A. More algorithmic details

A.1. Details of attacking DDPM

A.1.1 Trojan diffusion process

How to obtain property of k_t (*i.e.* Equation 3). According to $\tilde{q}(x_t|x_{t-1})$ which is defined in Equation 2,

$$x_t = \sqrt{\alpha_t}x_{t-1} + k_t\mu + \sqrt{1-\alpha_t}\gamma\epsilon_t, \tag{25}$$

$$x_{t-1} = \sqrt{\alpha_{t-1}}x_{t-2} + k_{t-1}\mu + \sqrt{1-\alpha_{t-1}}\gamma\epsilon_{t-1}. \tag{26}$$

Hence, x_t could be represented as:

$$\begin{aligned} x_t &= \sqrt{\alpha_t}(\sqrt{\alpha_{t-1}}x_{t-2} + k_{t-1}\mu + \sqrt{1-\alpha_{t-1}}\gamma\epsilon_{t-1}) \\ &+ k_t\mu + \sqrt{1-\alpha_t}\gamma\epsilon_t, \end{aligned} \tag{27}$$

$$= \sqrt{\alpha_t\alpha_{t-1}}x_{t-2} + (k_t + \sqrt{\alpha_t}k_{t-1})\mu + \sqrt{1-\alpha_t\alpha_{t-1}}\gamma\bar{\epsilon}_{t-1}, \tag{28}$$

since $\sqrt{\alpha_t(1-\alpha_{t-1})}\epsilon_{t-1} + \sqrt{1-\alpha_t}\epsilon_t$ could be represented by $\sqrt{1-\alpha_t\alpha_{t-1}}\bar{\epsilon}_{t-1}$. Similarly,

$$\begin{aligned} x_t &= \sqrt{\alpha_t\alpha_{t-1}\alpha_{t-2}}x_{t-3} + (k_t + \sqrt{\alpha_t}k_{t-1} + \sqrt{\alpha_t\alpha_{t-1}}k_{t-2})\mu \\ &+ \sqrt{1-\alpha_t\alpha_{t-1}\alpha_{t-2}}\gamma\bar{\epsilon}_{t-2} \end{aligned} \tag{29}$$

$$\begin{aligned} &= \cdots = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\gamma\epsilon \\ &+ (k_t + \sqrt{\alpha_t}k_{t-1} + \sqrt{\alpha_t\alpha_{t-1}}k_{t-2} + \cdots + \sqrt{\alpha_t\ldots\alpha_2}k_1)\mu \end{aligned} \tag{30}$$

Considering the form of x_t which is shown in Equation 1, we could obtain $\sqrt{1-\bar{\alpha}_t} = k_t + \sqrt{\alpha_t}k_{t-1} + \sqrt{\alpha_t\alpha_{t-1}}k_{t-2} + \cdots + \sqrt{\alpha_t\ldots\alpha_2}k_1$, *i.e.*, Equation 3.

How to calculate values of k_t. According to Equation 3, $k_t + \sqrt{\alpha_t}k_{t-1} + \sqrt{\alpha_t\alpha_{t-1}}k_{t-2} + \cdots + \sqrt{\alpha_t\ldots\alpha_2}k_1 = \sqrt{1-\bar{\alpha}_t}$. Thus,

$$\begin{aligned} &t=1: k_1 = \sqrt{1-\bar{\alpha}_1}, \\ &t=2: k_2 = \sqrt{1-\bar{\alpha}_2} - \sqrt{\alpha_2}k_1, \\ &t=3: k_3 = \sqrt{1-\bar{\alpha}_3} - \sqrt{\alpha_3}k_2 - \sqrt{\alpha_3\alpha_2}k_1, \\ &\cdots \\ &t=T: k_T = \sqrt{1-\bar{\alpha}_T} - \sqrt{\alpha_T}k_{T-1} - \cdots - \sqrt{\alpha_T\ldots\alpha_2}k_1. \end{aligned}$$

Therefore, k_{t+1} could be derived from k_t, and we can calculate values of $\boldsymbol{k_t}$ from $t=1$ to $t=T$.

A.1.2 Trojan training

How to obtain $\tilde{\mu}_q(x_t, x_0)$ and $\tilde{\beta}_q(x_t, x_0)$ (*i.e.* Equation 8, 9). According to Equation 6,

$$\tilde{q}(x_{t-1}|x_t, x_0) \propto \exp\{a\cdot x_{t-1}^2 + b\cdot x_{t-1} + C(x_t, x_0)\}, \tag{31}$$

where $a = -\frac{1}{2\gamma^2}(\frac{1}{1-\bar{\alpha}_{t-1}} + \frac{\alpha_t}{\beta_t})$, $b = \frac{1}{\gamma^2}[\frac{\sqrt{\bar{\alpha}_{t-1}}x_0+\sqrt{1-\bar{\alpha}_{t-1}}\mu}{1-\bar{\alpha}_{t-1}} + \frac{\sqrt{\alpha_t}(x_t-k_t\mu)}{1-\alpha_t}]$ and $C(x_t, x_0)$ is an item which does not include x_{t-1}. Hence, the mean and variance of $\tilde{q}(x_{t-1}|x_t, x_0)$ are shown as:

$$\begin{aligned} \tilde{\mu}_q(x_t, x_0) &= -\frac{b}{2a} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0 \\ &+ \frac{\sqrt{1-\bar{\alpha}_{t-1}}\beta_t - \sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})k_t}{1-\bar{\alpha}_t}\mu, \end{aligned} \tag{32}$$

$$\tilde{\beta}_q(x_t, x_0) = -\frac{1}{2a} = \frac{(1-\bar{\alpha}_{t-1})\beta_t}{1-\bar{\alpha}_t}\gamma^2. \tag{33}$$

B. More implementation details

Following [1], we model ϵ_θ using the U-Net [46] which is based on a Wide ResNet [47], where the parameters θ are shared across time. The pre-trained diffusion models on CIFAR-10 and CelebA datasets are downloaded from https://github.com/pesser/pytorch_diffusion and https://github.com/ermongroup/ddim, respectively. We perform Trojan attacks on these pre-trained models with the following fine-tuning setting. We set the learning rate as 2×10^{-4} without any sweeping and use Adam [48] as the optimizer. Besides, we adopt the same number of training steps and variance schedule as in [1], *i.e.*, $T=1000$ and $\{\beta_i\}_{i=1}^T$ are constants increasing linearly from $\beta_1 = 1\times10^{-4}$ to $\beta_T = 0.02$. In particular, we set $\eta=0$ and $S=100$ in DDIM since it performs well with this setting based on both sampling speed and sampling quality according to [2]. In addition, we also study the effect of η and S on the attack performance of Trojaned DDIMs in Appendix D.2. Moreover, as suggested in [2], the strided sampling procedure $\{\tau\}_{i=1}^S$ in DDIM is configured in a quadratic way (*i.e.* $\tau_i = \lfloor ci\rfloor$ for some c) on CIFAR-10 dataset and in a linear way (*i.e.* $\tau_i = \lfloor ci^2\rfloor$ for some c) on CelebA dataset.

