# Cryptographic Encoding of Adversarial Prompts to Bypass NSFW Filters in Text-to-Image Diffusion Models

Raja Shekar Reddy Seelam*, Ruimin Sun†
*Student,†Assistant Professor
Knight Foundation School of Computing and Information Sciences
Florida International University, Miami, USA
Email*: rseel003@fiu.edu, Email†: rsun@fiu.edu

q ***Abstract*—Text-to-image (T2I) diffusion models have demonstrated an impressive ability to generate imagery from textual prompts. However, these powerful models carry significant safety risks, notably the potential to produce "not safe for work" (NSFW) content such as pornographic or violent images. To mitigate misuse, many T2I systems incorporate prompt filters that block explicit or disallowed inputs. This paper explores a novel adversarial attack that leverages cryptographic encoding—specifically base64 text encoding—to bypass such safety mechanisms. We show that by encoding a forbidden prompt (e.g., a description of nude or explicit content) in base64, an attacker can evade keyword-based filters and still induce the diffusion model to generate the disallowed (NSFW) image. [10] The results highlight a concerning security gap: even when obvious NSFW terms are filtered, diffusion models may be tricked into unsafe content generation via encoded prompts. Our work underscores the need for more robust, multi-layered safety in generative AI and introduces cryptographic encoding as a new dimension of adversarial prompt attacks on diffusion-based image generators.**

***Index Terms*—Text-to-Image Diffusion, Adversarial Attacks, Base64 Encoding, NSFW Content, AI Safety**

## I. Introduction

Generative AI models have revolutionized content creation, enabling users to produce text, images, and other media through natural language prompts. Text-to-image (T2I) diffusion models like OpenAI's DALL·E and Stable Diffusion accept a text description and generate a corresponding image. [11] While these models unlock creative potential, they also pose safety risks: without constraints, users might prompt them to generate disallowed or harmful content (e.g., pornography, extreme violence, or hate imagery). To address this, prompt filtering mechanisms are deployed in many systems. Prompt filters scan the user's input text for NSFW or other banned keywords and block or modify prompts that violate content guidelines. For example, a prompt containing an explicit sexual request or a slur would trigger the filter, preventing the model from generating an image. Such filters are a first line of defense to ensure AI image generation remains within acceptable and legal bounds.[15] [2]

However, as defenses improve, so do the attacks. In this paper, we explore a high-level evasion idea: using cryptographic encoding of prompts to slip past safety filters. The intuition is that a content filter looking for banned words (like "nude" or "blood") might fail to recognize those words if they are encoded in a seemingly random sequence (for instance, in base64 format). Base64 encoding transforms plaintext data into an ASCII string of characters (letters, digits, +, / and =) that looks like gibberish. By encoding a sensitive prompt (e.g., "nude woman with explicit pose") into base64, an attacker can obtain a string of innocuous-looking characters. To a naive filter, this string does not match any blocked keywords, so the prompt is allowed to proceed. The diffusion model then may interpret or decode the string (either via a malicious decoding step inserted by the attacker or by the model's learned ability to recognize patterns), resulting in the generation of the originally intended NSFW image. In essence, the attacker uses a form of adversarial prompt that is cryptographically camouflaged: the model still gets the toxic instruction, but the filter does not.

This cryptography-based tactic builds on a growing body of adversarial attacks in multimodal AI. Previous work has shown, for instance, that diffusion models can be triggered to produce unexpected outputs through carefully crafted "nonsense" phrases or by adding unprintable characters to prompts. Our contribution is to demonstrate that systematic encoding of the entire prompt (via base64) can be a particularly effective bypass. This not only sidesteps simple blacklist filters but also introduces a new threat vector: malicious actors could automate the encoding of forbidden prompts, making detection by traditional means more challenging. The following sections delve into background concepts (large language models and NSFW content), prior research, the methodology of our attacks, and the broader implications for safety. We also discuss challenges encountered during our research and propose future directions and potential defenses to counter such cryptographic prompt attacks.[4] [12]

