Chiffrement asymétrique RSA

Christophe Viroulaud

Terminale - NSI

Archi 23

Le chiffrement asymétrique de Diffie-Hellman permet d'échanger des clés via un canal non sûr mais ne gère pas les problèmes liés à l'authentification des interlocuteurs.

Comment authentifier avec certitude les participants ?

Sommaire

Chiffrement RSA - Principe

- 1977 : Ron Rivest, Adi Shamir et Len Adleman.
- breveté en 1983 ; expiration du brevet en 2000.
- utilise *des fonctions mathématiques à sens unique* (comme Diffie-Hellman)
- une paire de clés publique et privée.

À retenir

Une fonction à sens unique est une fonction mathématique facile à calculer mais pour laquelle il est très compliqué de retrouver l'antécédent d'une image.

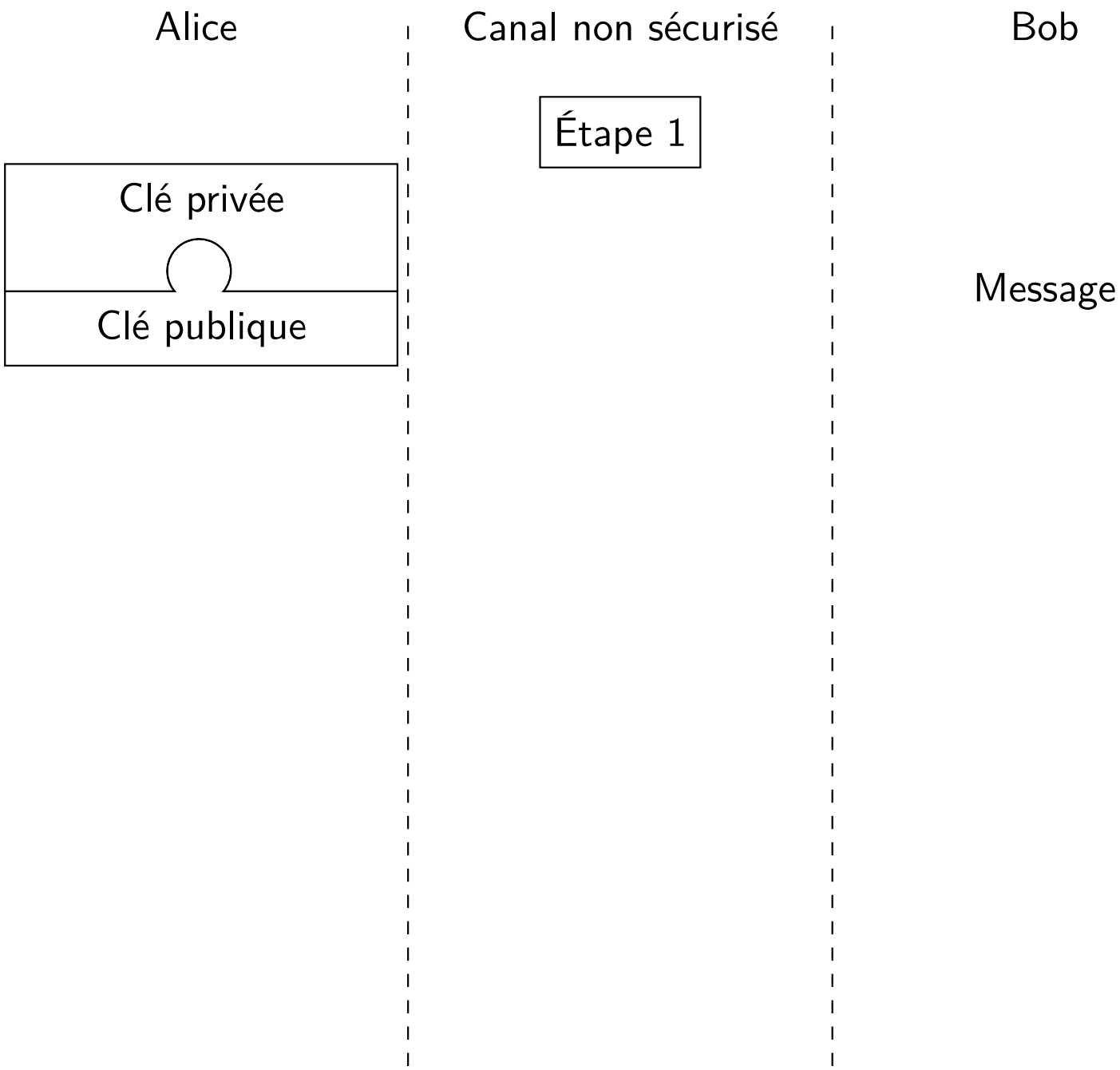
À retenir

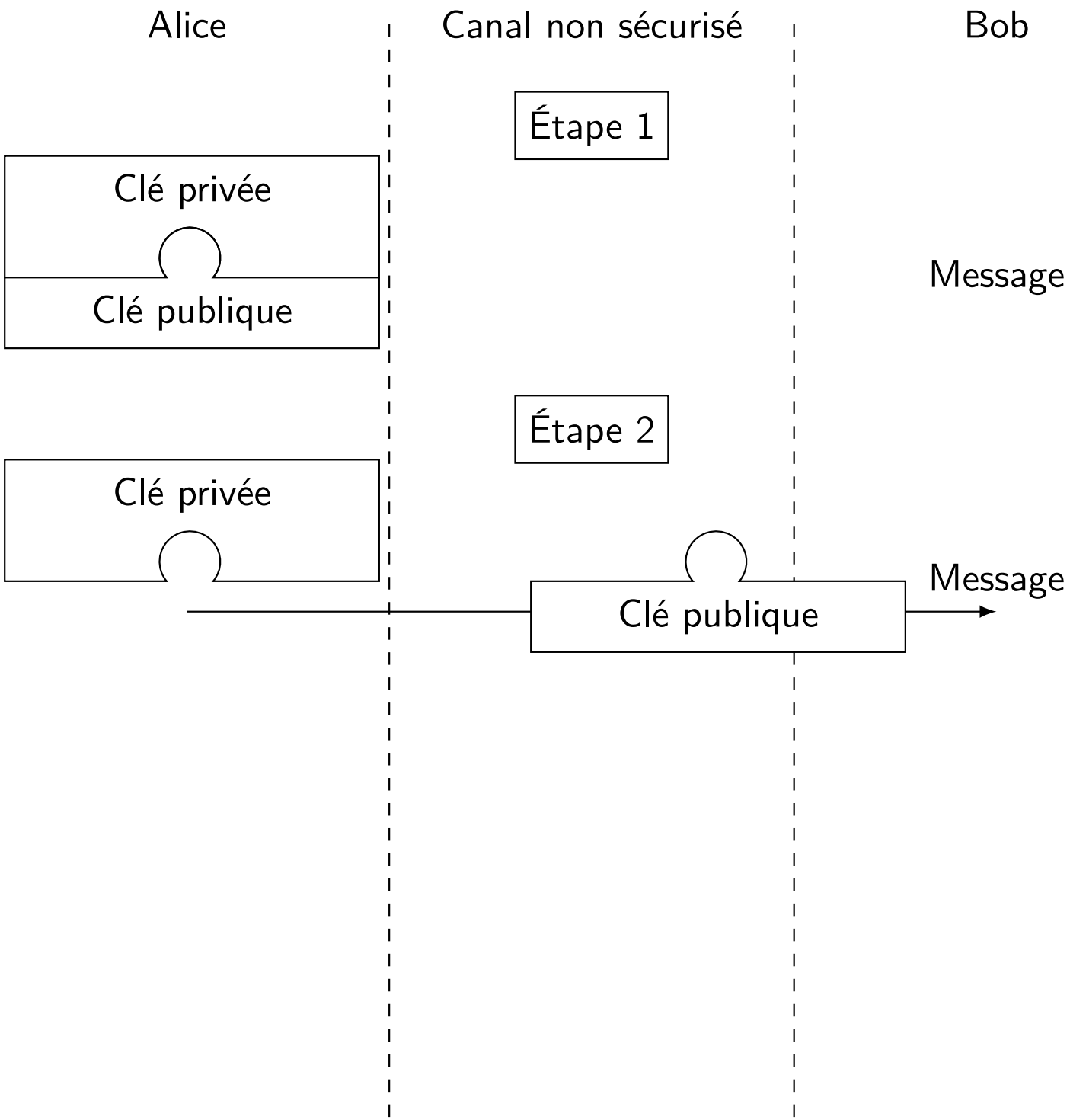
Le principe du protocole RSA s'inspire de la méthode de Diffie-Hellman :