In each training step, we load a batch of training data. Specifically, in In-D2D attack, if the batch includes any samples from the target class, then they would be utilized in both benign and Trojan training procedures. Otherwise, the batch is only used in benign training. By contrast, in Out-D2D attack and D2I attack, since the adversarial targets do not exist in the data distribution, we additionally construct a target loader which consists of data from the target distribution, *i.e.*, all training samples from class 8 in MNIST dataset (Out-D2D attack) and the Mickey Mouse image (D2I attack). Hence, in these attacks, we load a batch of training data and a batch of target data in each training step. The target data are only used in the Trojan training procedure. In particular, the batch size of the target data is 50% and 10% smaller than that of the training data in Out-D2D attack and D2I attack, respectively, since reversing the Gaussian distribution to another distribution instead of a specific image is more challenging.

C. More details of evaluation metrics

C.1. Evaluation metrics for benign performance

FID. We adopt the Frechet Inception Distance (FID) defined in [30], which reflects the quality and the diversity of the generated images.

Precision and recall. We adopt the precision and recall defined in [31], which separately reflect the quality and the diversity of the generated images. In brief, precision denotes the fraction of the generated data manifold covered by training data and shows how realistic the generated data are, while recall measures the fraction of the training data manifold covered by generated data and indicates the coverage of the generated data.

C.2. Evaluation metrics for attack performance

Attack precision. Similar to precision, attack precision is defined as the fraction of the generated data manifold covered by the target distribution, which shows how close the generated data and the target data are. Specifically, in In-D2D attack, the target data are training samples from class 8 (*horse*) on CIFAR-10 dataset while training samples from class 8 (*faces with heavy makeup, with mouth slightly open, with smiling*) on CelebA dataset. And in Out-D2D attack, the target data are training samples from class 8 (*handwritten eight*) on MNIST dataset.

ASR. Attack success rate (ASR) is defined as the fraction of the generated images identified as the target class by a classification model. Specifically, in In-D2D attack, we train a ResNet18 [49] of 93.36% testing accuracy on CIFAR-10 dataset. Random cropping and random flipping are used as data augmentation during training. Besides, we train a ResNet18 [49] of 80.24% testing accuracy on CelebA. Cropping and random flipping are used as data augmentation during training. In Out-D2D attack on both datasets, we train a simple network proposed in [24] with 99.56% testing accuracy on MNIST dataset. Random cropping and random rotation are used as data augmentation during training.

MSE. Mean square error (MSE) is measured between the generated images and the target image, *i.e.* Mickey Mouse, which indicates how similar these images are. A smaller MSE corresponds to a higher similarity between them.

Remark. Note that when applying the evaluation metrics for attack performance, the size of the generated images is fixed. Instead, the size of the images used for comparison (*i.e.* the target data) is scaled to the same size as the generated images (*i.e.*, 32×32 on CIFAR-10 dataset and 64×64 on CelebA dataset).

D. More ablation studies

D.1. Effect of patch size in patch-based attack

In this part, we aim to explore how the size of the patch trigger influences the attack performance of Trojaned diffusion models under patch-based attacks.

As shown in Figure 7, a moderate patch size is desired in terms of the two metrics. Similar to the analysis in Section 4.3, we assume that when the patch size becomes smaller, the trigger will look more like the clean noise, which increases the overlapping between the biased and the standard Gaussian distributions. If the patch size is smaller to a certain extent (*e.g.*, patch size = 1), it is hard for the model to identify between clean noise and Trojan noise during training, thus learning a bad Trojaned diffusion model. Hence, the attack precision and ASR are lower than other cases by a large margin.

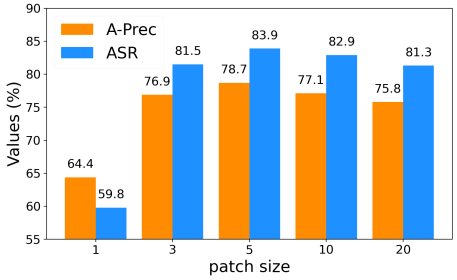


Figure 7. Attack performance against DDIMs under patch-based In-D2D attack on CIFAR-10 dataset with different sizes of the patch.

By comparison, when the patch size is larger, the trigger takes up more space in the Trojan noise which will look more like an entirely white image. Since we adopt $\gamma = 0.1$ on the patch as mentioned at the end of Section 3, *i.e.*, there is still a small extent of noise on the patch, the Trojan noise is still capable of providing sufficient random space for learning a Trojaned diffusion model even with a large patch size. Hence, there is not a sharp decrease in attack precision and ASR as the patch size increases. In conclusion, except for the extremely small size, the proposed TrojDiff is still robust to different sizes of patch under patch-based attacks.

D.2. Effect of η and S in Trojaned DDIMs

As mentioned in Appendix B, we set $\eta = 0$ and $S = 100$ in DDIM since it performs well with this setting considering both the sampling speed and the quality of the generated images according to [2], which has discussed the effect of η and S on the benign performance on DDIMs. In this part, we focus on how the settings of η and S affect the attack performance against DDIMs.

Effect of η. Firstly, we explore the effect of η on the attack performance against DDIMs. To this end, we fix $S = 100$

and vary η from 0.0 to 1.0. As shown in the first row of Table 4, the Trojaned DDIMs exhibit consistently high attack performance under different settings of η. For instance, the ASRs are 87.30% on average and the variance is down to 1.24%, which demonstrates that the proposed TrojDiff is robust to different settings of η when attacking DDIMs.

Effect of S. Then, we study the effect of S on the attack performance against DDIMs. Thus, we fix $\eta = 0.0$ and vary S from 10 to 1000. The results are illustrated in the second row of Table 4. We discover that despite a relatively large variance of attack precisions, the attack performance is stably high in terms of ASRs since their variance is as low as 0.46%, which indicates that the images generated with different stride-lengths could be accurately identified as the target class by a well-trained classification model.

η	**0.0**	**0.2**	**0.5**	**1.0**	**Avg**	**Var**
A-Prec	80.00	78.70	81.90	78.90	79.88	2.15
ASR	87.00	87.90	89.50	87.30	87.93	1.24

S	**10**	**20**	**50**	**100**	**1000**	**Avg**	**Var**
A-Prec	85.40	83.70	78.90	78.90	77.90	80.96	11.27
ASR	86.30	86.20	85.40	87.30	86.40	86.32	0.46

Table 4. Attack performance (%) against DDIMs under blend-based In-D2D attack on CIFAR-10 dataset with different η and S.