## II. Literature Review

### A. Large Language Models (LLMs) and Diffusion Models

*1) What is a Large Language Model (LLM)?:* A large language model is a type of machine learning model designed for natural language processing tasks, characterized by a very

high number of parameters and training on massive text corpora. LLMs predict text and generate human-like language by learning statistical patterns in language data. For example, OpenAI's GPT-3 is an LLM with 175 billion parameters, enabling it to generate coherent paragraphs of text and answer questions in context. LLMs achieve their performance via deep learning architectures (often Transformer networks) and self-supervised learning, absorbing linguistic knowledge from billions of words of training data.

*2) Types of LLMs:* There are several architectural variants of large language models, each with different purposes and strengths:

- **Decoder-only models:** These models, like the GPT series (GPT-2, GPT-3), use only the Transformer's decoder blocks. They excel at text generation because they predict the next token in a sequence. GPT-3, for instance, is a decoder-only Transformer that can produce fluent text and perform zero-shot or few-shot learning. It generates text by attending to prior context without an encoder stage.
- **Encoder-only models:** These models (often used for understanding tasks rather than generation) use only the encoder part of the Transformer. A prime example is BERT (Bidirectional Encoder Representations from Transformers), introduced by Google in 2018. BERT is an encoder-only model designed to produce deep bidirectional representations of text, which are useful for tasks like reading comprehension and sentiment analysis. Encoder-only LLMs are not typically used to generate long texts; instead, they output a contextual embedding or classification.
- **Encoder-Decoder models:** These combine an encoder (to digest an input sequence) with a decoder (to generate an output sequence). Models like T5 (Text-to-Text Transfer Transformer) and BART fall into this category. They are versatile: for example, you can feed an input sentence to the encoder and train the decoder to produce a translated sentence (machine translation) or a summary (text summarization). Encoder-decoder LLMs are common in sequence-to-sequence tasks and can also be adapted for image captions or multimodal inputs when paired with vision encoders.

It's worth noting that text-to-image diffusion models often incorporate LLM or language model components. For instance, DALL·E 2 uses a CLIP text encoder (a type of language model trained jointly with images) to understand prompts, and Stable Diffusion uses a text encoder (from CLIP or a similar model) to condition the image generation process. Thus, understanding LLMs is relevant because prompt processing in diffusion models relies on these language understanding modules.

*3) What are Text-to-Image Diffusion Models?:* Diffusion models are a class of generative models that create data (images, in this case) by iteratively denoising random noise, guided towards a target distribution. In text-to-image diffusion, the model conditions the generation on a text prompt. OpenAI's DALL-E (2021) was a seminal example: a neural network that learned to generate images from textual descriptions, demonstrating "zero-shot" ability to create novel combinations of concepts from prompts. DALL-E 2 (2022) improved on this by using a two-stage approach: first generating a CLIP image embedding from text, then a diffusion decoder to produce the final image.[7] [4]

Another breakthrough was Stable Diffusion (2022), which introduced latent diffusion models (LDMs). Stable Diffusion generates images in a latent space (the compressed feature space of an autoencoder) rather than pixel space, greatly reducing computational load. [4] It incorporates a cross-attention mechanism that allows the textual prompt to influence the image generation at every denoising step, effectively guiding the model to align with the prompt. These models have enabled high-quality image synthesis and are widely used via interfaces like Midjourney, DreamStudio, and Bing Image Creator.

