received images, and not their nature. If annotator 2 gets access to image 2, and annotator 3 to image 1, the naive LTL definition would not consider this as a violation of the property as the final state is considered as valid (line 0 of the LTL definition). Adding the intermediate check "*if annotator 1 received an image, it must be the image 1*" allows to circumvent this shortcoming and to correct the LTL.

This iterative correction procedure was achieved for all stated properties. Complete validation of the protocol was then achieved by successive modifications of both the Promela implementation of the model, and the LTL definition of the properties until all properties were satisfied with the resulting Promela model of SAMAS.

An example of correction of our implementation to ensure commitment to property 1 is given on Fig. 4. In the naive implementation of Fig. 4a, image is removed from the cache after it was sent to the annotator. However, if 2

annotators pass first the condition of the if statement (the image is still in the cache) simultaneously, the image could be sent to both annotators before removing it from the cache. Property 1 is then violated.

If we correct the model as shown on Fig. 4b, the first condition of the if statement can now be passed only if the image can be removed from the cache. The image is then removed before sending it to the annotator, and before another TTP process can transmit it. Property 1 is now verified.

This example shows the importance of allowing the TTP to download an image only and only if the image is not already beeing downloaded.

We obtained a valid Promela implementation of SAMAS by following the same steps for all properties. This implementation can be found on github: https://anonymous.4open.science/r/samas-C288.

5 Discussion

One limitation of our work is the fact that verification with Spin relies on a correct definition of the abstract model in Promela and properties with LTL. Depending on the real system we implement it for, and on the constraints of the given programation language, all of these definition should be checked, and the Promela model should be completed.

Indeed, we simplified our simulation by assuming that human annotators would always complete the annotation tasks they started. However, if an annotator abandons a task midway, it implies that the annotator accessed the image but did not return an annotation, thereby violating property 3. Similarly, issues arise if no annotator completes the annotation. To address this, a new property could be introduced: "Both annotated and unannotated images should not remain in the cache for more than X minutes," with X being specific to the use case. Unfortunately, this property cannot be verified with Spin, as it does not support real-time constraints. Another potential property is: "Annotation tasks should be completed within Y minutes, or access to the image should be revoked."

Additionally, our model assumes the trustworthiness of medical experts accessing the TTP. A malicious entity could, for example, steal an expert's private key to gain unauthorized access to the TTP.

6 Conclusion

In this paper, we proposed a Secure and Anonymous Multiparty Annotation System (SAMAS) that could be used for securely annotating data in both Active Learning and Continuous annotation workflow.

By providing a secure and anonymous framework for data annotation, SAMAS enhances the confidentiality of medical information and encourages the active participation of medical experts in annotation tasks by preserving their anonymity.

The rigorous verification process ensures that SAMAS operates securely and efficiently, adhering to all specified security properties before real-world implementation.

This validation not only confirms the robustness of the protocol design but also strengthens trust in the system's ability to securely handle sensitive medical data.

References

1. Advanced encryption standard (AES). Technical report, November 2001. https://doi.org/10.6028/nist.fips.197
2. Abadi, M., et al.: Deep learning with differential privacy. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pp. 308–318 (2016)
3. Asad, M., Aslam, M., Jilani, S.F., Shaukat, S., Tsukada, M.: SHFL: K-anonymity-based secure hierarchical federated learning framework for smart healthcare systems. Future Internet **14**(11), 338 (2022). https://doi.org/10.3390/fi14110338
4. Chandu, Y., Kumar, K.S.R., Prabhukhanolkar, N.V., Anish, A.N., Rawal, S.: Design and implementation of hybrid encryption for security of IoT data. In: 2017 International Conference on Smart Technologies for Smart Nation (SmartTechCon), pp. 1228–1231 (2017). https://doi.org/10.1109/SmartTechCon.2017.8358562
5. Goh, H.W., Mueller, J.: ActiveLab: active learning with re-labeling by multiple annotators (2023). https://doi.org/10.48550/ARXIV.2301.11856. https://arxiv.org/abs/2301.11856
6. Holzmann, G.: Formal verification. https://spinroot.com/spin/whatispin.html
7. Holzmann, G.: Linear time temporal logic. https://spinroot.com/spin/Man/ltl.html
8. Lugan, S., Desbordes, P., Brion, E., Ramos Tormo, L.X., Legay, A., Macq, B.: Secure architectures implementing trusted coalitions for blockchained distributed learning (TCLearn). IEEE Access **7**, 181789–181799 (2019). https://doi.org/10.1109/ACCESS.2019.2959220
9. McMahan, B., Moore, E., Ramage, D., Hampson, S., Arcas, B.A.Y.: Communication-efficient learning of deep networks from decentralized data. In: Singh, A., Zhu, J. (eds.) Proceedings of the 20th International Conference on Artificial Intelligence and Statistics. Proceedings of Machine Learning Research, vol. 54, pp. 1273–1282. PMLR, 20–22 April 2017. https://proceedings.mlr.press/v54/mcmahan17a.html
10. Ryffel, T., et al.: A generic framework for privacy preserving deep learning. arXiv preprint arXiv:1811.04017
11. Tanno, R., Saeedi, A., Sankaranarayanan, S., Alexander, D.C., Silberman, N.: Learning from noisy labels by regularized estimation of annotator confusion. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 11236–11245 (2019). https://doi.org/10.1109/CVPR.2019.01150
12. Wittkopp, T., Acker, A.: Decentralized federated learning preserves model and data privacy. In: Hacid, H., et al. (eds.) ICSOC 2020. LNCS, vol. 12632, pp. 176–187. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-76352-7_20

13. Zhao, B., Fan, K., Yang, K., Wang, Z., Li, H., Yang, Y.: Anonymous and privacy-preserving federated learning with industrial big data. IEEE Trans. Ind. Inf. **17**(9), 6314–6323 (2021). https://doi.org/10.1109/tii.2021.3052183
14. Zhou, X., Tang, X.: Research and implementation of RSA algorithm for encryption and decryption. In: Proceedings of 2011 6th International Forum on Strategic Technology, vol. 2, pp. 1118–1121 (2011). https://doi.org/10.1109/IFOST.2011.6021216