Privacy-Diffusion: Privacy-Preserving Stable Diffusion Without Homomorphic Encryption

Po-Chu Hsu
Animechain.ai Inc.
California, USA
hsu@animechain.ai

Ziying Yu
Amazon
California, USA
ziyingy@amazon.com

Shuhei Mise
Animechain.ai Inc.
Tokyo, Japan
mish@animechain.ai

Hideaki Miyaji
Ritsumeikan University
Osaka, Japan
h-miyaji@fc.ritsumei.ac.jp

2025 IEEE International Conference on Consumer Electronics (ICCE) | 979-8-3315-2116-5/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICCE63647.2025.10929778

Abstract—**Text-to-image generation is trending in the generative AI field. Stable Diffusion is the state-of-the-art among open-source projects. Many artists and service providers customize the diffusion model for special textures. However, there is no protection for the privacy of the user's input text prompt, output image, and the customized model on the server. Privacy is crucial for user trust and protecting intellectual property. Existing privacy-preserving diffusion models use fully homomorphic encryption (FHE), which is time-consuming and can degrade image quality. We propose Privacy-Diffusion, a framework that preserves privacy without FHE by leveraging the irreversible properties of neural network layers and the property that in the diffusion process, the predicted noise is a normalized Gaussian distribution. Our framework protects clients' input text prompts and generated images from the server and safeguards customized models from clients. Compared with existing research HE-diffusion which spent 200% extra time and visible quality loss, our protocol can reach the same security level with only 4% extra time and has no quality loss. To our knowledge, we are the first to achieve this goal without FHE while maintaining high-quality image output.**

Index Terms—**AI Security, Privacy ML, Stable Diffusion, Generative AI.**

I. Introduction

Text-to-image generation is a key area in generative artificial intelligence (GenAI). Stable Diffusion [16] is the leading open-source project, invented the diffusion algorithm to create high-quality images from text prompts. Algorithms like DreamBooth [19] and LoRA [9] allows artists and service providers to customization the flavors of the output image. Protecting these customized models, as well as the client's input text prompt and output image, is crucial for privacy and intellectual property.

Required Properties: For a text-to-image generation service, the privacy-preserving diffusion algorithm must ensure:

- **Input Text Prompt Privacy**: The server cannot access the client's text prompt in plaintext.
- **Output Image Privacy**: The server cannot access the output image.
- **Model Privacy**: The client cannot access the model.

A. Backgrounds

To understand the challenges of building a privacy-preserving diffusion model, we introduce Stable Diffusion and existing privacy-preserving machine learning (Privacy ML) techniques.

- **Stable Diffusion** Image generation in Stable Diffusion [16] is a step-by-step procedure. As shown in Fig. 1, starting from a random noise, the diffusion model predicts the noise at each step and refines the image iteratively to generate a high-quality output.
- **Privacy ML Techniques** Privacy ML techniques protect training data [2], [10], the model, and the prediction process [3].
  - **Fully Homomorphic Encryption (FHE)**: FHE [1], [4] allows computations on encrypted data, producing encrypted outputs that only the client can decrypt.
  - **Downsizing and Quantization [11]**: Protecting private information by reducing model size to allows local predictions on personal devices.
  - **Differential Privacy**: Differential Privacy [6] adds noise to obscure sensitive information while allowing accurate aggregate statistics.

B. Difficulties and Challenges

Maintaining privacy while ensuring efficiency and image quality is challenging. Existing Privacy ML techniques have limitations:

- **FHE is Computationally Heavy**: The BGV [1] scheme is 23202 times slower, and the CKKS [4] scheme is 2055 times slower than plaintext multiplication. Such slowdowns are unacceptable for Stable Diffusion.

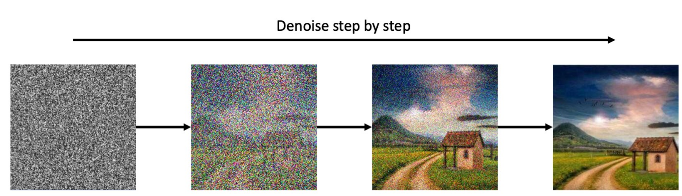

Fig. 1: Image generation in Stable Diffusion. Starting from random noise (left), the noise is removed step by step to generate a high-quality image (right).

- **Approximation Reduces Output Image Quality**: FHE libraries like Microsoft SEAL [15] only support addition and multiplication. Model accuracy can degrade because non-linear functions requires approximations.
- **Downsizing and Quantization Cannot Protect Model Privacy and May Reduce Accuracy**: Downsizing and quantization [11] allow local predictions but expose the model to the client and may reduce accuracy.
- **Differential Privacy May Decrease Accuracy**: There is a trade-off between privacy and accuracy. High privacy levels may add too much noise, reducing model accuracy.

