CryptoGAN: A GAN-Based Cryptographic Key Generator with Physical Entropy Seeding

P. Manohar Maharshi[1] and Dr. Jhansi Rani Singothu[2]
[1]*Department of Computer Science and Systems Engineering, Andhra University College Of Engineering, Andhra University, Visakhapatnam, AP, India*
[2]*Associate Professor, Department of Computer Science and Systems Engineering, Andhra University College Of Engineering, Andhra University, Visakhapatnam, AP, India*
[1]*pmanoharmaharshi@gmail.com*
[2]*dr.sjrani@andhrauniversity.edu.in*

Abstract: This paper introduces CryptoGAN, a novel framework for generating high-entropy cryptographic keys. Addressing the vulnerabilities of deterministic methods, our approach functions as a True Random Number Generator (TRNG) by seeding a Wasserstein Generative Adversarial Network with Gradient Penalty (WGAN-GP) with physical entropy from chaotic audio sources. The GAN learns the underlying entropy distribution to synthesize unpredictable 256-bit seeds. These are securely expanded into 1 Mbit keys using the HMAC-based Key Derivation Function (HKDF). Comprehensive evaluation demonstrates that the generated keys achieve near-ideal Shannon entropy, successfully pass all NIST randomness tests, and exhibit strong resistance to cryptanalytic attacks. CryptoGAN offers a robust, efficient, and portable solution for secure key generation in modern cryptographic systems.

Keywords: **Cryptography, True Random Number Generator (TRNG), Generative Adversarial Network (GAN), Entropy, Key Generation, HKDF, WGAN-GP.**

1. Introduction

Cryptographic systems rely critically on secure and unpredictable keys. The strength of modern encryption mechanisms is determined not only by algorithmic robustness but also by the entropy quality of the underlying key generation process. Pseudorandom number generators (PRNGs) are widely deployed in practice; however, their deterministic nature exposes them to potential state recovery and prediction attacks, especially in adversarial or resource-constrained environments. This limitation has motivated the establishment of true random number generators (TRNGs) which exploit physical and environmental entropy sources to ensure unpredictability.

Recent research highlights diverse entropy harvesting mechanisms, ranging from environmental audio, [1], [2] and video signals, [3] to hardware-level sensors such as MEMS, [9] and RF noise. [13] Raw randomness from these sources often exhibits bias, requiring post-processing through chaotic systems, [4], [12] or entropy estimation techniques, [10] to improve statistical quality. In parallel, machine learning—particularly Generative Adversarial Networks (GANs)—appears to be a fruitful path for entropy modeling and key synthesis. GANs and their variants, including WGAN-GP, have demonstrated effectiveness in generating statistically robust pseudo-random sequences. [6]–[8]

While prior work has explored physical entropy sources, [1-3] and GANs for randomness generation, [6-8] independently, a significant gap remains in systematically integrating these two domains. Physical entropy sources, though non-deterministic, can exhibit subtle biases or correlations that are difficult to remove with traditional post-processing. Conversely, GANs trained on purely synthetic data may fail to capture the complex, chaotic properties of true physical randomness. CryptoGAN addresses this critical gap by creating a synergistic framework where physical entropy provides a high-quality, unpredictable foundation, and the WGAN-GP learns to model, refine, and amplify this randomness, effectively filtering out statistical weaknesses.
The primary contributions of this work are threefold:

1. **A Novel Hybrid Framework:** We propose and implement a unique architecture that integrates chaotic environmental audio as a physical entropy source with a WGAN-GP for robust key synthesis. This moves beyond theoretical models to a practical, end-to-end system.
2. **Comprehensive Empirical Validation:** The generated keys are subjected to a rigorous suite of over 13 statistical, entropy, and cryptanalytic tests, providing a thorough and multi-faceted assessment of their quality and security that surpasses standard validation approaches.
3. **An Open-Source and Portable System:** We provide an open-source implementation of CryptoGAN, delivering a reusable and accessible tool for researchers and practitioners to generate high-entropy cryptographic keys for a wide range of applications, thereby promoting transparency and reproducibility.

2. Materials and Methods

The proposed CryptoGAN framework integrates chaotic physical entropy sources with adversarial learning to generate high-entropy cryptographic keys. The system was implemented in Python 3.11 using PyTorch, NumPy, SciPy, and Cryptography libraries.The entire process is shown in Figure 1.