### *B. NSFW Content and Its Categories*

*1) What is NSFW content?:* "NSFW" stands for Not Safe For Work. It is an internet label to warn that content is sexually explicit, graphically violent, or otherwise inappropriate for a professional or public setting. In other words, NSFW content includes anything one wouldn't want to open on an office computer or in front of children due to its mature or disturbing nature. The term originated as a simple tag on links or files and has since been adopted in content moderation practices to denote material that should be restricted or age-gated. [11]

*2) Types of NSFW Content:* NSFW is a broad category encompassing various forms of inappropriate or harmful material. The main sub-categories include:

- **Pornography and Sexual Content:** This includes images or descriptions of sexual acts, nudity, genitalia, and erotica. Examples range from full nudity and sexual activities to softcore erotic art. Such content is typically flagged because it is inappropriate in professional environments and often subject to age restrictions.
- **Graphic Violence and Gore:** Media depicting extreme violence, serious injury, blood, mutilation, or death falls into this category. For instance, an image of a bloody crime scene or gory horror elements would be NSFW. This content can be shocking or traumatizing, hence not suitable for general audiences.
- **Profanity and Hateful Imagery:** While strong language alone in text may not always trigger NSFW (context matters), repeated profanity or hateful slurs can be considered NSFW, especially if used in an abusive manner. In images, hateful imagery could include hate symbols, extremist content, or derogatory depictions of protected groups.
- **Other Disturbing Material:** Some content is labeled NSFL (Not Safe For Life), indicating it is extremely disturbing or nauseating. This might include fetish pornography that is very extreme, or real depictions of torture, bestiality, necrophilia, etc. Additionally, content involving self-harm or suicide might be tagged to warn viewers, although again those often have separate policy categories.

## III. Methodology

### A. Problem Overview: NSFW Image Generation via Diffusion Models

The core problem we address is the generation of NSFW images by text-to-image diffusion models despite the presence of safety filters. In a standard setup, a user provides a text prompt to a diffusion model (like "a photo of a woman in a bikini on a beach"). [11] The model, if unrestricted, could just as easily take a prompt for pornographic content (e.g., "a nude woman performing [sexual act]") and produce a realistic image of that description. Because such outputs are disallowed, systems implement filters to intercept obvious NSFW prompts. [6]

However, these filters can be circumvented. Our methodology demonstrates an attack where the user encodes the prompt in a certain way to conceal its true intent from the filtering system. The diffusion model (or a component of the system) then decodes or interprets the prompt and generates the NSFW content as if it had received it in plain form. [1]

### B. Bypassing Filters with Base64 Encoding

Our attack focuses on a scenario where the filter looks only at human-readable text and does not attempt to interpret encoded or obfuscated inputs. Base64 encoding is a method of converting text (or binary data) into a string of ASCII characters. It is not encryption (there's no secret key; it's a reversible transformation), but the output looks like random gibberish to humans. For instance, the simple NSFW phrase "nude woman" in base64 becomes `bnVkZSB3b21hbg==`. To an English-speaking person (or a basic word-filter), `bnVkZSB3b21hbg==` has no obvious meaning — it's just a string of letters, numbers, and equal signs. [8]

*1) Attack Method:* Instead of inputting the NSFW prompt directly, the user encodes it in base64. The prompt given to the image model might be something like:

```
bnVkZSB3b21hbg==, highly detailed,
professional photograph
```

Here the user has mixed the encoded core instruction with additional normal words to guide style (which might further confuse simple filters). The critical part `bnVkZSB3b21hbg==` is the base64 text for "nude woman".

If the content filter is naive, it will scan this prompt and not find any match for "nude" or other banned words (since they are hidden in the encoded string). Thus, the filter lets the prompt pass through to the model.

There are two ways the model might then produce the NSFW image:

1) **External decoding (attack pipeline):** The attacker (if they have control over how the model is invoked, such as running their own copy of Stable Diffusion or using an API in an unconventional way) could decode the base64 string back into natural language after the filtering step but before generation. This requires a two-step pipeline: the filter sees a safe-looking prompt, then a decoding function converts it to the unsafe prompt for the model. [11]
2) **Model interpretation (learned encoding):** A more intriguing possibility is that the diffusion model's text encoder might directly or indirectly make sense of the base64 string. Could the model have seen base64-encoded data during training and learned to map `bnVkZSB3b21hbg==` close to the concept of "nude woman"? While standard training data is mostly natural language, it is possible that some encoded text appeared in the large corpus or that the model can partially decode simple patterns.