C. Our Contributions

We propose Privacy-Diffusion, a privacy-preserving diffusion framework with no computation overhead or image quality loss. By leveraging the irreversible property of neural network layers and the normalized Gaussian distribution of predicted noise, our protocol protects input text prompt, output image, and customized model privacy without FHE, downsizing, quantization, or differential privacy techniques. Our implementation is available at https://github.com/Animechain-ai/Privacy-Diffusion. Our contributions are:

- **Security Without FHE**: Utilizing neural network layers' irreversible property and the normalized Gaussian distribution of predicted noise, our protocol is secure without FHE or encryption schemes.
- **Privacy Without Computation Overhead**: Our protocol has no extra computation overhead, relying on proper distribution of computations between client and server.
- **No Quality Loss**: Our protocol does not use differential privacy or approximations, maintaining high-quality image output.

This paper demonstrates related Privacy ML protocols in Section II, introduces preliminaries in Section III, proposes our Privacy-Diffusion protocol in Section IV, discusses security in Section V, and demonstrates implementation and optimization in Section VI. We conclude in Section VII.

II. Related Works

Various methods has been used to protect neural network privacy, such as homomorphic encryption (HE) [12]. CryptoNets [7] first applied HE to neural networks. Prior works on privacy-preserving diffusion models focus on protecting training data [2], [10] from malicious parties, emphasizing differential privacy [6] and protection against membership inference attacks [14]. Protecting training data is crucial, but the privacy of the image generation process is also important.

HE-Diffusion [3] is the first framework focused on the image generation process. They reduce computation time by protecting the noise prediction part with the irreversibility property of neural network layers and only the denoising part requires FHE. They optimize performance using partial encryption, image division, and sparse encryption. We propose a method to protect diffusion model privacy and security without FHE or encryption schemes, maintaining high-quality image output with only 4% extra time compared to the 200% extra time of HE-Diffusion.

III. Preliminaries

This section defines the Stable Diffusion model and its components: text encoder, UNet, denoise function, and Variational Autoencoder (VAE).

Definition 1 (Text Encoder): A text encoder [5], [17] tokenizes and encodes text **prompt** into vectors $\mathbf{e} = (\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_n) \in \mathbf{E}$, where $\mathbf{e}_i \in \mathbb{R}^d$. Define $\mathit{TextEncoder}(\mathbf{prompt}) = (\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_n) = \mathbf{e}$

Definition 2 (UNet): Given a noisy image $\mathbf{z}_t \in \mathbf{Z}$ and text embedding $\mathbf{e} \in \mathbf{E}$, the UNet [18] model outputs predicted noise $\epsilon_t \in \mathbf{Z}$. Define $\epsilon_t = \mathit{UNet}(\mathbf{z}_t, \mathbf{e}, t)$

Definition 3 (Denoise): Given a noisy image $\mathbf{z}_t \in \mathbf{Z}$ and noise $\epsilon_t \in \mathbf{Z}$, the denoise algorithm [16] outputs a clearer image $\mathbf{z}_{t-1} \in \mathbf{Z}$. Define $\mathbf{z}_{t-1} = \mathit{Denoise}(\mathbf{z}_t, \epsilon_t)$

Definition 4 (Variational Autoencoder (VAE)): VAE [13], [8] converts images between pixel space $\mathbf{X}$ and latent space $\mathbf{Z}$. Define $\mathbf{z}_1 \approx \mathit{encoder}(\mathbf{x}_1), \quad \mathbf{x}_2 \approx \mathit{decoder}(\mathbf{z}_2)$

Definition 5 (Stable Diffusion): The Stable Diffusion [16] text-to-image process refines a noisy image step-by-step to produce a high-quality image.

Algorithm 1 Stable Diffusion text-to-image

Input: Text input **prompt**, iterations T
Output: Generated image $\mathbf{x}_0$

```
e ← TextEncoder(prompt)
z_T ← Z
for t = T to 1 do
    ε_t ← UNet(z_t, e, t)
    z_{t−1} ← Denoise(z_t, ε_t)
end for
x_0 ← VAE.decoder(z_0)
return x_0
```

IV. Our Protocol: Privacy-Diffusion

Privacy-Diffusion is a privacy-preserving diffusion framework that protects both the privacy of the client and the server. It can protect the client's text prompt and the generated image from being learned by the server. It can also protect the server's customized model from being learned by the client. Note that we are the first protocol that achieves these properties without using FHE and differential privacy techniques. The basic idea is to keep the computations directly relate to the text prompt and the image on the client side. Starting from