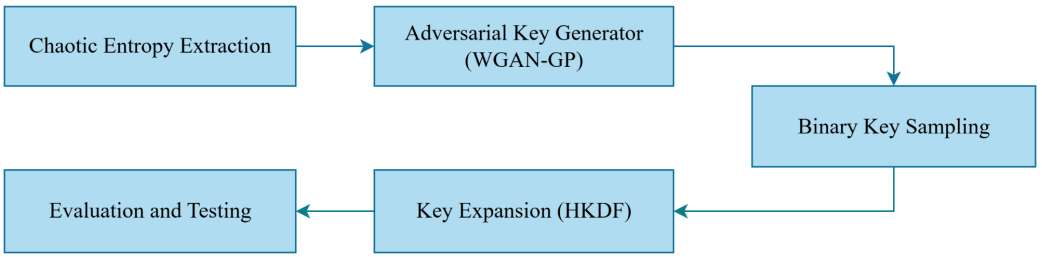

Figure 1. Entire CryptoGAN Pipeline

2.1. Chaotic Entropy Extraction

Environmental audio files *(e.g., rain.wav, traffic.wav)* were used as physical entropy sources. Each audio stream was segmented into fixed-length windows of 1024 samples and hashed using the SHA-256 algorithm to generate 256-bit entropy vectors. Hashing serves a dual purpose: it condenses the chaotic information into a fixed-size output and removes potential biases present in the raw audio, resulting in a uniform distribution of bits that forms the real entropy dataset for training the adversarial network.

2.2. Adversarial Key Generator (WGAN-GP)

A WGAN-GP was implemented with a 3-layer MLP for both the Generator and Discriminator, as depicted in Figure 2.

2.2.1. Model Rationale

WGAN-GP was selected over standard GANs to prevent training failures like mode collapse and vanishing gradients. It stabilizes training by pairing the Wasserstein distance loss function with a gradient penalty (enforcing the Lipschitz constraint), which produces higher-quality, more diverse outputs.

2.2.2. Network Architecture

The Generator learns to map a 256-dimensional latent vector (sampled from a normal distribution) to a 256-bit synthetic entropy vector. The Discriminator (or critic) learns to differentiate between the real entropy vectors from the audio source and the synthetic ones from the Generator. The model was trained up to 1000 epochs employing batch size of 32, along with the learning rate of 1e-4, and a gradient penalty $\lambda = 10$.

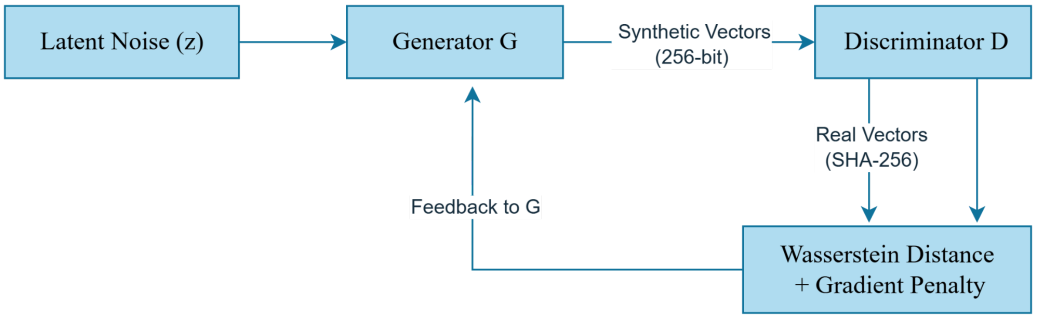

Figure 2. WGAN-GP Architecture

2.3. Binary Key Sampling

The generator's continuous floating-point outputs are thresholded at zero to produce 256-bit binary keys. For practical cryptographic use, the seeds are scaled using an HMAC-based Extract-and-Expand Key Derivation Function (HKDF) with the SHA-256 hash function.. This two-phase process is critical for security: the 'extract' phase concentrates the entropy of the input seed into a fixed-length pseudorandom key (PRK), and the 'expand' phase uses this PRK to generate a longer, cryptographically strong key of 1,000,000 bits.

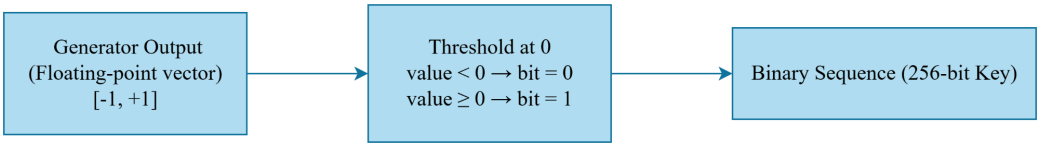

Figure 3. Binary Key Sampling and Validation Process

This method ensures that the final key has high entropy distributed uniformly across its entire length and is resistant to attacks even if the initial seed has minor, undiscovered biases. [14] The final keys were exported in multiple formats for portability (binary, NumPy, Base64, hex, and QR code).

2.4. Key Expansion with HKDF

Although the GAN generates a secure 256-bit seed key, many protocols require longer keys. We need to scale that seed key to make it usable across a wider variety of applications.