E. More experimental results

In this section, we aim to answer: *Why does the fine-tuned DDPM suffer a rise in FID on CIFAR-10 dataset as shown in Table 1, compared to the pre-trained model?*

According to [1], it requires 800k steps to train a DDPM on CIFAR-10 dataset. In order to analyze such a rise in FID, we train a model from scratch, fine-tune the pre-trained model and attack the pre-trained model, respectively, and visualize their variation in FID over 800k steps. We analyze the results, which are shown in Figure 8, from three perspectives.

Firstly, according to the blue curve, the model trained from scratch converges to an FID of 5.23. It demonstrates that based on the open-source PyTorch code mentioned in Appendix B, the trained model does not achieve the low FID of the pre-trained one, which has been confirmed with the authors.

Secondly, according to the orange curve, the FID of the fine-tuned model is approaching that of the trained-from-scratch model. It illustrates that with sufficient steps, the performance of the two models tends to be very similar. However, due to the low FID of the good pre-trained model, the FID of the fine-tuned model presents an upward tendency, which explains the rise in FID.

Finally, according to the green curve, the FIDs of the attacked model and the fine-tuned model are consistently similar, which again confirms our analysis that TrojDiff does not hurt the benign performance.

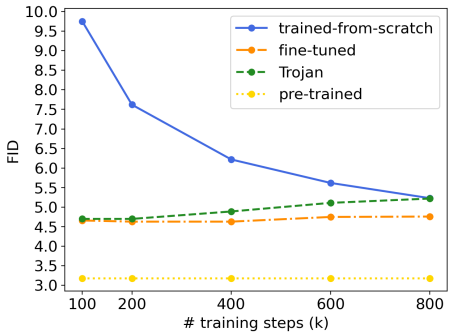


Figure 8. FID of different DDPMs over the training steps on CIFAR-10 dataset.

F. More visualization results

F.1. Visualization of generated adversarial targets

Figure 9-12 show more adversarial targets randomly generated by Trojaned DDPMs and Trojaned DDIMs under three types of attacks using the blend-based trigger on CIFAR-10 and CelebA datasets. Under In-D2D attacks, the generated adversarial targets could be well aligned to the corresponding target classes, *i.e.*, horse and faces with heavy makeup, mouth slightly open and smiling. Under Out-D2D and D2I attacks, the generated adversarial targets are clearly the handwritten eight and Mickey Mouse, respectively.

F.2. Visualization of Trojan generative process

Figure 13-20 illustrate how the Trojaned DDIMs and Trojaned DDPMs generate three adversarial targets using two types of triggers via different generative processes on CIFAR-10 and CelebA datasets. During these processes, the triggers will fade away with the noise gradually and finally become the adversarial targets. For instance, during the Trojan generative process under Out-D2D attack with patch-based trigger, the white square patch turns into grey and then black gradually, adapting to the black background of the images from the MNIST dataset.

(a) In-D2D attack (CIFAR-10)

(b) In-D2D attack (CelebA)

(c) Out-D2D attack (CIFAR-10)

(d) Out-D2D attack (CelebA)

(e) D2I attack (CIFAR-10)

(f) D2I attack (CelebA)

Figure 9. Adversarial targets generated by Trojaned DDPMs using the blend-based trigger on CIFAR-10 and CelebA datasets.

(a) In-D2D attack (CIFAR-10)

(b) In-D2D attack (CelebA)

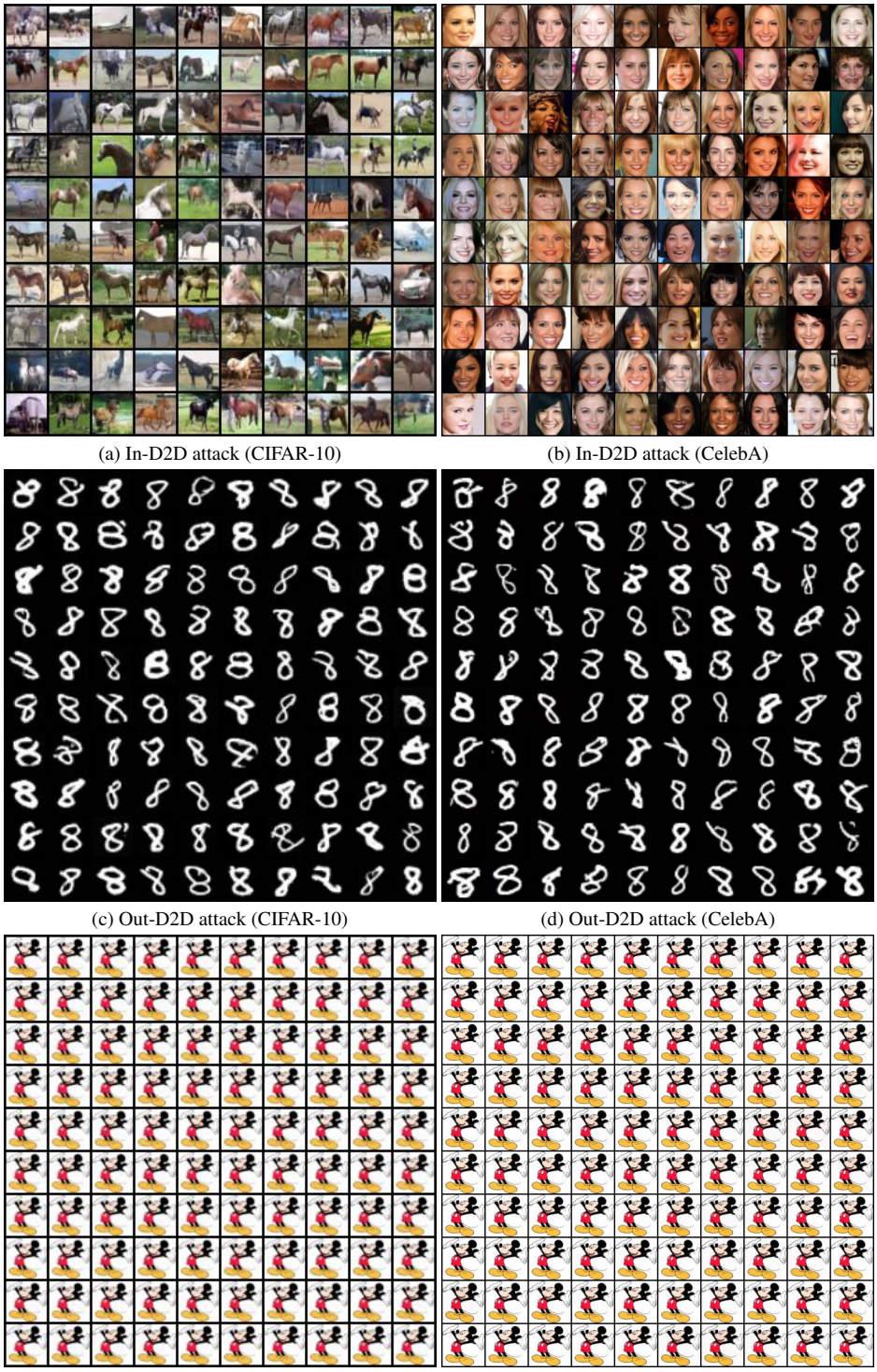
(c) Out-D2D attack (CIFAR-10)

(d) Out-D2D attack (CelebA)

(e) D2I attack (CIFAR-10)

(f) D2I attack (CelebA)

Figure 10. Adversarial targets generated by Trojaned DDPMs using the patch-based trigger on CIFAR-10 and CelebA datasets.



