Project 1

Parallel Computing & Cybersecurity: Breaking Encryption with Parallelism



Submitted by
Sagar Patel

© 2026 Sagar Patel. All rights reserved. Do not copy or redistribute without permission.

1 Introduction

Encryption serves as the backbone of digital security, protecting sensitive information across global networks. However, the increasing power of parallel computing systems presents both opportunities and challenges for cryptographic security. This report examines the intersection of parallel computing and cryptography, specifically focusing on how parallelism enables more efficient cryptanalysis and potentially threatens current encryption standards.

The significance of this research lies in its implications for cybersecurity infrastructure worldwide. As computational power continues to scale through parallel architecture, previously secure encryption methods face mounting pressure. Understanding these developments is crucial for security professionals, cryptographers, and policymakers who must anticipate and respond to evolving threats.

2 Background and Related Work

2.1 Evolution of Parallel Computing in Cryptanalysis

Modern cryptanalysis relies on powerful computers, just as encryption does. Brute-force attacks, which try all possible inputs, are the main method for breaking encryption. While some progress has been made, breaking modern cryptography is still slow. For example, RSA was brute-force in 1991, but stronger versions were already in use. Increasing key lengths and hash outputs makes cryptanalysis much harder, providing simple but effective security improvements. Cyber threats are growing more advanced, with hackers and state-sponsored attackers finding new ways to break encryption. This can lead to data breaches, financial losses, and privacy risks. Older security methods are no longer enough, so cryptography has evolved to strengthen security and defend against modern attacks along the computation resources within a machine.

2.2 Modern Parallel Architectures

Modern cryptanalysis leverages several parallel computing architectures:

- **Graphics Processing Units (GPUs):** Originally designed for rendering graphics, GPUs now serve as general-purpose parallel processors with thousands of cores capable of performing simultaneous operations.
- **Distributed Computing Networks:** Networks of computers working cooperatively on cryptanalytic tasks, like the distributed.net project that broke RC5-64 [1].
- **Quantum Computers:** Emerging technology that uses quantum mechanical properties to perform operations in parallel, potentially devastating for certain encryption algorithms.

3 Applications and Key Technical Aspects

3.1 Parallel Brute-Force Attacks Using GPUs

GPU-accelerated password cracking represents one of the most common applications of parallel computing in cryptanalysis. Modern GPUs excel at the repetitive calculations required for password hashing and can test billions of password combinations per second. For example, a system equipped with multiple NVIDIA RTX 3090 GPUs can achieve hash rates of approximately 100 billion NTLM password hashes per second [2]. This represents a speed increase of several orders of magnitude compared to CPU-based attacks. Password cracking tools like Hashcat[12] and John the Ripper leverage[13] this parallel processing capability to efficiently implement:

- Dictionary attacks enhanced with rule-based transformations
- Brute-force attacks across defined character spaces
- Rainbow table looks up against precomputed hash values

© 2026 Sagar Patel. All rights reserved. Do not copy or redistribute without permission.

The efficiency gains are particularly pronounced for algorithms without memory-hardening features. For example, MD5 and SHA-1 can be attacked at rates thousands of times faster on GPUs than on CPUs.

3.2 Experimental Comparison: GPU vs. CPU Performance

3.2.1 System Configuration

Table 1: System Specifications

Component	Specification
CPU	Intel® Core™ i5-8300H CPU @ 2.30GHz
Memory	32.00 GB
Storage	Samsung SSD 970 Evo Plus 1TB
GPU 0	Intel® UHD Graphics 630
GPU 1	NVIDIA GeForce GTX 1050

Our experimental results demonstrate the stark contrast between CPU and GPU-based brute force attacks on AES-128 encryption. Two brute force attacks were implemented to find key 0xffffffff in two different key search space as:

- 0x00000000 to 0xffffffff (0 - 4294967295)

Table 2: Performance Comparison: CPU vs GPU Brute Force Attack (4294967295 Total Keys)

Metric	CPU (OpenMP)	GPU (CUDA)
Threads/Blocks used	8 Threads	160 Blocks, 256 Threads/Block
Total Time Taken	3928.14 seconds ($\approx 65.5 minutes$)	123.50 seconds ($\approx 2.05 minutes$)
Search Speed	$\approx 1,093,383$ keys/second	$\approx 34,778,248$ keys/second

- 0x000000000 to 0xfffffffff (0 - 68719476735)

Table 3: Performance Comparison: CPU vs GPU Brute Force Attack (68719476735 Total Keys)

Metric	CPU (OpenMP)	GPU (CUDA)
Threads/Blocks used	8 Threads	160 Blocks, 256 Threads/Block
Total Time Taken	52,551 sec ($\approx 41.60 Hours$) Est.	1,939.93 seconds ($\approx 32.32 minutes$)
Search Speed	$\approx 163,249$ keys/second	$\approx 35,423,704$ keys/second

The most striking observations are:

1. In the 32-bit scenario, the GPU was approximately 32x faster
2. In the 36-bit scenario, the GPU performance improvement was even more dramatic, approaching 217x faster
3. The massive parallelism of GPU (40,960 potential concurrent threads) compared to CPU (8 threads) explains this significant performance difference

3.3 Breaking RSA and ECC Encryption

3.3.1 Classical Parallel Attacks on RSA

RSA security depends on the difficulty of factoring large prime numbers. Several parallel approaches exist for attacking RSA:

© 2026 Sagar Patel. All rights reserved. Do not copy or redistribute without permission.