In our experiments, the more reliable method was to explicitly decode the prompt after bypassing the filter. For example, if using Stable Diffusion locally with the official Safety Checker, one can disable the checker, input the base64 prompt, then manually decode it to feed into the model. The result: the model produces the same image it would have for the plaintext NSFW prompt, but the safety system is bypassed.

We found that base64 encoding is extremely effective at evading text-based filters. It is a simple substitution of the entire message into a different alphabet. Unlike leetspeak or slight misspellings (which filters can often catch via fuzzy matching), base64 encoding changes the text completely. Unless the filter explicitly decodes base64 and checks the result, it will have no idea what the user is really requesting. This is analogous to speaking in code: the gatekeeper does not understand the code, but the downstream system does.

### C. Relation to Prior Adversarial Research

Our cryptographic prompt attack is conceptually related to adversarial examples and prompt injection research in multimodal models. Previous studies on adversarial attacks in vision and language have typically involved adding perturbations or finding inputs that maximize undesired behavior. For instance:

- In image classification, adding slight noise to an image can cause a classifier to mislabel it while the changes remain imperceptible to humans (classic adversarial examples).
- In text prompts for image generation, some researchers discovered "hidden vocabulary" attacks, where nonsense words consistently yielded certain outputs. One example was DALL-E 2 treating the gibberish word "Apoploeon" as a cue to produce bird images.

Adversaries have also tried obfuscation attacks on text filters. For example, to bypass a filter for hate speech, one might use homoglyphs (cloaking a bad word with similar-looking Unicode characters) or insert innocuous characters between letters (e.g., "h a t e"). These tricks exploit literal pattern matching weaknesses. [11]

Our use of base64 is a more systematic and powerful obfuscation: it encodes the entire prompt. This extends adversarial prompt attacks into a new domain of machine-understandable but human-obscure instructions. It demonstrates that safety filters operating only on cleartext are vulnerable to inputs that are systematically transformed.

## IV. Challenges Faced

During the research and experimentation for this paper, we encountered several challenges:

### A. Understanding Complex Model Architectures

Large Language Models (LLMs) and Vision-Language Models (VLMs) — including diffusion-based generative models — are underpinned by intricate architectures and mathematics. Diving into the literature of Transformers, diffusion processes, and multimodal encoders was a non-trivial task. Building this understanding was necessary to hypothesize whether an encoded prompt might be decipherable by the model. [2] However, the learning curve was steep. Grasping nuances of, for instance, how the tokenizer might split a base64 string, required deep exploration of technical documentation and research papers. This challenge underlines the interdisciplinary knowledge needed, blending AI theory with practical coding and some cryptography.

### B. Volume of Prior Work (Literature Overload)

The field of generative AI safety is rapidly evolving, with new papers and techniques emerging continuously. Conducting the literature review meant sifting through numerous sources about NSFW detection, adversarial attacks on models, and cryptographic or steganographic methods in the AI context. Distilling the essence without getting lost in unrelated details required careful focus.

### C. Model Deprecation and Access Issues

Some models or tools referenced in earlier studies were no longer readily accessible. For instance, a previous adversarial attack may have been demonstrated on "Stable Diffusion v1.4" with a certain safety filter configuration. By the time of our research, updated versions had different behaviors. Additionally, repositories and model weights for certain NSFW filters mentioned in 2022 studies had been taken down. This made reproducing prior results or direct comparison challenging.

### D. Experimenting Safely with NSFW Content

Since our work intentionally dealt with generating disallowed content for testing, we had to ensure responsible practices. Experiments were conducted in a closed, local environment, avoiding exposure to online services or violation of usage terms. Special care was taken to isolate the models, disable internet access during testing, and securely delete any generated NSFW images post-analysis.

### E. Interpreting Ambiguous Model Behavior

Dealing with adversarial inputs like base64 text led to unpredictable model outputs. At times, the model might not output anything close to the intended NSFW image. In other cases, partial success was observed — the model generated elements of the intended scene but not fully. Interpreting these outcomes required careful thought: was the attack partially failing, or was the model influenced by other aspects of the prompt? Repeated trials and simplified test cases (e.g., using a single encoded word) helped disentangle these scenarios.