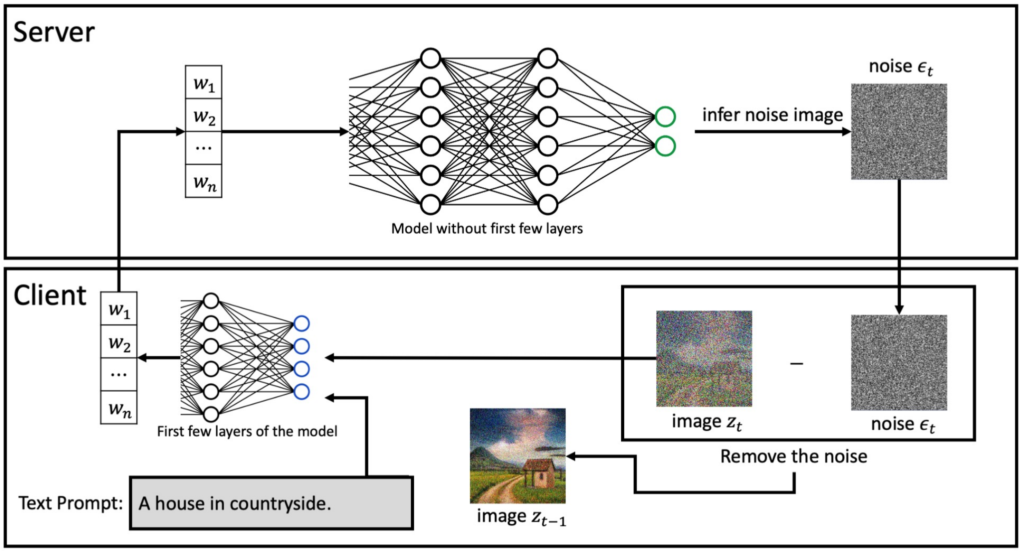

Fig. 2: Privacy-Diffusion

a randomly sampled noisy image, the algorithm repeats the following steps until the noise is totally removed.

- **Predict the Noise (Server Side):** The client performs the first few layers of the model to obfuscate the input and sends the intermediate result to the server. The server finishes the prediction and sends the predicted noise back to the client.
- **Remove the Noise (Client Side):** The client removes the noise based on the predicted noise and continues to the next iteration.

A. Notations

- $\mathbf{S}$: The server.
- $\mathbf{C}$: The client.
- *Denoise*: The algorithm used to reduce the noise.
- κ: The security parameter.
- T: The number of iterations to perform denoising.
- **prompt**: The client's text input.
- $\mathbf{e}$: The text embedding.
- $\mathbf{X}$: The pixel space of the output image.
- $\mathbf{x}_t$: The image in pixel space $\mathbf{X}$ at iteration t.
- $\mathbf{Z}$: The latent space of the output image.
- $\mathbf{z}_t$: The image in latent space $\mathbf{Z}$ at iteration t.
- ϵ_t: The predicted noise in latent space $\mathbf{Z}$ at t.

B. Our Protocol

We define a function *Split* that splits a model $\mathbf{M}$ into two parts: $\mathbf{M}_1$ and $\mathbf{M}_2$ based on a security parameter κ.

Definition 6 (Split function): Given a n layers neural network model $\mathbf{M}$ and a security parameter $0 \leq \kappa \leq 1$, the function *Split* splits the model into two parts: $\mathbf{M}_1$ contains the first $\lfloor n \cdot \kappa \rfloor$ layers and $\mathbf{M}_2$ contains the rest $n - \lfloor n \cdot \kappa \rfloor$ layers. Define $(\mathbf{M}_1, \mathbf{M}_2) \leftarrow Split(\mathbf{M}, \kappa)$.

Theorem 1 (Correctness of the Split function): Given a model $\mathbf{M}$, the split function is correct if for all input $\mathbf{x}$ in the domain of $\mathbf{M}$, the equation $\mathbf{M}(\mathbf{x}) = \mathbf{M}_2(\mathbf{M}_1(\mathbf{x}))$ holds.

We assume a client-server architecture where the server is stateless. The client controls the whole diffusion process. The client's algorithm is shown in Algorithm 2 and the server's algorithm is shown in Algorithm 3.

Algorithm 2 Client Algorithm

Input: Text input **prompt**, the number of iterations T
Output: Generated image $\mathbf{x}_0$

```
Receive M_1 from server.
e ← TextEncoder(prompt)
z_T ← Z
for t = T to 1 do
    ẑ_t ← M_1(z_t, e, t)
    ε_t ← S.PredictNoise(ẑ_t)
    z_{t-1} ← Denoise(z_t, ε_t)
end for
x_0 ← VAE.decoder(z_0)
return x_0
```

Algorithm 3 Server Algorithm

Input: *UNet* model, security parameter κ

```
(M_1, M_2) ← Split(UNet, κ)
Send M_1 to client.
procedure PredictNoise(ẑ_t)
    ε_t ← M_2(ẑ_t)
    Send ε_t back to the client.
end procedure
```

V. Security

Assuming a malicious client and an honest but curious server.

