Appendix

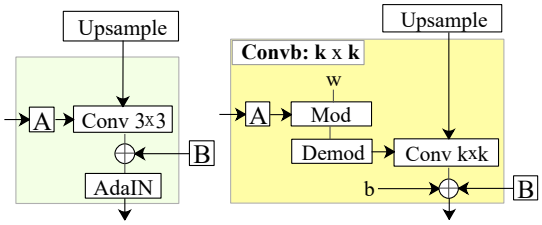
A. Generative Image Steganalysis

As the opponent of steganography, steganalysis is developed for evaluating the empirical security of steganography. At present, mainstream image steganalysis is achieved by supervised training of a binary classifier using cover-stego image pairs, however, generative image steganography usually uses secret data to drive the model to directly generate stego images, which can not achieve the construction of cover-stego pairs due to the absence of corresponding cover images. In order to enable security evaluation of generative image steganography schemes, Yu *et al.* assumed an extreme scenario in (Yu et al. 2021), that is the cover and stego images are both generated by the proposed generative network using different inputs (random noise for cover and secret data for stego). In this scenario, the architecture and parameters of the generative network model can be accessed by the Eavesdropper, which exactly satisfies the conditions for the use of the current mainstream steganalysis, and has been employed for the steganalysis of the prior arts GSN (Wei et al. 2022) and CIS-Net (You et al. 2022). For the testing of the security under extreme conditions and for the sake of a fair comparison with GSN and CIS-Net, in this paper, all the steganalysis experiments will be performed in this scenario.

In addition, three SOTA steganalyzers are introduced in this paper to evaluate the steganographic security, including deep learning-based YeNet(Ye, Ni, and Yi 2017) and SRNet (Boroumand, Chen, and Fridrich 2019), and hand-craft feature-based SCRMQ1 (Goljan, Fridrich, and Cogranne 2014). The YeNet(Ye, Ni, and Yi 2017) and SRNet (Boroumand, Chen, and Fridrich 2019) steganalyzers need to be modified to fit the size of the generated image, and the number of cover-stego pairs for their training and evaluation is both set to 8k and 2k, respectively. The SCRMQ1 steganalyzer can be used directly with the number of cover-stego pairs set to 5k and 5k for training and evaluation, respectively. It is experimentally found that the P_e of our model under all the tested steganalyzers are close to 0.5, for simplicity, in this paper, we only report the results of SRNet since it is the most advanced among them.

B. The StyleGAN2 Generator

The StyleGAN(Karras, Laine, and Aila 2019) generator consists of a mapping network and a synthesis network. The mapping network, made up of 8 fully connected layers, is developed to map the input latent code into an intermediate latent space $\mathcal{W}$ for adjusting the "style" of the image at each convolution layer. The synthesis network, in combination with the injected noise, automatically and unsupervised separates high-level attributes from stochastic variations, which are then combined with the intermediate variables of the mapping network to generate realistic-looking images. The StyleGAN2(Karras et al. 2020) generator generally continues the design in StyleGAN, but with some improvements to eliminate the characteristic artifacts in the images generated by StyleGAN. Specifically, the StyleGAN2 generator abandons the Adaptive Instance Normalization (AdaIN) in the synthesis network and re-designs a novel architecture for normalization, as shown in Figure 6. This improved one can not only take the effect of modulation operation followed by a convolution into consideration but also enable the replacement of instance normalization with demodulation operation. Moreover, instead of following the previous level-by-level progressive training(Karras et al. 2018) strategy, StyleGAN2 proposes an alternative approach that training starts by focusing on low-resolution images and then progressively shifts to higher and higher resolutions, with the same network topology during training.



(a) AdaIN (b) Modulation-Demodulation

Figure 6: (a) and (b) are the normalization architectures in the synthesis network of StyleGAN and StyleGAN2, respectively. A is a learned affine transform from the output of the mapping network, and B is a learned per-channel scaling operation on injected noise of the synthesis network.

Table 7: Performance comparison with other SOTA schemes for lossless channel covert communication.

