Proof of Higher Computational Complexity of RSA Decryption Compared to Encryption

June 29, 2025

1 Introduction

This document proves why RSA decryption is computationally more intensive than RSA encryption, explaining the observed time difference in a benchmark of a Shamir's Secret Sharing scheme with RSA encryption. The benchmark shows decryption times (e.g., 38.2 ms for 30 shares at 100% threshold) significantly exceeding encryption times (e.g., 2.7 ms).

2 RSA Algorithm

RSA is an asymmetric encryption algorithm based on modular exponentiation. Let:

- $n = p \cdot q$, the modulus, where p and q are large primes.
- e, the public exponent, typically small (e.g., $e = 65537$).
- d, the private exponent, such that $d \cdot e \equiv 1 \pmod{\phi(n)}$, where $\phi(n) = (p-1)(q-1)$.
- m, the plaintext (e.g., a share in Shamir's scheme).
- c, the ciphertext.

Encryption computes $c = m^e \mod n$, and decryption computes $m = c^d \mod n$.

3 Modular Exponentiation Complexity

RSA relies on modular exponentiation, typically implemented using the square-and-multiply algorithm. For $x^y \mod n$:

- $|y| = \lceil \log_2(y) \rceil$, the bit length of the exponent.
- $|n|$, the bit length of the modulus (e.g., 2048 for a 2048-bit key).

The algorithm requires:

- $\approx |y|$ squarings, each $O(|n|^2)$.
- $\approx \frac{|y|}{2}$ multiplications (for 1s in y's binary form), each $O(|n|^2)$.

Total complexity:

$$O(|y| \cdot |n|^2)$$

4 Comparing Exponents e and d

- Public Exponent e: Commonly $e = 65537 = 2^{16} + 1$.
  - Bit length: $|e| = \lceil \log_2(65537) \rceil = 17$.
  - Hamming weight: Binary 10000000000000001_2, with 2 ones.
  - Operations: ≈ 17 squarings, ≈ 2 multiplications.
- Private Exponent d: Computed as $d = e^{-1} \mod \phi(n)$.
  - Since $\phi(n) \approx n$, $|d| \approx |n| \approx 2048$.
  - Hamming weight: $\approx \frac{|d|}{2} \approx 1024$ (assuming random d).
  - Operations: ≈ 2048 squarings, ≈ 1024 multiplications.

5 Complexity Comparison

- Encryption: Exponent e, complexity $O(17 \cdot |n|^2)$.
- Decryption: Exponent d, complexity $O(2048 \cdot |n|^2)$.

The ratio of operations:

$$\frac{2048 + 1024}{17 + 2} \approx 165$$

Decryption requires approximately 165 times more operations than encryption for a 2048-bit key.

6 Application to the Code

In the code:

- Each share is encrypted with the public key ($e = 65537$).
- Each share is decrypted with the private key ($d \approx n$).
- For n shares and threshold t, encryption performs n exponentiations with e, and decryption performs t exponentiations with d.

Benchmark example (30 shares, 100% threshold):

- Encryption: 2.7 ms for 30 shares (≈ 0.09 ms per share).
- Decryption: 38.2 ms for 30 shares (≈ 1.27 ms per share).
- Per-share decryption is $\approx \frac{1.27}{0.09} \approx 14$ times slower, less than the theoretical maximum due to optimizations.

7 Conclusion

RSA decryption is computationally more intensive than encryption because $|d| \approx |n| \gg |e|$, leading to $O(|n| \cdot |n|^2)$ vs. $O(|e| \cdot |n|^2)$ complexity. For a 2048-bit key with $e = 65537$, decryption requires approximately 120–165 times more operations, explaining the observed benchmark results where decryption times significantly exceed encryption times.