Steganography, Perfect Security, and Computational Security

1 What is steganography?

Steganography is the problem of hiding a secret message inside an ordinary-looking object, called the *covertext*, so that an observer cannot tell whether hidden communication is taking place.

In the LLM setting:

- the innocent channel is the normal language model,
- the covertext is the model's response,
- the hidden message is a payload embedded in that response,
- the adversary sees the response and tries to decide whether it is ordinary model output or steganographic output.

Formally, for a prompt p, let

$$P = \text{Model}(p)$$

be the innocent distribution over responses, and let

$$Q = \text{Steg}_k(p, m)$$

be the steganographic distribution when hiding message m using key k.

The adversary performs hypothesis testing:

$$H_0 : x \sim P$$

versus

$$H_1 : x \sim Q.$$

The goal is that the adversary cannot reliably distinguish these two cases.

2 Perfect information-theoretic steganography

The strongest possible notion is **perfect information-theoretic indistinguishability**.

This means that the stego output distribution is exactly the same as the innocent output distribution:

$$P = Q.$$

Equivalently,

$$d_{\text{TV}}(P, Q) = 0,$$

where total variation distance is

$$d_{\mathrm{TV}}(P, Q) = \frac{1}{2} \sum_x |P(x) - Q(x)|.$$

Total variation distance is the operational distance for hypothesis testing. If the adversary gets one sample and the two hypotheses have equal prior probability, then the optimal success probability is

$$\Pr[\text{correct}] = \frac{1}{2} \left(1 + d_{\mathrm{TV}}(P, Q)\right).$$

So if $d_{\mathrm{TV}}(P, Q) = 0$, no adversary, even an unbounded one, can distinguish stego output from innocent output.

3 Why perfect steganography needs large secret randomness

Suppose:

- M is the hidden message,
- K is the shared secret key or randomness,
- X is the transmitted stegotext.

Assume perfect hiddenness:

$$I(M; X) = 0.$$

That means the observed stegotext X reveals no information about the hidden message.
Also assume perfect decoding:

$$H(M \mid X, K) = 0.$$

That means the receiver can recover M exactly from the stegotext and the secret key.
Then:

$$H(M) = I(M; X, K).$$

By the chain rule,

$$I(M; X, K) = I(M; X) + I(M; K \mid X).$$

Since $I(M; X) = 0$,

$$H(M) = I(M; K \mid X).$$

And since mutual information is bounded by entropy,

$$I(M; K \mid X) \leq H(K \mid X) \leq H(K).$$

Therefore,

$$\boxed{H(M) \leq H(K).}$$

So, for perfect information-theoretic steganography with perfect recovery, the shared secret must contain at least as much entropy as the hidden message.

For arbitrarily many hidden messages, a finite fixed key is not enough. One needs an unbounded supply of shared secret randomness, or some mechanism for refreshing it.

This is analogous to the one-time pad: perfect secrecy consumes secret key material.

4 Why this is often impractical

If sender and receiver already have a secret channel capable of exchanging large amounts of fresh randomness, then they could often use that channel directly for communication or key refreshment.

That does not make steganography useless, because steganography hides *the existence of communication*, whereas encryption only hides *the content*. But it does mean that perfect information-theoretic steganography is usually expensive in secret randomness.

Therefore, practical schemes often use a short reusable secret key and settle for **computational security**.

5 Computational steganography

In computational steganography, the requirement is weakened.

Instead of demanding

$$P = Q$$

against all adversaries, we demand that no efficient adversary can distinguish P from Q.
So the guarantee is not:

> no one can distinguish the two distributions.

It is:

> no polynomial-time adversary can distinguish them with non-negligible advantage.

An unbounded adversary may still break the scheme.

6 Pseudorandom functions

A **pseudorandom function** (PRF) is a family of keyed functions

$$F = \{F_k : \{0,1\}^{\ell_1(\lambda)} \to \{0,1\}^{\ell_2(\lambda)} \mid k \in \{0,1\}^\lambda\}.$$

It is a PRF if:

1. F_k is efficiently computable given k.
2. For every probabilistic polynomial-time distinguisher D,

$$\left| \Pr_{k \leftarrow \{0,1\}^\lambda} \left[D^{F_k(\cdot)}(1^\lambda) = 1 \right] - \Pr_{f} \left[D^{f(\cdot)}(1^\lambda) = 1 \right] \right| \leq \mathrm{negl}(\lambda),$$

where f is a truly random function with the same domain and range.
Intuitively:

- a truly random function requires an enormous amount of randomness,
- a PRF uses only a short key,
- but to efficient algorithms, it looks like a truly random function.

7 How the paper uses this idea

The LLM normally samples from a next-token distribution. For example, if the model predicts

$$\Pr(\text{cat}) = 0.5, \quad \Pr(\text{dog}) = 0.3, \quad \Pr(\text{fox}) = 0.2,$$

then ordinary sampling draws a uniform random number $u \in [0, 1]$ and chooses:

- cat if $u < 0.5$,
- dog if $0.5 \leq u < 0.8$,
- fox if $0.8 \leq u < 1$.

The steganographic scheme does the same thing, except u is generated from a secret-keyed PRF.

Because the PRF output behaves like a uniform random number to efficient observers, the marginal token distribution is unchanged. The model still outputs cat, dog, and fox with probabilities $0.5, 0.3, 0.2$.

The hidden message is encoded by making the PRF input depend on the hidden symbol currently being transmitted. The decoder, knowing the key, tests which candidate hidden symbol best explains the observed sequence of sampled tokens.

8 Attack cost

If the secret key has length λ, the generic brute-force attack is to try all possible keys:

$$2^\lambda.$$

For each candidate key, the attacker checks whether the observed text has the statistical structure expected under that key. If the correct key is found, the attacker can run the retrieval algorithm and decode the payload.

So the brute-force attack cost is roughly

$$O(2^\lambda).$$

More carefully:

- the attack cost is *at most* 2^λ by exhaustive search;
- a weak PRF or flawed construction could allow faster attacks;
- a well-designed scheme aims to make exhaustive search the best available strategy.

9 Relation to assumptions

The relevant assumption is not simply

$$P \neq NP.$$

That is too weak for modern cryptography.

A more standard assumption is the existence of **one-way functions**, which implies the existence of PRFs.

So the computational-security story is:

1. Assume secure PRFs exist.
2. Replace true randomness by PRF-generated randomness.
3. Efficient adversaries cannot distinguish the PRF from true randomness.
4. Therefore efficient adversaries cannot distinguish stego output from innocent output, except with negligible advantage.

But an unbounded adversary can still brute-force the key or distinguish the PRF family from a truly random function.

10 Final summary

Perfect information-theoretic steganography requires

$$P_{\text{stego}} = P_{\text{innocent}}$$

exactly, and with perfect decoding it requires secret randomness satisfying

$$H(K) \geq H(M).$$

So unlimited perfect hidden communication requires an unlimited or refreshed secret resource.

Computational steganography instead uses a short reusable key and a PRF. The output distribution is computationally indistinguishable from innocent model output, assuming the PRF is secure. This gives practical steganography, but only against efficient adversaries.

The tradeoff is:

$$\text{perfect security} \Rightarrow \text{large fresh secret randomness}$$

whereas

$$\text{short reusable key} \Rightarrow \text{computational security only.}$$