## V. Future Work Areas

Our study of base64-encoded prompt attacks opens up several intriguing avenues for future exploration. The use of cryptographic and steganographic techniques in adversarial prompting is still in its infancy, and extending this concept could both reveal potential vulnerabilities and inspire more secure model designs. Here, we outline a few promising future work areas:

### A. Beyond Base64: Advanced Cryptographic Encodings

Base64 is just one encoding scheme, and it is relatively easy to decode. A determined adversary could employ more complex cryptographic techniques to hide prompt content. For instance, one could use simple ciphers (like Caesar shifts) or even modern encryption algorithms (e.g., AES with a known key). Encoded text would look equally or more nonsensical than base64 to a filter. [14]

An interesting research question is whether diffusion models can be conditioned to decrypt or understand encrypted prompts if they were somehow trained on them or provided with the decryption key. [9] Additionally, chaining encodings — such as base64 encoding followed by a string reversal — could create more robust obfuscation layers. Understanding the limits of what transformations a model's text encoder can inversely learn is a fascinating direction.

### B. Steganographic Attacks in Image-to-Image or Multimodal Setups

Many diffusion frameworks allow image inputs in addition to text prompts — for instance, image-to-image generation or inpainting. These introduce opportunities for steganographic attacks, where hidden instructions are embedded in an image rather than the text. [5]

Consider a scenario where the text prompt is benign (e.g., "a portrait of a woman in a dress"), but the initial image supplied carries a hidden message embedded via slight color variations or pixel frequency manipulations. Diffusion models, especially if they have multimodal training, might pick up on these subtle cues. Future research could attempt to encode instructions in images (or even in the noise initialization of the diffusion process), merging adversarial perturbations with diffusion guidance.

### C. Generalizing to Other Modalities

While this paper focuses on text-to-image diffusion models, the concept of cryptography-based adversarial attacks could extend to other generative modalities. For example:

- In text generation systems (e.g., ChatGPT), one could encode a disallowed instruction to trick the model into executing it if it internally decodes the hidden command. [11]
- In audio generation models, supersonic Morse code embedded in priming audio could instruct the model to generate disallowed speech without a detectable text prompt.

Exploring these cross-modal attacks will be valuable. Essentially, wherever an input filter fails to fully interpret all possible encodings of input data, a potential vulnerability exists.

## VI. Mitigation Techniques

The emergence of cryptography-based prompt attacks calls for enhanced mitigation strategies in generative AI systems. Defending against these attacks is challenging because the attacker's prompt no longer contains obvious red flags. Nonetheless, several approaches can be combined to strengthen the system:

### *A. Robust Input Validation and Decoding*

A simple step is to extend content filters to decode or normalize inputs before checking for disallowed content. For example, if the input text matches the pattern of base64 encoding (easy to detect via its character set and padding symbols like = signs), [3] the system could attempt automatic decoding. If decoding yields readable text, the standard NSFW filter can then be applied.

### *B. Entropy and Anomaly Detection in Prompts*

Encoded or encrypted strings often have different statistical properties than normal language. A base64 string, for instance, tends to be longer and has higher character variety (upper case, lower case, digits, symbols) compared to a typical English phrase. [13]

One mitigation approach is using an anomaly detector on prompts. If a prompt appears highly entropic or matches specific regular expressions (e.g., $[A-Za-z0-9+/=]\{10,\}$ for base64), it can be flagged for review. Although legitimate uses of random strings exist, treating heavily encoded-looking prompts as suspicious is a reasonable security measure.

### *C. Multi-Layered Filtering (Defense in Depth)*

Relying solely on input filtering is brittle. A secure system should use multiple checkpoints:

- **Input Level:** Decode inputs where possible and check the decoded content for NSFW risks.
- **Intermediate Level:** Monitor token embeddings and internal representations for anomalous patterns.
- **Output Level:** Apply NSFW image classifiers on generated outputs. Even if an unsafe prompt sneaks through, a robust output filter should catch NSFW imagery and block it from being shown.