Fig. 3: Generated images from original Stable Diffusion (top) and Privacy-Diffusion (bottom).

TABLE I: Execution time of client and server (50 iterations, 512x512). Privacy-Diffusion requires 4% extra computation time.

	Original	Privacy-Diffusion
Client	0.1s	0.51s
Server	5.18s	5.02s
Total	5.28s	5.53s

1) Client's View: The client aims to learn the *UNet* model on the server.

- **Predicted noise ϵ_t:** The noise is a normalized Gaussian distribution in the latent space $\mathbf{Z}$. It is difficult for the client to learn the model by ϵ_t.
- **$\mathbf{M}_1$, the first few layers of *UNet*:** Learning only few layers does not enable the client to reproduce the model.

2) Server's View: The server aims to learn the client's text input **prompt** and the denoised image $\mathbf{z}_{t-1}$. The server's view includes:

- **Intermediate variable $\hat{\mathbf{z}}_t$:** The output of neural network $\mathbf{M}_1$, which is difficult to reverse-engineer due to irreversible layers.

Our protocol ensures input text prompt privacy, output image privacy, and model privacy as defined in Section I. It is simpler and faster than HE-diffusion as it does not require encryption of ϵ_t.

VI. Implementation

We benchmark on an AMD Ryzen 9 7950X3D CPU, 128GB RAM, and an NVIDIA RTX 4070 Ti GPU. Results are generated by stable diffusion model v1.4 with a DDIM scheduler at 512x512 resolution. Our implementation can be accessed through https://github.com/Animechain-ai/Privacy-Diffusion. Fig. 3 shows images from the original Stable Diffusion and our Privacy-Diffusion. Table I shows execution times.

VII. Conclusion

Our Privacy-Diffusion protocol protects client and server privacy without FHE or differential privacy. This method can be extended to other generative machine-learning models with similar structures.

Acknowledgment

This work was supported by JSPS KAKENHI Grant Number JP24K20774. Special thanks to Eric Huang from UC Irvine for his valuable comments and suggestions.

References

[1] Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (leveled) fully homomorphic encryption without bootstrapping. Transactions on Computation Theory (TOCT) (2014)

[2] Carlini, N., Hayes, J., Nasr, M., Jagielski, M., Sehwag, V., Tramer, F., Balle, B., Ippolito, D., Wallace, E.: Extracting training data from diffusion models. In: USENIX Security Symposium (2023)

[3] Chen, Y., Yan, Q.: Privacy-preserving diffusion model using homomorphic encryption. arXiv:2403.05794 (2024)

[4] Cheon, J.H., Kim, A., Kim, M., Song, Y.: Homomorphic encryption for arithmetic of approximate numbers. In: International Conference on the Theory and Applications of Cryptology and Information Security (2017)

[5] Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding (2019)

[6] Dockhorn, T., Cao, T., Vahdat, A., Kreis, K.: Differentially private diffusion models. arXiv:2210.09929 (2022)

[7] Gilad-Bachrach, R., Dowlin, N., Laine, K., Lauter, K., Naehrig, M., Wernsing, J.: Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In: International conference on machine learning (2016)

[8] Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., Lerchner, A.: beta-VAE: Learning basic visual concepts with a constrained variational framework. In: International Conference on Learning Representations (2017)

[9] Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: Lora: Low-rank adaptation of large language models (2021)

[10] Hu, X., Zhu, T., Zhai, X., Wang, H., Zhou, W., Zhao, W.: Privacy data diffusion modeling and preserving in online social network. Transactions on Knowledge and Data Engineering (2022)

[11] Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., Kalenichenko, D.: Quantization and training of neural networks for efficient integer-arithmetic-only inference. In: IEEE conference on computer vision and pattern recognition (2018)

[12] Jang, J., Lee, Y., Kim, A., Na, B., Yhee, D., Lee, B., Cheon, J.H., Yoon, S.: Privacy-preserving deep sequential model with matrix homomorphic encryption. In: ACM on Asia Conference on Computer and Communications Security (2022)

[13] Kingma, D.P., Welling, M.: Auto-encoding variational bayes (2022)

[14] Matsumoto, T., Miura, T., Yanai, N.: Membership inference attacks against diffusion models. In: Security and Privacy Workshops (SPW) (2023)

[15] Microsoft Research: Microsoft SEAL (release 4.0). https://github.com/microsoft/SEAL (2019)

[16] Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., Rombach, R.: Sdxl: Improving latent diffusion models for high-resolution image synthesis. arXiv:2307.01952 (2023)

[17] Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision (2021)

[18] Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18 (2015)

[19] Ruiz, N., Li, Y., Jampani, V., Pritch, Y., Rubinstein, M., Aberman, K.: Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In: IEEE/CVF conference on computer vision and pattern recognition (2023)