Example 11 — RSA as a Partially Homomorphic System

Concept Refresher: What is "Homomorphic Encryption"?

A cryptosystem is said to be *homomorphic* if we can perform certain mathematical operations on encrypted data without decrypting it first. In simpler words: *The math "works through" the encryption.*

For example, suppose encrypting a number M gives $E(M)$. If the system is **additively homomorphic**, then

$$E(M_1 + M_2) = E(M_1) \cdot E(M_2).$$

If it is **multiplicatively homomorphic**, then

$$E(M_1 \times M_2) = E(M_1) \times E(M_2).$$

RSA happens to be *multiplicatively homomorphic*, but not additively so. Let's explore why.

Step-by-Step Walkthrough

Given: Public key (n, e), where $n = pq$ and e is relatively prime to $(p-1)(q-1)$. RSA encryption function:

$$E_{(n,e)}(M) = M^e \bmod n.$$

Let M_1 and M_2 be plaintexts such that $0 \le M_1, M_2 < n$.

Then,

$$E(M_1) \cdot E(M_2) = (M_1^e \bmod n) \cdot (M_2^e \bmod n) \bmod n = (M_1 M_2)^e \bmod n = E(M_1 M_2).$$

Interpretation: Multiplying two ciphertexts corresponds to multiplying their plaintexts before encryption! This is the "magic trick" of RSA's partial homomorphism.

Important distinction:

$$E(M_1) + E(M_2) \neq E(M_1 + M_2)$$

so RSA is not additively homomorphic. Only multiplication "passes through" the encryption function.

Example: Demonstrating RSA's Multiplicative Homomorphism

Let $(n, e, d) = (77, 7, 43)$. Encrypt two plaintext messages $M_1 = 5$ and $M_2 = 9$.

Step 1. Encrypt each:

$$E(5) = 5^7 \bmod 77 = 78125 \bmod 77 = 36.$$

$$E(9) = 9^7 \bmod 77 = 4782969 \bmod 77 = 71.$$

Step 2. Multiply ciphertexts:

$$E(5) \cdot E(9) \bmod 77 = 36 \cdot 71 \bmod 77 = 2556 \bmod 77 = 15.$$

Step 3. Multiply plaintexts and encrypt:

$$E(5 \cdot 9) = E(45) = 45^7 \bmod 77 = 15.$$

They match! So indeed, RSA is multiplicatively homomorphic.

Practice Problems

Problem A (Easier). Using $(n, e) = (77, 7)$, compute $E(2)$ and $E(3)$, then verify that

$$E(2 \cdot 3) = E(2) \cdot E(3) \pmod{77}.$$

Hint: $E(M) = M^7 \bmod 77$.

Problem B (Similar). With $(n, e) = (2537, 13)$ (the RSA system from earlier examples), let $M_1 = 14$ and $M_2 = 15$. Show numerically that

$$E(M_1) \cdot E(M_2) \equiv E(M_1 M_2) \pmod{2537}.$$

Use a calculator or write a small Python snippet if needed.

Problem C (Harder Challenge). Explain, in your own words:

1. Why RSA cannot be additively homomorphic.
2. How this limitation affects using RSA for cloud computations or secure voting.
3. Why Craig Gentry's 2009 breakthrough (fully homomorphic encryption) was such a big deal.

Helpful Tips and Intuition

- **Think of encryption as a "mathematical disguise."** RSA preserves multiplication but not addition, so we can "multiply in disguise" but not "add in disguise."
- **Be cautious with modular arithmetic.** When results are large, always reduce modulo n before the next step.
- **Modern context.** Homomorphic encryption allows computation on encrypted data — like asking Google to calculate your taxes without revealing your salary. RSA can't do that completely, but it was the seed of that dream.
- **Curiosity spark.** Look up Craig Gentry's 2009 Ph.D. thesis from Stanford — it's the start of lattice-based, fully homomorphic encryption (FHE).