# Look Ahead! Practical CCA-secure Steganography:

## Cover-Source Switching meets Lattice Gaussian Sampling

Russell W. F. Lai[1] Ivy K. Y. Woo[1] Hoover H. F. Yin[2]

[1]Aalto University, Finland [2]The Chinese University of Hong Kong

*Published at EUROCRYPT 2026*

One of these two photos secretly carries an encrypted file in its pixel values. The other is just a photo.
**Can you tell which?**

**Photo A**

**Photo B**

### The reveal

You cannot tell, and neither can any attackers. Photo B was shot at a higher ISO, so the camera sensor captured more noise. **Photo A** was shot at *lower ISO* but where an encryption was embedded in a way that *mimics additional noise*. The result is provably indistinguishable from the higher-ISO Photo B, even against an attacker who can intercept and tamper (CCA security). On a 46 MB photo we can encode a message up to ∼11.5 MB.

## 1 Hiding the act of hiding

Traditional encryption scrambles a message into gibberish, but gibberish itself looks suspicious. **Steganography** hides the very fact that one is communicating: The output *stegotext* looks like an ordinary *cover source*, such as a photo. The receiver recovers the message with a secret key; others see only a cat photo.

## 2 Our approach: Camera sensor noise

Take the same scene at two ISO settings. The only difference is *sensor noise* – the faint random grain from nature, which grows with the ISO setting. Instead of trying to minimise changes due to embedding (the traditional steganography), our stegotext *mimics another photo* shot at different ISO.

### Cover-source switching for photos [Bas16]

Hiding a message via *difference between two distributions*: Embed the message to a low-ISO source $\mathcal{C}$, so that the resulting stegotext is provably indistinguishable from a high-ISO source $\widetilde{\mathcal{C}}$.

Per-pixel noise is modelled as a **Gaussian**. Embedding takes three steps.

1. Take a real low-ISO photo (a sample of $\mathcal{C}$).
2. Encrypt the message. The ciphertext $\mathbf{c}$ looks like random bits.
3. Use a **lattice Gaussian sampler** [MP12] to sample Gaussian noise that (1) looks like genuine sensor noise and (2) secretly encodes the ciphertext.

### Lattice Gaussian-preimage sampling [MP12]

For the gadget matrix $\mathbf{G} = \mathbf{I} \otimes (2^0, 2^1, \ldots, 2^{\lceil \log q \rceil})$, any target $\mathbf{c}$, any centre $\mathbf{p}^*$, and any appropriate width $\sigma$, there is an efficient sampler

$$\tilde{\mathbf{p}} \leftarrow \mathsf{SampPre}(\mathbf{c}, \sigma, \mathbf{p}^*)$$

s.t. the sample $\tilde{\mathbf{p}}$ is statistically close to *discrete Gaussian* of width $\sigma$ centred at $\mathbf{p}^*$, *subject to* $\mathbf{G} \cdot \tilde{\mathbf{p}} = \mathbf{c} \bmod q$, i.e. left-multiplication by $\mathbf{G}$ maps to $\mathbf{c}$.

Centre $\mathbf{p}^*$ models the "true pixel value". A photo's actual pixel values follow Gaussian centred at $\mathbf{p}^*$, whose Gaussian width determined by the ISO setting. Outcome: A stegotext $\tilde{\mathbf{p}}$ that is indistinguishable from a real high-ISO shot (a sample of $\widetilde{\mathcal{C}}$), and can be decoded back to the message in two steps.

1. Recover ciphertext by computing $\mathbf{G} \cdot \tilde{\mathbf{p}} = \mathbf{c} \bmod q$.
2. Decrypt the ciphertext $\mathbf{c}$ using secret key.

## 3 CCA: Secure even against tampering

Real attackers may actively intercept and modify the targeted stegotexts, and obtain decoded messages of modified stegotexts. The prior scheme is insecure here: Sampling Gaussian-preimages is easy – an attacker samples another stegotext $\tilde{\mathbf{p}}'$ encoding the same ciphertext $\mathbf{c}$, then decodes $\tilde{\mathbf{p}}'$ to the message.

### Our solution, and the main result

**Fujisaki–Okamoto transform [FO13], adapted**: Generate the stegotext $\tilde{\mathbf{p}}$ with hash values as randomness, so that the receiver can recompute the stegotext using the same hash values, and reject anything tampered.
Result: A **CCA-secure** steganography scheme that is **concretely efficient**, achieves **high-embedding rate**, and uses only basic cryptographic primitives (PKE, PRF, hash functions) plus **Gaussian-preimage sampling**.

A proof-of-concept on real **24-megapixel RAW photos** (cover $\mathcal{C}$ = ISO-100, each file 46 MB). The higher the target ISO, the more one can embed:

| $\widetilde{\mathcal{C}}$ | \|msg\| | Encoding time | Rate |
|---|---|---|---|
| ISO-200 | ∼4 MB | ∼50 s | ∼9% |
| ISO-400 | ∼9 MB | ∼35 s | ∼19% |
| **ISO-800** | **∼11.5 MB** | ∼65 s | ∼**25**% |

## 4 Find out more

Paper (full scheme + proofs): ia.cr/2026/549

Implementation: github.com/gaussianstego/ns-gembed

### Open directions

- Other photo formats not RAW, e.g. JPEG?
- Other cover sources not photos, e.g. audio or video?
- More user-friendly pipeline? How to obtain noise profile more easily? (Needed for sampling appropriately distributed Gaussian noise.)

Selected references:
[BC05] Michael Backes and Christian Cachin. "Public-Key Steganography with Active Attacks". In: *TCC 2005*. 2005
[Bas16] Patrick Bas. "Steganography via Cover-Source Switching". In: *IEEE WIFS 2016*. 2016
[BL18] Sebastian Berndt and Maciej Liskiewicz. "On the Gold Standard for Security of Universal Steganography". In: *EUROCRYPT 2018*. 2018
[FO13] Eiichiro Fujisaki and Tatsuaki Okamoto. "Secure Integration of Asymmetric and Symmetric Encryption Schemes". In: *Journal of Cryptology* (2013)
[MP12] Daniele Micciancio and Chris Peikert. "Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller". In: *EUROCRYPT 2012*. 2012

Aalto Cryptography Group
research.cs.aalto.fi/crypto

Aalto University
School of Science