### 3. Defense Techniques against RSA Factorization Attacks

#### 3.1 Principle of Defense

The most effective defense against factorization attacks such as Pollard’s rho is to ensure **secure RSA key generation**. As Dan Boneh (1999) highlighted in his seminal review, the security of the RSA algorithm fundamentally depends on the computational difficulty of factoring the modulus n = p × q. Among all known attacks, factorization remains the most direct and powerful threat. If the modulus n is too small, or if the prime numbers p and q are poorly chosen, theoretical attacks can easily become practical.

Therefore, the cornerstone of RSA security lies not in the encryption or decryption process itself, but in **the generation of strong cryptographic keys**. The main defense principle is to carefully select primes p and q so that their product n can resist all known efficient factorization algorithms.

In practice, the defense strategy focuses on two main objectives:

1. **Increase computational complexity:** Ensure that the modulus n is large enough so that any factorization algorithm (including Pollard’s rho and the Number Field Sieve) would require an impractical amount of time and computing power.
2. **Eliminate structural weaknesses:** Make sure that the primes p and q have no special mathematical structure that could be exploited by specific algorithms such as Pollard’s p−1 method.

#### 3.2 Detailed Defense Methods

##### 3.2.1 Using a Sufficiently Large Modulus

Choosing a sufficiently large modulus n is the most straightforward and effective way to resist factorization attacks. The length of the RSA key directly determines the level of computational effort required to break it. As Nisha and Farik (2017) point out, a **2048-bit RSA key** is currently recognized as the minimum secure standard and is widely used in modern systems such as TLS/SSL. For applications that require long-term security or protection against nation-state adversaries, **3072-bit or 4096-bit keys** are strongly recommended.

The security rationale is based on the time complexity of factorization algorithms. For example, the time complexity of Pollard’s rho algorithm is approximately O(), where p is the smaller prime factor of n. Therefore, doubling the key length does not simply double the difficulty—it increases it **exponentially**, making real-world attacks computationally infeasible. By increasing the key size, organizations effectively raise the cost of attack to a point beyond realistic limits, forming a solid defensive barrier against factorization attempts.

##### 3.2.2 Generating Strong Primes

In the RSA cryptosystem, a strong prime is not only a large random prime number but one that satisfies specific mathematical conditions to resist specialized factorization methods. A strong prime p should meet the following requirements:

1. ( p ) itself is a large prime number;
2. ( p−1 ) contains a large prime factor ( r ), which makes Pollard’s p−1 attack inefficient;
3. ideally, ( p+1 ) also has a large prime factor to resist algorithms such as Williams’ p+1 method;
4. ( r−1 ) itself contains a large prime factor for deeper protection.

To achieve this, the RSA key generation process should include several key steps:

1. Use a **cryptographically secure pseudorandom number generator (CSPRNG)** to produce candidate numbers with sufficient entropy, ensuring they cannot be predicted.
2. Apply rigorous **probabilistic primality tests**, such as the Miller–Rabin test, to ensure that both ( p ) and ( q ) are very likely to be prime.
3. For modern key sizes (≥2048 bits), strong primes occur naturally with high probability. However, verifying strong-prime conditions remains a best practice to further prevent potential structural weaknesses and resist side-channel attacks.

By following these measures, the RSA cryptosystem can maintain a strong level of security even in the face of advanced factorization algorithms.

Reference：

1. Boneh, D., 1999. Twenty years of attacks on the RSA cryptosystem. Notices of the American Mathematical Society, 46(2), pp.203–213.
2. Nisha, S. and Farik, M., 2017. RSA public key cryptography algorithm – A review. International Journal of Scientific & Technology Research, 6(7), pp.187–191.