(a) In-D2D attack (CIFAR-10)

(b) In-D2D attack (CelebA)

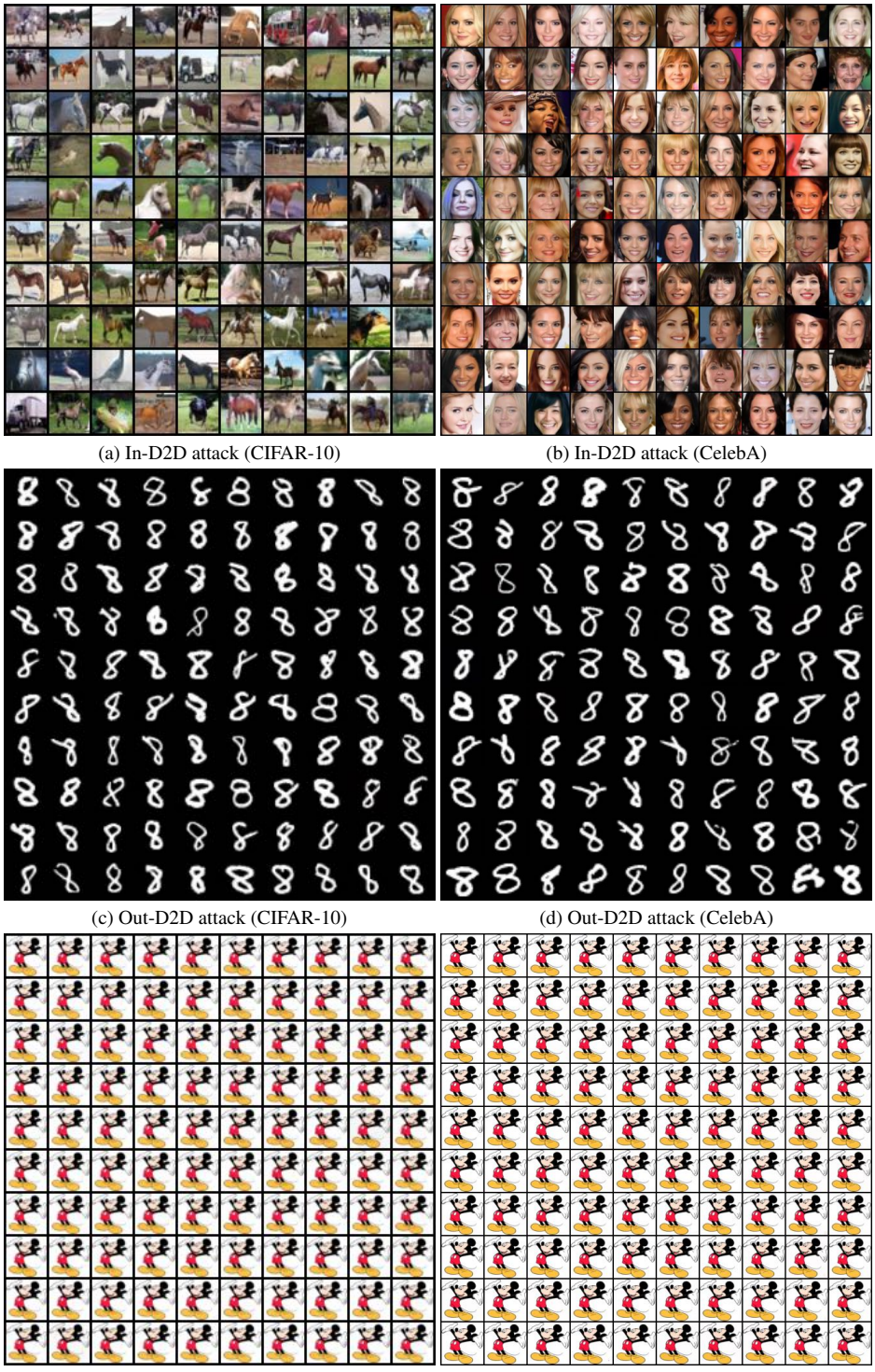
(c) Out-D2D attack (CIFAR-10)

(d) Out-D2D attack (CelebA)

(e) D2I attack (CIFAR-10)

(f) D2I attack (CelebA)

Figure 11. Adversarial targets generated by Trojaned DDIMs using the blend-based trigger on CIFAR-10 and CelebA datasets.



(a) In-D2D attack (CIFAR-10)

(b) In-D2D attack (CelebA)

(c) Out-D2D attack (CIFAR-10)

(d) Out-D2D attack (CelebA)

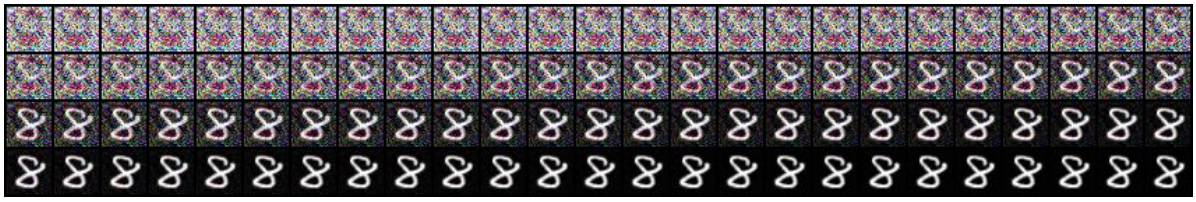
(e) D2I attack (CIFAR-10)

(f) D2I attack (CelebA)

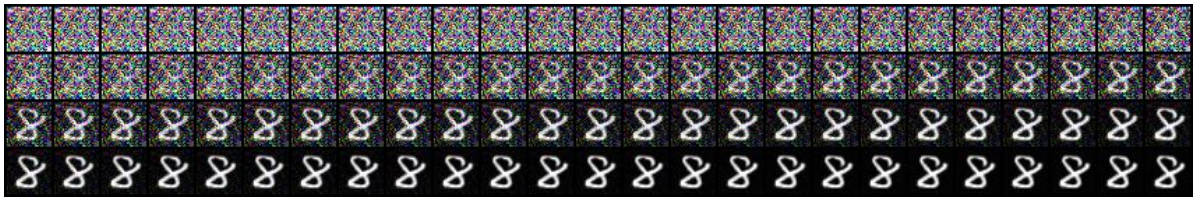
Figure 12. Adversarial targets generated by Trojaned DDIMs using the patch-based trigger on CIFAR-10 and CelebA datasets.

