

# Cryptography Lab 2

Jordi Nadeu Ferran

## 1 Introduction

The objective of this laboratory is to simulate a Man-in-the-Middle (MITM) attack on the Diffie-Hellman (DH) key exchange process.

## 2 How the attack works

To simulate a Man-in-the-Middle (MITM) attack using the provided scripts, follow these steps.

We will run the scripts in a particular order and observe the outcomes to check if Eve successfully intercepts the communication between Alice and Bob.

1. Run the setup script:

```
python dh_config.py
```

This will generate the shared Diffie-Hellman parameters and create the shared directory that we use to simulate to exchange the messages via a network.

2. Run the key exchange script as Alice:

```
python dh_key_exchange.py alice
```

Alice will generate her Diffie-Hellman key pair, send her public key to Bob, and then wait for Bob's public key. The script also generates a shared key with Bob using their public keys and encrypts a message to send to Bob.

3. Run the key exchange script as Bob:

```
python dh_key_exchange.py bob
```

Bob will read Alice's public key, send his own public key to Alice, and then compute the shared key with Alice's public key. Once the shared key is established, Bob decrypts Alice's message and sends back an encrypted response.

4. Run the key exchange script as Eve (MITM):

```
python dh_key_exchange.py eve
```

Eve will intercept Alice's public key, substitute Bob's public key with her own (fake public key), and then intercept the message from Alice. She will also send Bob a fake public key. The key here is that Eve will generate two different key pairs for her interaction with Alice and Bob.

In summary, the expected behaviour for each role would be as follows:

**Alice:**

- Sends her public key to Bob and waits for Bob's public key.
- Computes a shared secret with Bob and sends an encrypted message.

**Bob:**

- Receives Alice's public key, sends his public key to Alice.
- Computes a shared secret and sends a response back.

**Eve:**

- Intercepts Alice's public key and sends her own public key to Bob, pretending to be Alice.
- Intercepts the encrypted message sent by Alice, decrypts it, prints the message, and sends a fake encrypted response back to Alice.

### 3 How the mitigation works

To mitigate the MITM (Man-in-the-Middle) attack, we need to introduce measures that detect the interception and modification of communication between Alice and Bob. In a typical Diffie-Hellman key exchange setup, if an attacker intercepts the public keys and replaces them with their own, the exchanged shared keys will be different from what Alice and Bob expect and that will our check in the script.

To do that we need to run the key exchange script as Angel:

```
python dh_key_exchange.py angel
```

The strategies to detect MITM attacks in a Diffie-Hellman key exchange are:

- **Check for Shared Key Mismatch:** Alice and Bob will compute the shared key independently based on their private key and the public key they receive. If Eve is performing a MITM attack, the shared keys Alice and Bob compute will not match.
- **Message Authentication:** To ensure the integrity of the messages and to detect tampering, Alice and Bob can exchange a "nonce" or a simple hash of their message, or use a message authentication code (MAC).
- **Use Certificate Authorities (CAs):** For real-world applications, you would use certificates signed by a trusted CA to validate the authenticity of the public keys.

## 4 Results

Below you can see the result of executing all the scripts in the order explained above.

```
~/Doc/Mas/Sh/masters-degree/CI/lab2 P main +3 114 74 > python dh_key_exchange.py alice
Alice: Public key sent.
Alice: Shared key established.
Alice: Encrypted message sent: Hello Bob, this is Alice!
Alice: Received response: Message received, Alice!

~/Doc/Mas/Sh/masters-degree/CI/lab2 P main +3 114 74 > _
6s masters-degree

~/Doc/Mas/Sh/masters-degree/CI/lab2 P main +3 114 74 > python dh_key_exchange.py bob
Bob: Public key sent.
Bob: Shared key established.
Bob: Received message: Hello Bob, this is Alice!
Bob: Encrypted response sent: Message received, Alice!

~/Doc/Mas/Sh/masters-degree/CI/lab2 P main +3 114 74 > _
masters-degree

~/Doc/Mas/Sh/masters-degree/CI/lab2 P main +3 114 74 > python dh_key_exchange.py eve
Eve: Intercepted Alice's public key and sent Bob her fake key.
Eve: Shared key with Alice computed.
Eve: Intercepted Bob's public key and sent Alice her fake key.
Eve: Shared key with Bob computed.
Eve: Impersonate Alice's message.
Eve: Message from Alice: Hi Bob, this is Alice! ^.^
Eve: Modified message forwarded to Bob.
Eve: Impersonate Bob's message.
Eve: Message from Bob: Message received, Alice! ^.^
Eve: Modified response forwarded to Alice.
```

```
lab2/shared/dh_parameters.pem
Dh Parameters (p): 195157932037088864293651248213594264838948774543682298865149076
65238467074233383046440211612227881925522392141450768680322465701157259283933492
95252075712175356415852107648768515087068895056821097882534158238991908763237483633
0064221398320789387136792953488791678567688741819693818356458793723394888389238532
40181175889874489742077215952867188833062073826653338589714338977491583869416932002
811936779572628984465124278172257986612531448862007699664769838940355454742582482
88850345426536737261917273713631587950913898869637632626524339447475586237612282221
46193165901121227677471814937145985976110737415532143
Dh Parameters (g): 2

~/Doc/Mas/Sh/masters-degree/CI/lab2 P main +3 114 74 > python dh_key_excha
nge.py angel
Angel: Monitoring communication...
Angel: MITM detected! Shared keys do not match!
Angel: All files deleted. Communication blocked.

~/Doc/Mas/Sh/masters-degree/CI/lab2 P main +3 114 74 > _
masters-degree
```