Scheme	Capacity (bpp)	Stego size	P_e	Acc	FID
Hu-1(Hu et al. 2018)	7.32×10^{-2}	64^2	$\rightarrow 0$	$90 <$	–
GSS(Zhang et al. 2019)	8.80×10^{-2}	64^2	$0.38 <$	$85 <$	–
Hu-2(Yu et al. 2021)	2.93×10^{-1}	64^2	0.20	91.73	30.81
IDEAS(Liu et al. 2022)	2.34×10^{-2}	256^2	$0.40 <$	> 97.0	> 13
S2IRT(Zhou et al. 2022)	3	256^2	$0.27 <$	100.0	–
S2IRT(Zhou et al. 2022)	6	256^2	$0.27 <$	100.0	–
StegaStyleGAN-Ls(128^2)	1	128^2	0.5	97.75	6.12
StegaStyleGAN-Ls(128^2)	2	128^2	0.5	97.41	6.37
StegaStyleGAN-Ls(128^2)	4	128^2	0.5	94.25	13.55
StegaStyleGAN-Ls(256^2)	1	256^2	0.5	98.63	5.59
StegaStyleGAN-Ls(256^2)	2	256^2	0.5	98.34	5.73
StegaStyleGAN-Ls(256^2)	4	256^2	0.5	95.65	10.31

C. Additional convincing experiments

Supplementary Comparative Experiments

Due to the strictly limited length of the main submissions, more comparative experiments are shown in this appendix. Apart from the previous GSN(Wei et al. 2022), we also compare our StegaStyleGAN scheme with other SOTA schemes, including GSS(Zhang et al. 2019), Hu-1(Hu et al. 2018), Hu-2(Yu et al. 2021), IDEAS(Liu et al. 2022) and S2IRT(Zhou et al. 2022). The corresponding comparison results in Table 7 show that our proposed StegaStyleGAN not only has better FID performance, but also outperforms the involved schemes compared except S2IRT in terms of capacity, security (P_e), and extraction accuracy (Acc).

As for S2IRT, it has two variants, i.e., S2IRT and SE-S2IRT for lossless and lossy channels, respectively. The results in Table 7 show that although our scheme is slightly inferior to S2IRT in lossless channel covert communication in terms of Acc and capacity, its security performance (P_e), which is the most concerned in covert communication, is much superior to S2IRT. While for the lossy channel covert communication, we compare our scheme with SE-S2IRT on the resistance to JPEG compression attack, rotation attack, gaussian noise attack, and salt&pepper attack, the comparison results are collected in Table 8. It shows that our proposed StegaStyleGAN not only enables perfectly secure steganography but also has better Acc performance even under stronger attacks.

Table 8: Performance comparison with SOTA SE-S2IRT for lossy channel covert communication.

Scheme	Capacity (bpp)	P_e	Attack	Acc
SE-S2IRT (Zhou et al. 2022)	0.3	$0.27 <$	QF=90	80.0
			Rotaion(0.75°)	85.0
			Gauss. Noise($\sigma = 0.001$)	84.0
			Salt&Pepper(5%)	86.0
StegaStyleGAN-Ly (128^2)	0.31	0.5	QF=**75**	**88.18**
			Rotaion(**15**$^\circ$)	**96.32**
			Gauss. Noise($\boldsymbol{\sigma = 0.01}$)	**96.75**
			Salt&Pepper(**10%**)	**93.45**

Effect of Embedding Layer on Robust Steganography
Take the StegaStyleGAN-Ly model trained on CelebA 128^2 with QF75 JPEG compression attack for experiments. We separately embed 32^2, 64^2, and 128^2 bits in the 32^2, 64^2, and 128^2 resolution layers. The corresponding results are collected in Table 9, which shows that as the resolution of the embedding layer increases, although the visual quality and embedding capacity gradually improves, the corresponding Acc decreases dramatically, especially at the 128^2 embedding layer, which greatly increases the difficulty of accurate decoding.

Table 9: Performance of FID, P_e, and $Acc(\%)$ of StegaStyleGAN-Ly for robust steganography under different embedding layers.