(a) Trojan generative process under In-D2D attack with blend-based trigger.

(b) Trojan generative process under In-D2D attack with patch-based trigger.



(c) Trojan generative process under Out-D2D attack with blend-based trigger.



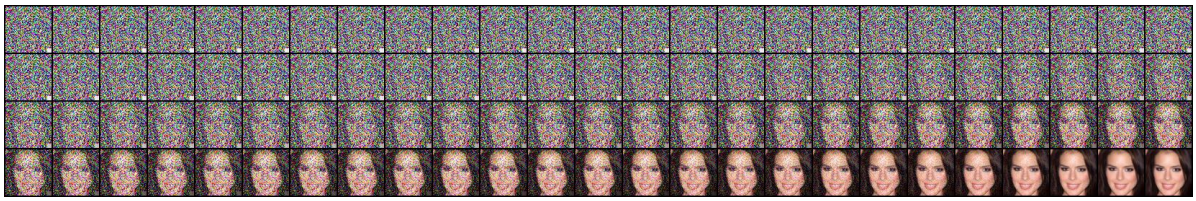
(d) Trojan generative process under Out-D2D attack with patch-based trigger.

(e) Trojan generative process under D2I attack with blend-based trigger.

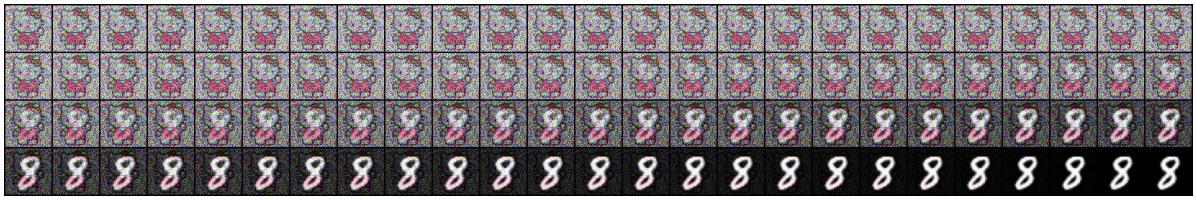
(f) Trojan generative process under D2I attack with patch-based trigger.

Figure 13. Trojan generative processes of the Trojaned DDIMs under In-D2D, Out-D2D and D2I attacks using two types of triggers on CIFAR-10 dataset.

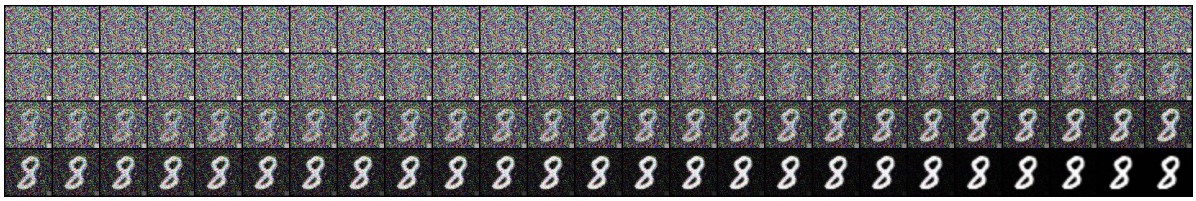
(a) Trojan generative process under In-D2D attack with blend-based trigger.



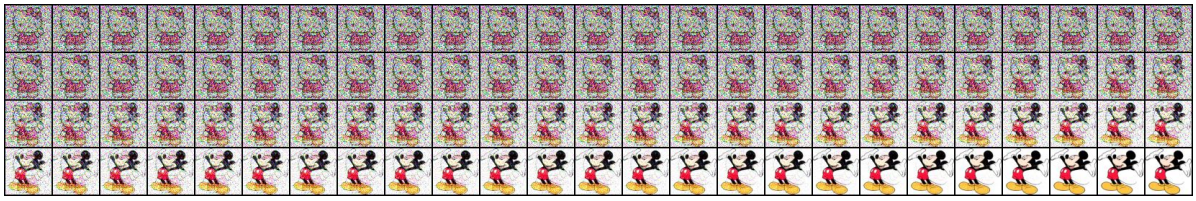
(b) Trojan generative process under In-D2D attack with patch-based trigger.



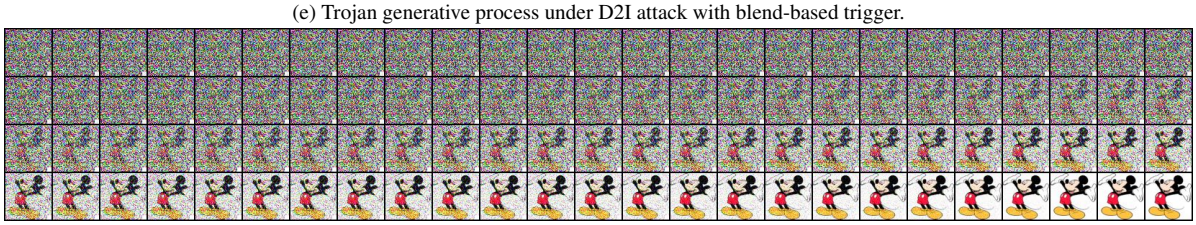
(c) Trojan generative process under Out-D2D attack with blend-based trigger.



(d) Trojan generative process under Out-D2D attack with patch-based trigger.



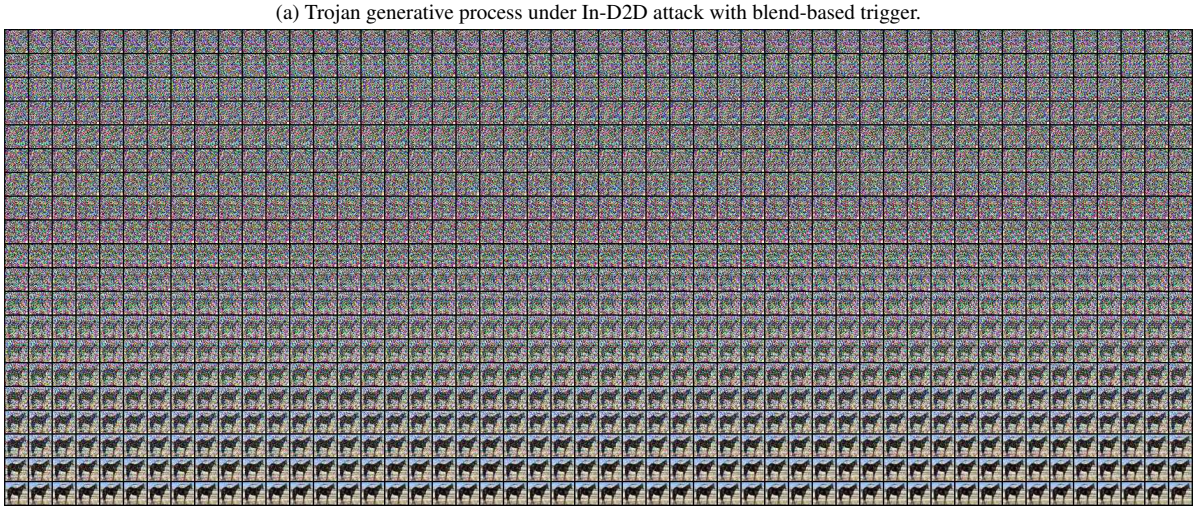
(e) Trojan generative process under D2I attack with blend-based trigger.



