**防御针对D-H攻击MITM的方式**

**3.2 Core Defense Scheme: Pre-Shared Key Based Authentication Mechanism**

Research has demonstrated that the traditional Diffie-Hellman protocol, due to its lack of an authentication mechanism, allows attackers to easily forge public keys and perform man-in-the-middle interception [1]. To effectively counter the man-in-the-middle attack threats faced by the Diffie-Hellman key exchange protocol while ensuring the scheme exhibits good engineering feasibility and demonstration effectiveness, this study adopts a pre-shared key based authentication mechanism as the core defense scheme. This scheme fundamentally resolves the issue of authenticating communicating parties' identities by integrating a lightweight identity verification mechanism, while fully preserving the original key agreement functionality of the DH protocol.

**3.2.1 Technical Principle**

At the technical implementation level, the core of this defense mechanism lies in integrating message authentication functionality into the standard DH key exchange process. Before initiating key exchange, the two communicating parties must first securely pre-share an authentication key via a secure channel. This key operates independently of the key agreement process and is specifically used to authenticate the exchanged public key data. In concrete implementation, when transmitting its DH public key, the sending party uses the pre-shared key to generate a corresponding message authentication code (MAC) via an HMAC algorithm. Upon receiving the data, the receiving party employs the same key and algorithm to verify the public key. If the MAC verification fails, it indicates that a malicious attacker may be attempting to tamper with or forge the public key within the communication channel, prompting the receiver to immediately terminate the session establishment process.

**3.2.2 Implementation Procedure**

The implementation of this defense mechanism encompasses the following key steps:

1. **Initialization Phase:** The communicating parties (e.g., Alice and Bob) mutually agree upon a pre-shared key through a secure out-of-band channel prior to establishing communication. This key is typically a string possessing a certain level of complexity.
2. **Key Exchange and Authentication Phase:**  
   Alice generates her DH public key A and computes the authentication code HMAC(PSK, A), where PSK is the pre-shared key. Subsequently, she transmits the combination of the public key and the authentication code (A, HMAC(PSK, A)) to Bob.  
   Upon receiving the data, Bob recalculates the HMAC value using his locally stored PSK and the received public key A'. If the computed result matches the received authentication code, the public key is deemed authentic and originating from Alice; otherwise, a man-in-the-middle attack is assumed present in the channel, and the session is terminated immediately.  
   Bob employs the same mechanism to send his DH public key B and the corresponding authentication code to Alice, who performs an analogous verification operation. This mutual authentication mechanism aligns with the identity verification concepts proposed in recent improved Diffie-Hellman schemes [2].
3. **Secure Session Establishment:** A subsequent secure communication channel, utilizing the shared key negotiated via the DH protocol, is established only after both parties successfully complete the aforementioned PSK authentication. Authentication failure by either party results in connection termination, thereby preventing any unverified entity from participating in the key agreement.

**3.2.3 Scheme Advantage Analysis**

From a practical application perspective, this defense scheme offers multiple significant advantages. Regarding implementation complexity, it requires only the integration of an HMAC computation module into the existing protocol stack, eliminating the need for deploying complex Public Key Infrastructure (PKI) and substantially reducing engineering complexity and deployment costs. In terms of demonstration value, the binary nature of the authentication outcome makes the attack detection process intuitively visible, rendering it highly suitable for teaching and demonstration environments. Architecturally, the scheme is fully self-contained, operating independently of external authentication services, making it particularly apt for temporary or closed communication scenarios. This self-contained characteristic has proven its practical utility in certain wireless communication contexts [3]. Security-wise, provided the pre-shared key remains confidential, the scheme can effectively identify and block various types of identity forgery attacks, thereby providing reliable identity authentication assurance for the key exchange process.

1. Khader, A.S. and Lai, D., 2015, April. Preventing man-in-the-middle attack in Diffie-Hellman key exchange protocol. In *2015 22nd international conference on telecommunications (ICT)* (pp. 204-208). IEEE.
2. Galu, T.S., Adeyelu, A.A. & Otor, S.U. (2024) 'An Improved Deffie Hellman Scheme for Mitigating an Eavesdropping Attack on a Network'. *International Journal of Innovative Science and Research Technology (IJISRT)*, 9(4), pp. 3101-3108. Available at: <https://doi.org/10.38124/ijisrt/IJISRT24APR2479>
3. Shen, W., Cheng, Y., Yin, B., Du, J. and Cao, X., 2021. Diffie-Hellman in the air: A link layer approach for in-band wireless pairing. *IEEE Transactions on Vehicular Technology*, *70*(11), pp.11894-11907.

### 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[1]. 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[2]. 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.