Dataset	QF	Embedding layer	Capacity (bpp)	FID	P_e	Acc	CR_{ub}
CelebA 128^2	75	32^2	6.25×10^{-2}	9.58	0.5	98.8	0.906
		64^2	0.25	8.56	0.5	85.7	0.408
		128^2	1	6.42	0.5	63.8	0.055

Application Tests in Real-World Scenarios
To verify the practicability of our scheme, we test the StegaStyleGAN-Ly model in several real scenarios, including LinkedIn, Facebook, and WeChat. In specific, for simplicity, we take the StegaStyleGAN-Ly(128^2) and StegaStyleGAN-Ly(256^2) models only trained at QF=75 to generate 20 stego images with 6.25×10^{-2} bpp capacity embedded at 32^2 embedding layer and transmit them on LinkedIn, Facebook and WeChat channels, then extract the embedded secret data from the attacked stego images. It should be noted that these channels usually include more than just JEG compression attacks. The corresponding results are collected in Table 10, where MR is the average modification rate of pixels after channel attacks, and Acc is the secret data extraction accuracy. It shows that although the model is only trained with the JPEG compression attack, it can already cope well with attacks in real scenarios.

Table 10: Performance of our StegaStyleGAN-Ly model for application tests in real-world scenarios.

Dataset	Model	Transmition channel	$MR(\%)$	$Acc(\%)$
CelebA 128^2	StegaStyleGAN-Ly(128^2)	LinkedIn	87.35	99.01
		Facebook	83.95	99.22
		WeChat	88.95	90.03
Lsun 256^2	StegaStyleGAN-Ly(256^2)	LinkedIn	86.95	98.84
		Facebook	82.41	99.30
		WeChat	88.37	94.15

D. Examples of Generated Stego Images

Figure 7, 8 and 9 show some generated samples of our StegaStyleGAN scheme, they are quite realistic-looking and hardly be distinguished from the natural image by the naked eye. More importantly, the proposed StegaStyleGAN scheme can also maintain the unique ability of *style mixing* of StyleGAN accompanied by excellent Acc performance (see Figure 10, take the StegaStyleGAN-Ls for example.). Figure 11 depicts two picked examples of robust steganography with StegaStyleGAN-Ly(128^2) inversion, where the decoder is fine-tuned accordingly for 25k iterations. It shows that with the increase of training iterations of StegaStyleGAN-Ly inversion, the generated stego images gradually resemble the real target images along with promising Acc performance.

Figure 7: 128^2 stego images with 6.25×10^{-2} bpp capacity generated by StegaStyleGAN-Ly trained with CelebA.

(a) 1bpp

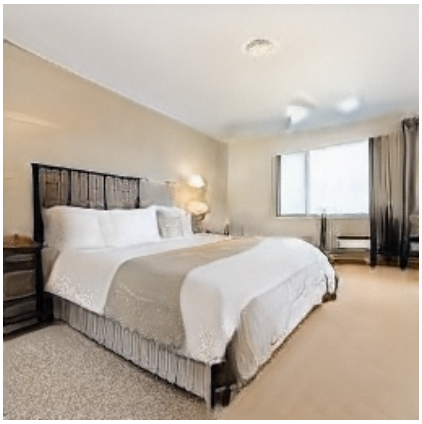
(b) 2bpp

(c) 4bpp

Figure 8: 128^2 stego images with different relative embedding capacities generated by StegaStyleGAN-Ls trained with CelebA.

(a) 1bpp

(b) 2bpp



(c) 4bpp

Figure 9: 256^2 stego images with different relative embedding capacities generated by StegaStyleGAN-Ls trained with Lusn-bedroom.