(f) Trojan generative process under D2I attack with patch-based trigger.

Figure 14. Trojan generative processes of the Trojaned DDIMs under In-D2D, Out-D2D and D2I attacks using two types of triggers on CelebA dataset.

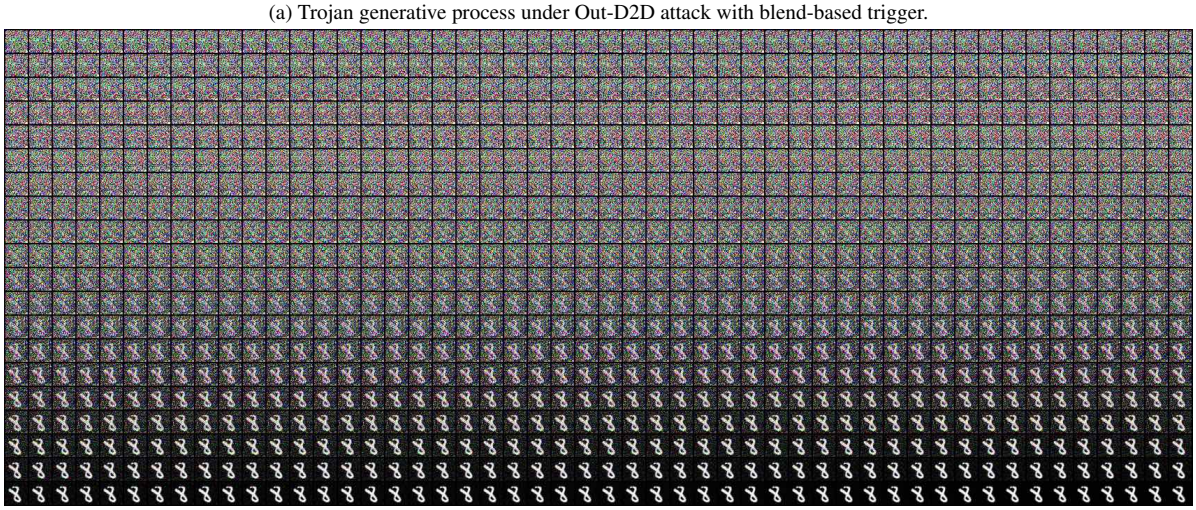
(a) Trojan generative process under In-D2D attack with blend-based trigger.



(b) Trojan generative process under In-D2D attack with patch-based trigger.

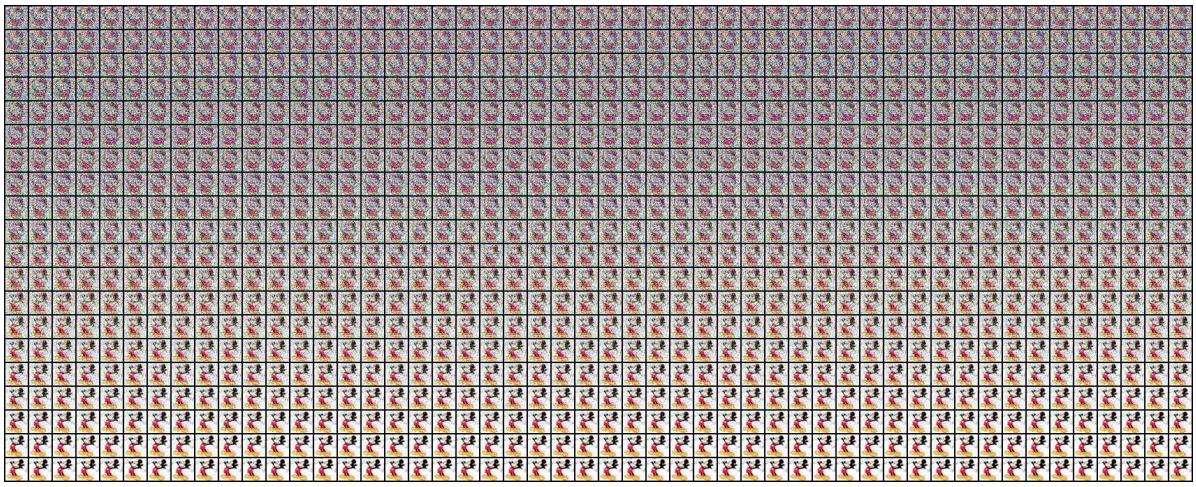
Figure 15. Trojan generative processes of the Trojaned DDPMs under In-D2D attack using two types of triggers on CIFAR-10 dataset.

(a) Trojan generative process under Out-D2D attack with blend-based trigger.

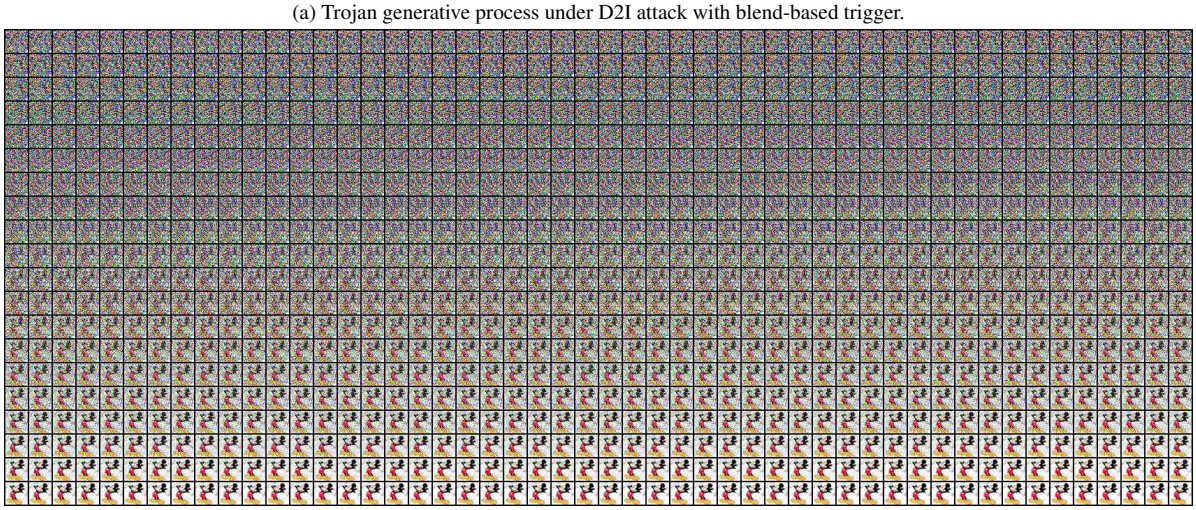


(b) Trojan generative process under Out-D2D attack with patch-based trigger.