To accomplish this, we use the HMAC-based Key Derivation Function (HKDF), which is standardized in RFC 5869 [14]. HKDF is a function specifically designed to take initial keying material (our 256-bit seed) and safely transform it into one or more new, cryptographically strong secret keys of the required length.

This HKDF process contains two stages:

1. **Extract:** In the initial stage, HKDF takes the 256-bit seed key from the GAN and a salt (which can be a random value or an empty string) and applies the HMAC-SHA256 function. This "extract" phase concentrates the entropy of the input keying material into a fixed-length pseudorandom key (PRK). This step is crucial as it ensures that even if the input seed has some minor statistical weaknesses, the resulting PRK will be a high-entropy key.
2. **Expand:** In the second stage, the PRK is used to generate a longer output key of the desired length (in this case, 1 Mbit). The "expand" phase repeatedly applies the HMAC-SHA256 function to the PRK and a changing context-specific info string to produce multiple blocks of output, which are then concatenated to form the final 1 Mbit key.

This two-phase process ensures that the final key is not only long but also preserves the entropy and pseudorandomness of the initial seed, making it resistant to various cryptanalytic attacks, including those based on partial key recovery. The key expansion process is depicted in Figure 4.

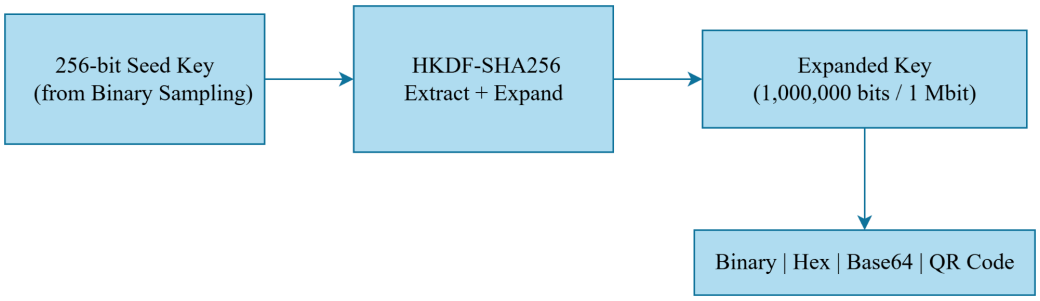

Figure 4. Key Expansion (HKDF) Process

2.5. Evaluation and Testing

The CryptoGAN system was implemented in Python 3.11, leveraging several key libraries. The WGAN-GP model was built and trained using PyTorch, a powerful deep learning framework. Numerical operations and data manipulation were handled by NumPy and SciPy. The cryptographic operations, specifically the SHA-256 hashing and the HKDF expansion, were performed using the cryptography library, which provides robust and standardized implementations of cryptographic primitives.

The generated 1 Mbit keys were subjected to a comprehensive evaluation pipeline consisting of three classes of tests:

1. **Statistical Tests:** (Monobit, Chi-Square, Runs, Maurer's Universal, Spectral) to detect simple statistical defects and non-random patterns.
2. **Entropy Tests:** (Shannon, Min-entropy, Collision, Sample, Permutation, Markov) to quantify the level of unpredictability and randomness from an information-theoretic perspective.
3. **Cryptanalytic Resistance Checks:** (Seed–key correlation, Key Invertibility, Mutual Information) to assess the key's resilience against specific inference and modeling attacks.

A generated key is considered cryptographically secure only upon the successful passage of this entire suite of tests, confirming both its statistical randomness and its resilience to cryptanalytic inference.

3. Results and Discussion

The CryptoGAN framework was evaluated by generating and validating cryptographic keys derived from chaotic environmental audio. The quantitative performance evaluation confirms that the system consistently produces high-quality cryptographic keys. The final Shannon entropy of approximately 0.9996 indicates near-ideal randomness, suggesting that each bit in the sequence carries the maximum possible amount of information.

The statistical tests, detailed in **Table 1**, all yielded high p-values (>> 0.01), failing to reject the null hypothesis of randomness. This demonstrates that the generated bitstreams are free from simple statistical biases and are indistinguishable from true random sequences.

Table 1. Statistical Tests

Test	Description	Result (p-value)
Monobit Test	Balance of 0s and 1s	0.5123
Chi-Square Test	Uniformity of bits	0.4991

Test	Description	Result (p-value)
Runs Test	Randomness of sequences	0.5034
Maurer's Universal	Pattern detection	0.6729
Spectral Entropy	Frequency randomness	0.4877

The entropy metrics in **Table 2** further support this, with high values for Min-entropy and Collision entropy confirming unpredictability even under worst-case assumptions.