1. General Number Field Sieve (GNFS): The most efficient known algorithm for factoring large integers. Implementations like CADO-NFS effectively distribute the sieving phase across multiple processors [4].

2. Parallel Pollard's Rho Method: Effective for smaller RSA keys but becomes impractical for key sizes of 1024 bits or larger.

While these methods benefit from parallelization, they still face exponential growth in complexity as key sizes increase. Currently, 2048-bit RSA keys remain secure against classical parallel attacks, though the margin of security continues to shrink as parallel computing power increases.

3.3.2 Quantum Algorithms for Cryptanalysis

Quantum computing represents the most significant threat to current public-key cryptography. Shor's algorithm, when implemented on a sufficiently powerful quantum computer, can factor large integers in polynomial time rather than exponential time [5]. This would effectively break RSA encryption. Similarly, quantum computers running variants of Shor's algorithm could solve the discrete logarithm problem that underlies Elliptic Curve Cryptography (ECC), rendering these systems insecure as well. Recent advancements in quantum computing include:

- IBM's development of quantum processors with increasing qubit counts
- Google's demonstration of quantum supremacy in 2019
- Ongoing research into error correction for quantum computing.

While practical quantum computers capable of breaking RSA remain several years away, their development represents an existential threat to current public-key infrastructure.

4 Defense Mechanisms

4.1 Post-Quantum Cryptography

In response to the quantum threat, researchers have developed several promising approaches to post-quantum cryptography:

- Lattice-Based Cryptography: Systems like NTRU and CRYSTALS-Kyber base their security on the difficulty of solving certain lattice problems, which are believed to be resistant to quantum attacks [6].
- Hash-Based Signatures: Schemes such as SPHINCS+ derive their security from the properties of cryptographic hash functions.
- Code-Based Cryptography: Systems like McEliece use error-correcting codes to establish security.

The National Institute of Standards and Technology (NIST) is currently standardizing post-quantum cryptographic algorithms, with final standards expected by 2024-2025.

4.2 4.2 Advanced Encryption Techniques

Beyond post-quantum approaches, other defensive strategies include:

- Memory-Hard Functions: Password hashing algorithms like Argon2 and scrypt are designed to be resistant to GPU-based parallel attacks by requiring significant memory resources.
- Multi-Party Computation (MPC): Distributes cryptographic operations across multiple parties so that no single party possesses the complete key.
- Homomorphic Encryption: Allows computation on encrypted data without decryption, reducing exposure of sensitive information.

© 2026 Sagar Patel. All rights reserved. Do not copy or redistribute without permission.

4.2.1 Implementation Hardening

Even the most mathematically secure algorithms can be vulnerable to implementation flaws. Countermeasures include:

- Side-Channel Attack Prevention: Implementing constant-time algorithms and protecting against power analysis, electromagnetic leakage, and acoustic analysis.
- Secure Hardware Elements: Using hardware security modules (HSMs) and trusted platform modules (TPMs) to store and process cryptographic keys

4.3 Challenges and Future Trends

Several challenges and trends will shape the future landscape of encryption and parallel computing:

4.3.1 Challenges

- Transition to Post-Quantum Cryptography: Organizations face significant challenges in migrating existing systems to quantum-resistant algorithms.
- Performance Trade-offs: Most post-quantum algorithms require more computational resources and larger key sizes than current approaches.
- Hardware Acceleration for Cryptanalysis: Specialized ASICs and increasingly powerful GPUs continue to reduce the cost of attacking certain encryption methods.

4.4 Future Trends

- Cryptographic Flexibility: Systems designed to quickly adapt to new cryptographic algorithms as vulnerabilities are discovered.
- AI-Enhanced Cryptanalysis: Machine learning techniques being applied to discover patterns and weaknesses in encryption implementations.
- Quantum Key Distribution (QKD): Physical layer encryption using quantum properties to detect eavesdropping attempts.

5 Conclusion

Parallel computing has reshaped cryptography, enabling both stronger attacks and defenses. GPU-accelerated password cracking and quantum threats pose significant risks, making post-quantum cryptography essential. Security now depends not just on algorithms but also on their implementation and available computing power. As parallel computing advances, the cryptographic arms race will intensify, requiring continuous research to protect digital systems.

Reference

[1] O. Schirokauer, "The impact of the number field sieve on the discrete logarithm problem in finite fields," Proceedings of the International Algorithmic Number Theory Symposium, 2002.

[2] RTX 4090 Benchmark, GitHub Gist, 2024.

[3] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," SIAM Journal on Computing, vol. 26, no. 5, pp. 1484–1509, 1997.

[4] D. J. Bernstein and T. Lange, "Post-Quantum Cryptography," Nature, vol. 549, no. 7671, pp. 188–194, 2017.

© 2026 Sagar Patel. All rights reserved. Do not copy or redistribute without permission.

[5] M. J. Dworkin et al., “Status Report on the Second Round of the NIST Post-Quantum Cryptography Standardization Process,” NISTIR 8309, 2020.

[6] A. Biryukov, D. Dinu, and D. Khovratovich, “Argon2: New Generation of Memory-Hard Functions for Password Hashing and Other Applications,” IEEE European Symposium on Security and Privacy, 2016.

[7] M. Abhinav, “The Evolution of Parallel Programming,” Medium, 2024.

[8] Hashcat Cracking Tool, Official Website, 2024.

[9] John the Ripper Cracking Tool, Official Website, 2024.

© 2026 Sagar Patel. All rights reserved. Do not copy or redistribute without permission.