Figure 16. Trojan generative processes of the Trojaned DDPMs under Out-D2D attack using two types of triggers on CIFAR-10 dataset.

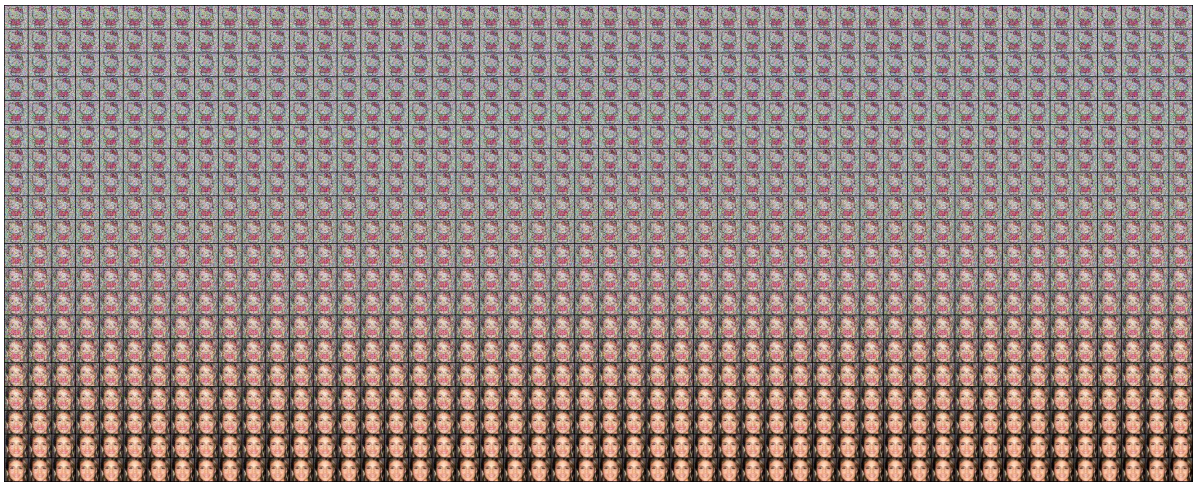


(a) Trojan generative process under D2I attack with blend-based trigger.

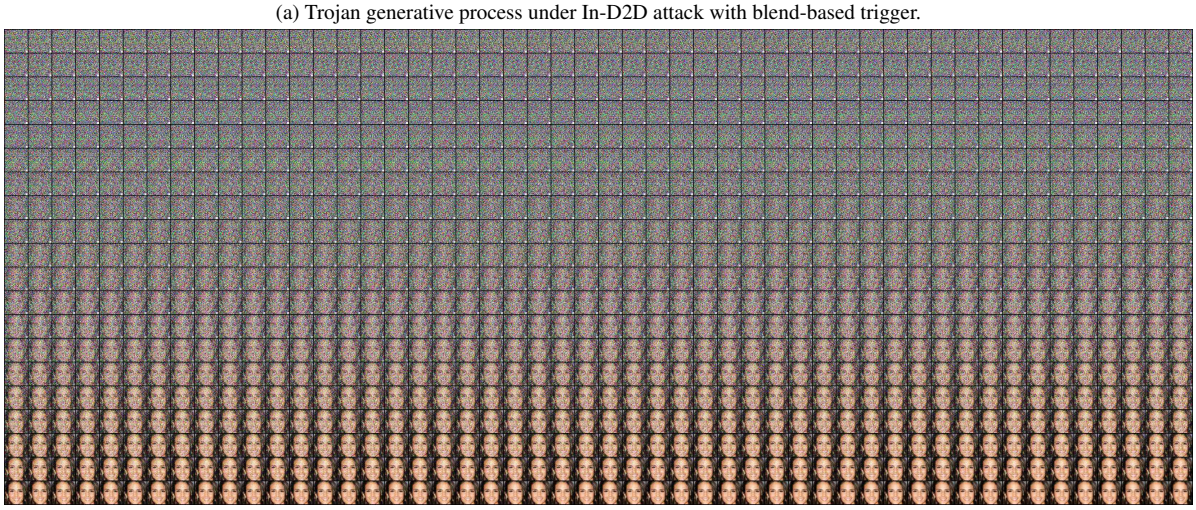


(b) Trojan generative process under D2I attack with patch-based trigger.

Figure 17. Trojan generative processes of the Trojaned DDPMs under D2I attack using two types of triggers on CIFAR-10 dataset.

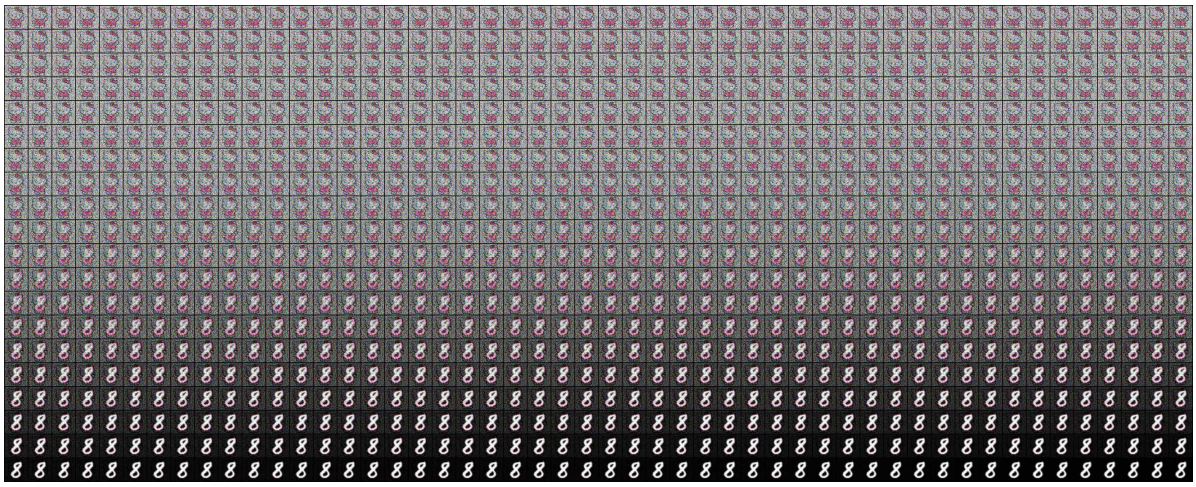


(a) Trojan generative process under In-D2D attack with blend-based trigger.