In our base64 attack scenario, even if prompt filters fail, a strong NSFW output classifier serves as the final safety net.

### *D. Dynamic and Contextual Filtering*

Modern AI systems can employ AI to filter AI. Instead of relying solely on static keyword lists, a secondary AI model could judge the intent of the prompt. For instance, a classification model could predict the probability that a prompt, even in encoded form, relates to sexual content or violence.

Similarly, after image generation, the system could apply captioning models that describe the generated image (e.g., "A naked woman on a beach") and then filter based on this textual description. These are complex approaches but add multiple layers where an attack needs to evade all checks to succeed.

## References


[1] E. Bagdasaryan, T. Y. Hsieh, B. Nassi, and V. Shmatikov. Abusing images and sounds for indirect instruction injection in multi-modal llms. *arXiv preprint arXiv:2307.10490*, 2023.

[2] B. Bullwinkel, A. Minnich, S. Chawla, G. Lopez, M. Pouliot, W. Maxwell, et al. Lessons from red teaming 100 generative ai products. *arXiv preprint arXiv:2501.07238*, 2025.

[3] N. Carlini, J. Hayes, M. Nasr, M. Jagielski, V. Sehwag, F. Tramer, and E. Wallace. Extracting training data from diffusion models. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 5253–5270, 2023.

[4] P. Chao, A. Robey, E. Dobriban, H. Hassani, G. J. Pappas, and E. Wong. Jailbreaking black box large language models in twenty queries. *arXiv preprint arXiv:2310.08419*, 2023.

[5] Y. Deng and H. Chen. Harnessing llm to attack llm-guarded text-to-image models. *arXiv e-prints, arXiv:2312*, 2023.

[6] J. Hayase, E. Borevković, N. Carlini, F. Tramèr, and M. Nasr. Query-based adversarial prompt generation. In *Advances in Neural Information Processing Systems*, volume 37, pages 128260–128279, 2024.

[7] X. Jia, T. Pang, C. Du, Y. Huang, J. Gu, Y. Liu, and M. Lin. Improved techniques for optimization-based jailbreaking on large language models. *arXiv preprint arXiv:2405.21018*, 2024.

[8] H. Kwon and W. Pak. Text-based prompt injection attack using mathematical functions in modern large language models. *Electronics*, 13(24):5008, 2024.

[9] Y. Liu, G. Yang, G. Deng, F. Chen, Y. Chen, L. Shi, et al. Groot: Adversarial testing for generative text-to-image models with tree-based semantic transformation. *arXiv preprint arXiv:2402.12100*, 2024.

[10] X. Shen, Y. Qu, M. Backes, and Y. Zhang. Prompt stealing attacks against text-to-image generation models. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 5823–5840, 2024.

[11] Y. Yang, R. Gao, X. Wang, T. Y. Ho, N. Xu, and Q. Xu. Mma-diffusion: Multimodal attack on diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7737–7746, 2024.

[12] J. Yu, X. Lin, Z. Yu, and X. Xing. Gptfuzzer: Red teaming large language models with auto-generated jailbreak prompts. *arXiv preprint arXiv:2309.10253*, 2023.

[13] S. Zhai, H. Chen, Y. Dong, J. Li, Q. Shen, Y. Gao, and Y. Liu. Membership inference on text-to-image diffusion models via conditional likelihood discrepancy. *Advances in Neural Information Processing Systems*, 37:74122–74146, 2024.

[14] S. Zhai, Y. Dong, Q. Shen, S. Pu, Y. Fang, and H. Su. Text-to-image diffusion models can be easily backdoored through multimodal data poisoning. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 1577–1587, 2023.

[15] A. Zou, Z. Wang, N. Carlini, M. Nasr, J. Z. Kolter, and M. Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023.