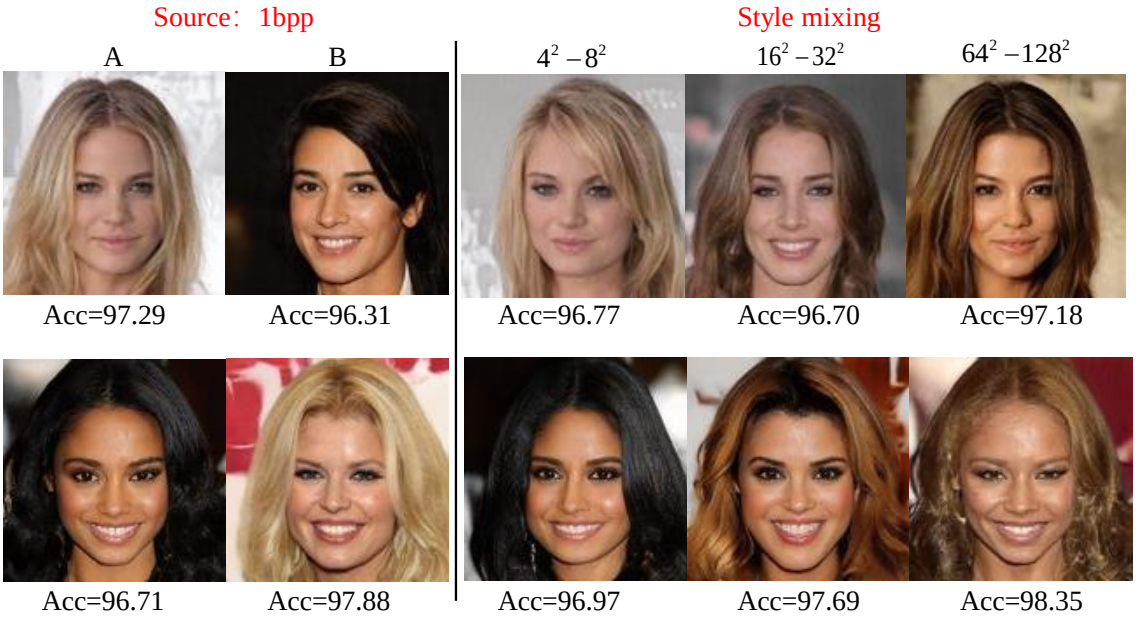


Figure 10: Similar to the operation in StyelGAN (Karras, Laine, and Aila 2019), we copy the styles of different resolutions from Source B to replace the corresponding ones in Source A to simulate *style mixing*.

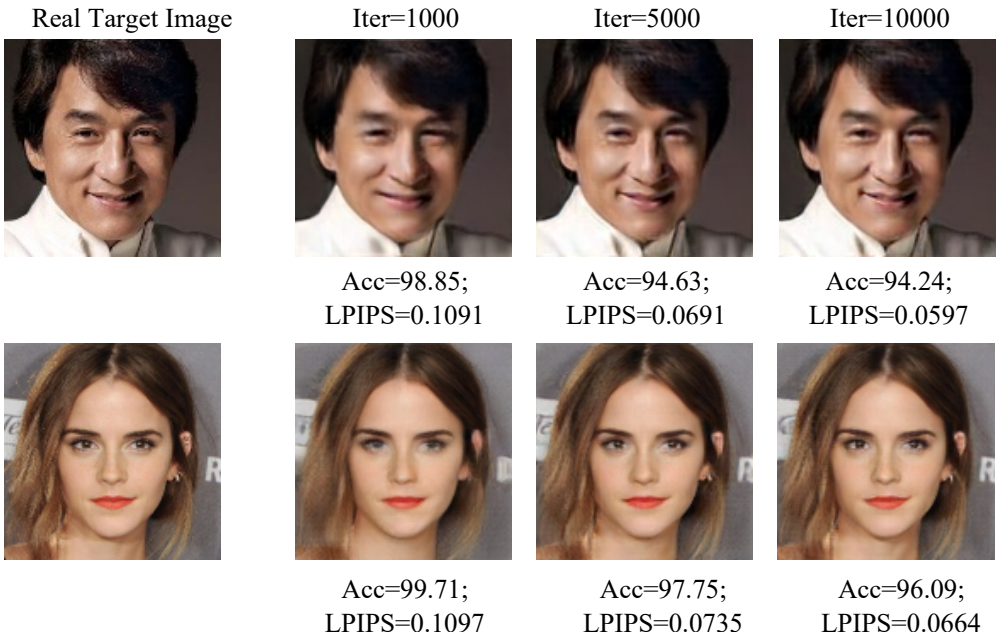


Figure 11: The inversion of StegaStyleGAN-Ly(128^2) with 6.25×10^{-2} bpp capacity against QF75 JPEG compression attack.