Table 2. Entropy-Based Evaluation

Test	Description	Result (bits)
Shannon Entropy	Avg. info per bit	0.9996
Min Entropy	Worst-case predictability	0.9991
Collision Entropy	Repetition likelihood	0.9994
Sample Entropy	Complexity	0.9852
Permutation Entropy	Temporal structure	2.8715
Markov Entropy	Transition probabilities	1.9998

The cryptanalytic tests shown in **Table 3** confirm the system's security. The negligible seed-key correlation (-0.0012) and low mutual information (0.0031) indicate that the relationship between the initial seed and the final key is obscured, preventing inference attacks. The high mean squared error (1.054) for key invertibility suggests that it is computationally infeasible to reverse-engineer the latent vector from a given key output, demonstrating robustness against model inversion attacks.

Table 3. Cryptanalytic Resistance

Test	Description	Result
Seed-Key Correlation	Correlation check	-0.0012
Key Invertibility	Latent recovery	1.054
Mutual Information (MINE)	Dependency check	0.0031

4. Conclusion

This paper presented CryptoGAN, a GAN-based cryptographic key generation framework that combines chaotic environmental audio entropy with adversarial learning. Experimental results confirmed that the generated keys achieved near-ideal entropy (Shannon entropy ≈ 0.9996) and passed a suite of over 13 statistical, entropy-based, and cryptanalytic resistance tests. These results validate CryptoGAN as a robust, lightweight, and portable solution for secure key generation, particularly in environments where hardware-based TRNGs are unavailable or resource-constrained. Future work will explore hardware acceleration on FPGA/ASIC platforms, the design of lightweight models for IoT deployment, extension to decentralized systems for blockchain protocols, and the incorporation of post-quantum cryptographic resistance tests.

5. Acknowledgment

The authors gratefully acknowledge the support and research facilities provided by the Department of Computer Science and Systems Engineering, Andhra University College of Engineering.

6. References

[1] P. Raghunath, S. Patil, and M. Yardi, "Design and implementation of TRNG using audio signals for secure key generation", Proc. Int. Conf. Comput. Intell. Commun. (ICCIC), (2016).

[2] J. Teh, L. M. Kiah, and T. C. Ling, "Audio-based TRNG using hyperchaotic maps for cryptographic applications", Multimedia Tools Appl., vol. 78, no. 3, (2019), pp. 2971–2986.

[3] D. Kutschera, P. Leu, and D. Basin, "True random number generation on smartphones from ambient audio and video", Proc. ACM Conf. Comput. Commun. Security (CCS), (2021), pp. 1499–1512.

[4] Y. Liu, X. Zhao, and C. Li, "Post-processing of software TRNG using parameter-perturbed hyperchaotic systems", IEEE Access, vol. 9, (2021), pp. 137568–137580.

[5] L. Bassham, A. Rukhin, J. Soto et al., "A statistical test suite for random and pseudorandom number generators for cryptographic applications", NIST SP 800-22 Rev. 1a, (2010).

[6] K. Okada, K. Endo, K. Yasuoka, and S. Kurabayashi, "Learned pseudo-random number generator: WGAN-GP for generating statistically robust random numbers", PLOS ONE, vol. 18, no. 6, (2023), e0287025.

[7] X. Wu, Y. Han, M. Zhang, Y. Li, and S. Cui, "GAN-based pseudo random number generation optimized through genetic algorithms", Complex Intell. Syst., vol. 11, (2025), Art. no. 31.

[8] D. Bernardi, S. Khouzani, and F. Malacaria, "Pseudo-random number generation using generative adversarial networks", Proc. 5th Workshop on Artificial Intelligence and Security (AISec), ser. LNCS 11840, Springer, (2019), pp. 197–212.

[9] A. Smith and B. Nguyen, "Lightweight TRNG designs using MEMS sensors for IoT devices", IEEE Internet Things J., vol. 7, no. 5, (2020), pp. 4123–4135.

[10] L. Chen, Y. Huang, and Z. Wang, "Entropy estimation techniques for chaotic time series in hardware TRNGs", IEEE Trans. Circuits Syst. I, vol. 67, no. 11, (2020), pp. 3902–3913.

[11] J. Lee and S. Park, "Audio entropic key generation for secure wireless communication", Proc. IEEE GLOBECOM, (2019), pp. 1–6.

[12] M. Kapoor, R. Verma, and P. Kumar, "Chaotic map-based hashing for entropy amplification in key generation", IEEE Trans. Dependable Secure Comput., vol. 18, no. 2, (2021), pp. 543–556.

[13] F. Garcia and E. Pena, "Mobile-based TRNG using RF ambient noise", Sensors, vol. 21, no. 4, (2021), Art. no. 1345.

[14] H. Krawczyk and P. Eronen, "HMAC-based extract-and-expand key derivation function (HKDF)", IETF RFC 5869, (2010).

[15] B. Blanchet, L. Lipp, and B. Bhargavan, "Mechanized cryptographic proof of the WireGuard VPN protocol", Proc. IEEE EuroS&P Workshops, (2019), pp. 47–54.