**Walkthrough: Crypto Fortress Challenge**

**Challenge Overview**

**Objective**: This document provides a detailed walkthrough of the "CTF Chakravyuha Challenge." This challenge incorporates multiple cryptographic layers, including **RSA encryption, AES encryption, XOR masking, Base64 encoding, and hash-based transformations.** The objective is to systematically reverse these cryptographic layers to recover the original flag.

--------------------------------------------------------------------------------------------------------------------------------------

**Step 1: RSA Key Generation**

1. Two large prime numbers p and q are generated.
2. The modulus N is computed as N = p \* q.
3. The public exponent e = 65537 is chosen.
4. The private key d is calculated using modular inverse.

**Key Takeaway:** The RSA key pair consists of (N, e) for encryption and d for decryption.

--------------------------------------------------------------------------------------------------------------------------------------

**Step 2: RSA Encryption of the Flag**

1. The flag (flag{CTF Chakravyuha}) is converted into a long integer.
2. RSA encryption is performed: ciphertext = message^e mod N.

**Key Takeaway:** To decrypt the RSA ciphertext, use message = ciphertext^d mod N.

--------------------------------------------------------------------------------------------------------------------------------------

**Step 3: XOR Masking**

1. A **SHA-256 hash key** (32 bytes) is generated randomly.
2. The RSA ciphertext is **XORed** with the hash-derived key.

**Key Takeaway:** XOR operations can be reversed by applying the same XOR key again.

--------------------------------------------------------------------------------------------------------------------------------------

**Step 4: Base64 Encoding**

1. The XOR-masked RSA ciphertext is encoded using **Base64**.

**Key Takeaway:** Base64 encoding is **not encryption** and can be easily reversed.

--------------------------------------------------------------------------------------------------------------------------------------

**Step 5: AES Encryption Layer**

1. A random **AES key** (16 bytes) is generated.
2. The Base64-encoded ciphertext is **padded** and encrypted using **AES in CBC mode**.
3. The **IV (Initialization Vector)** is prepended to the ciphertext for decryption.
4. **Key Takeaway:** AES decryption requires both the AES key and IV.

--------------------------------------------------------------------------------------------------------------------------------------

**Step 6: Nested Obfuscation**

1. Three random transformations are applied:
   * **Base64 encoding**
   * **XOR with cryptographic hash mask**
2. These transformations are **randomized** to increase complexity.

**Key Takeaway:** Undo these operations by reversing the order, decoding Base64, and applying XOR masks correctly.

--------------------------------------------------------------------------------------------------------------------------------------

**Step 7: Final Hash-Based Masking**

1. A cryptographic hash-based transformation is applied:
   * SHA-256 hash is computed using a salt (ai\_resistant\_layer).
   * The result is XORed with the obfuscated flag.

Key Takeaway: Since hashing is one-way, the flag cannot be recovered beyond this stage.

--------------------------------------------------------------------------------------------------------------------------------------**Challenge Breakdown:**

How to Reverse the Process :

To retrieve the original flag:

1. **Undo Hash Masking (Not Possible Beyond This Point)** - If the final hash transformation is applied, recovery is impossible.
2. **Undo Nested Obfuscation:** Identify and reverse **Base64 decodings and XOR transformations**.
3. **AES Decryption:** Extract the IV and decrypt the ciphertext using the **AES key**.
4. **Base64 Decode XOR-Masked RSA Ciphertext.**
5. **Apply XOR Mask to Retrieve Original RSA Ciphertext.**
6. **Decrypt Using RSA Private Key:** message = ciphertext^d mod N.
7. **Convert Integer to Bytes to Recover the Flag.**

--------------------------------------------------------------------------------------------------------------------------------------

**Challenge Data Output –**

* **RSA Public Key (N, e):** Provided in challenge output.
* **Final Encrypted Flag (Hex):** Provided in challenge output.

--------------------------------------------------------------------------------------------------------------------------------------

**Conclusion**

This challenge demonstrates the importance of combining cryptographic techniques to enhance security. It showcases:

* **Asymmetric Encryption (RSA)** for securing keys.
* **Symmetric Encryption (AES)** for fast and secure data encryption.
* **XOR Masking** for additional obfuscation.
* **Base64 Encoding** as a simple transformation.
* **Hash-Based Masking** to introduce irreversible obfuscation.

To solve the challenge, carefully reverse each transformation layer while considering the encryption mechanisms used.

Good luck!