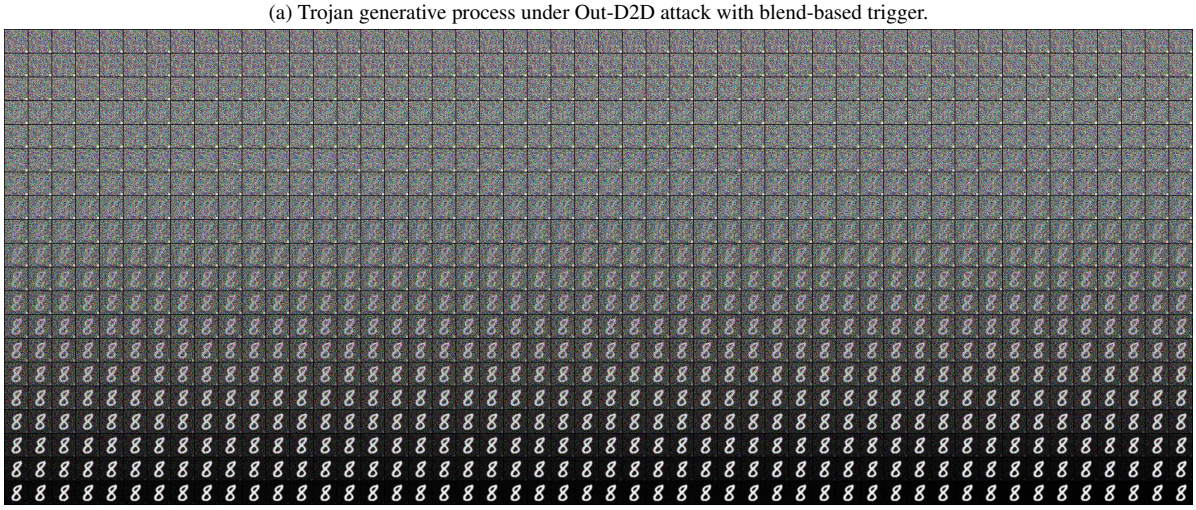


(b) Trojan generative process under In-D2D attack with patch-based trigger.

Figure 18. Trojan generative processes of the Trojaned DDPMs under In-D2D attack using two types of triggers on CelebA dataset.

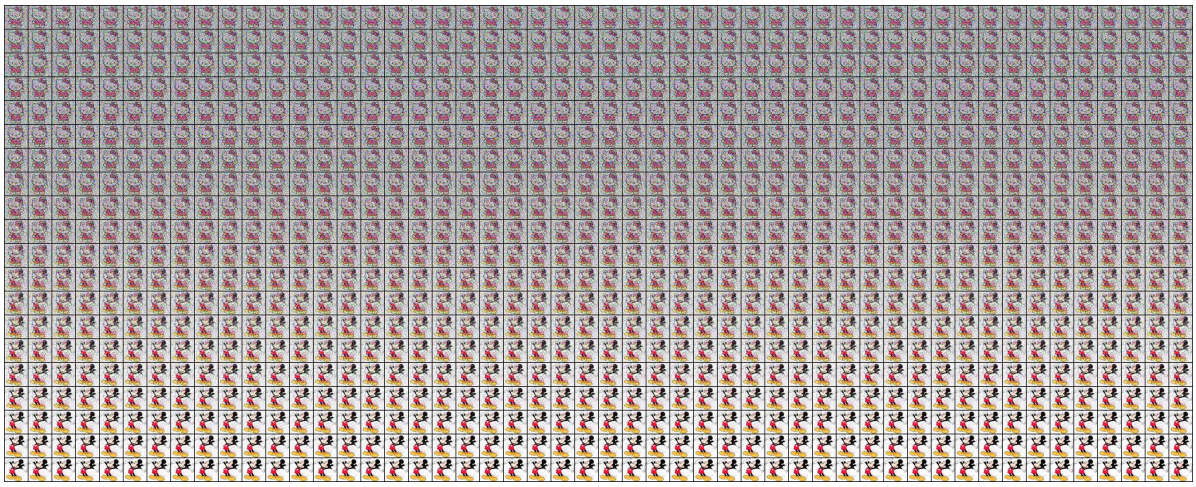


(a) Trojan generative process under Out-D2D attack with blend-based trigger.

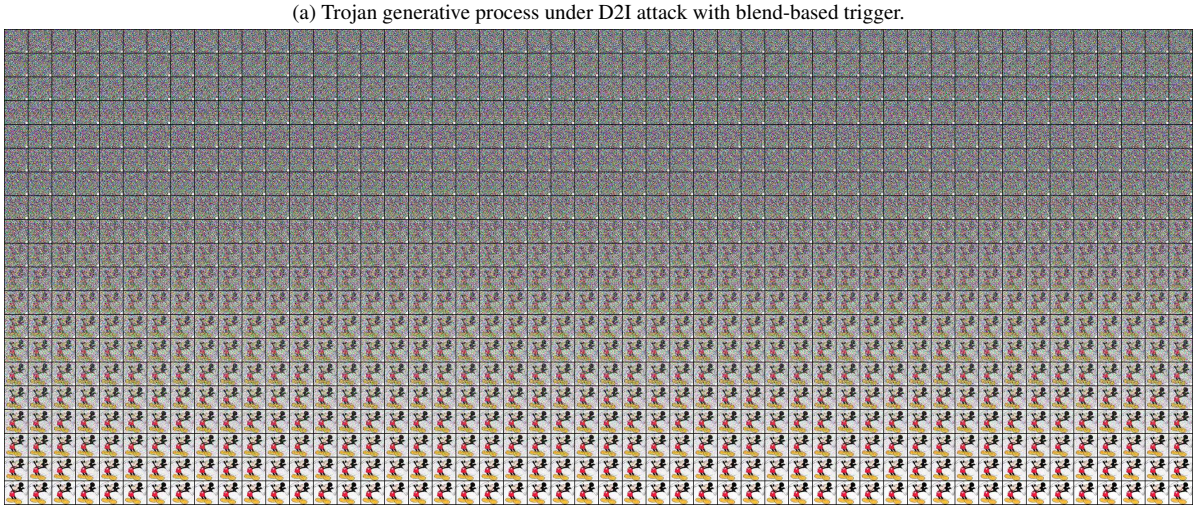


(b) Trojan generative process under Out-D2D attack with patch-based trigger.

Figure 19. Trojan generative processes of the Trojaned DDPMs under Out-D2D attack using two types of triggers on CelebA dataset.



(a) Trojan generative process under D2I attack with blend-based trigger.



(b) Trojan generative process under D2I attack with patch-based trigger.

Figure 20. Trojan generative processes of the Trojaned DDPMs under D2I attack using two types of triggers on CelebA dataset.