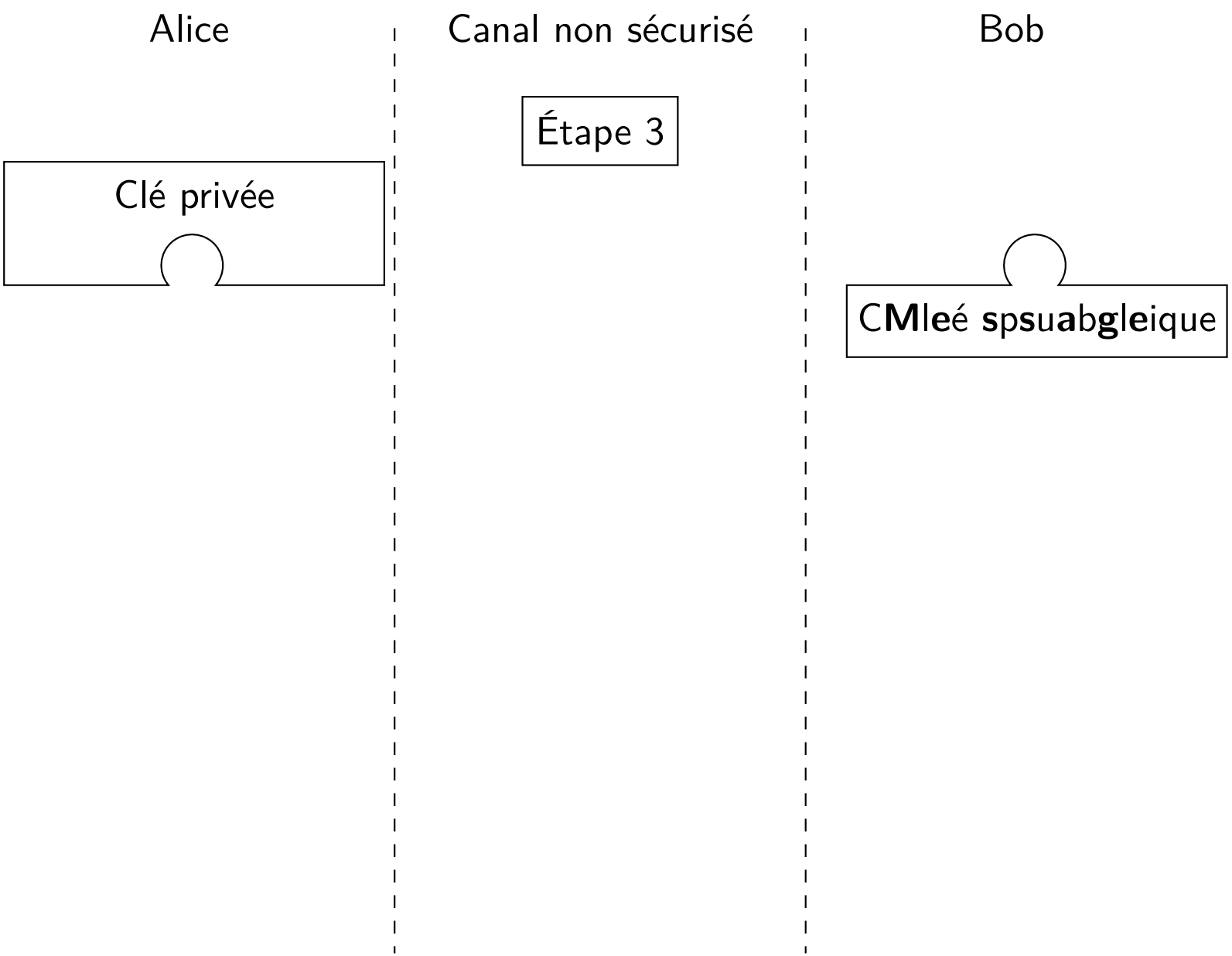
$$K_{priv}(K_{pub}(m)) = K_{pub}(K_{priv}(m)) = m$$

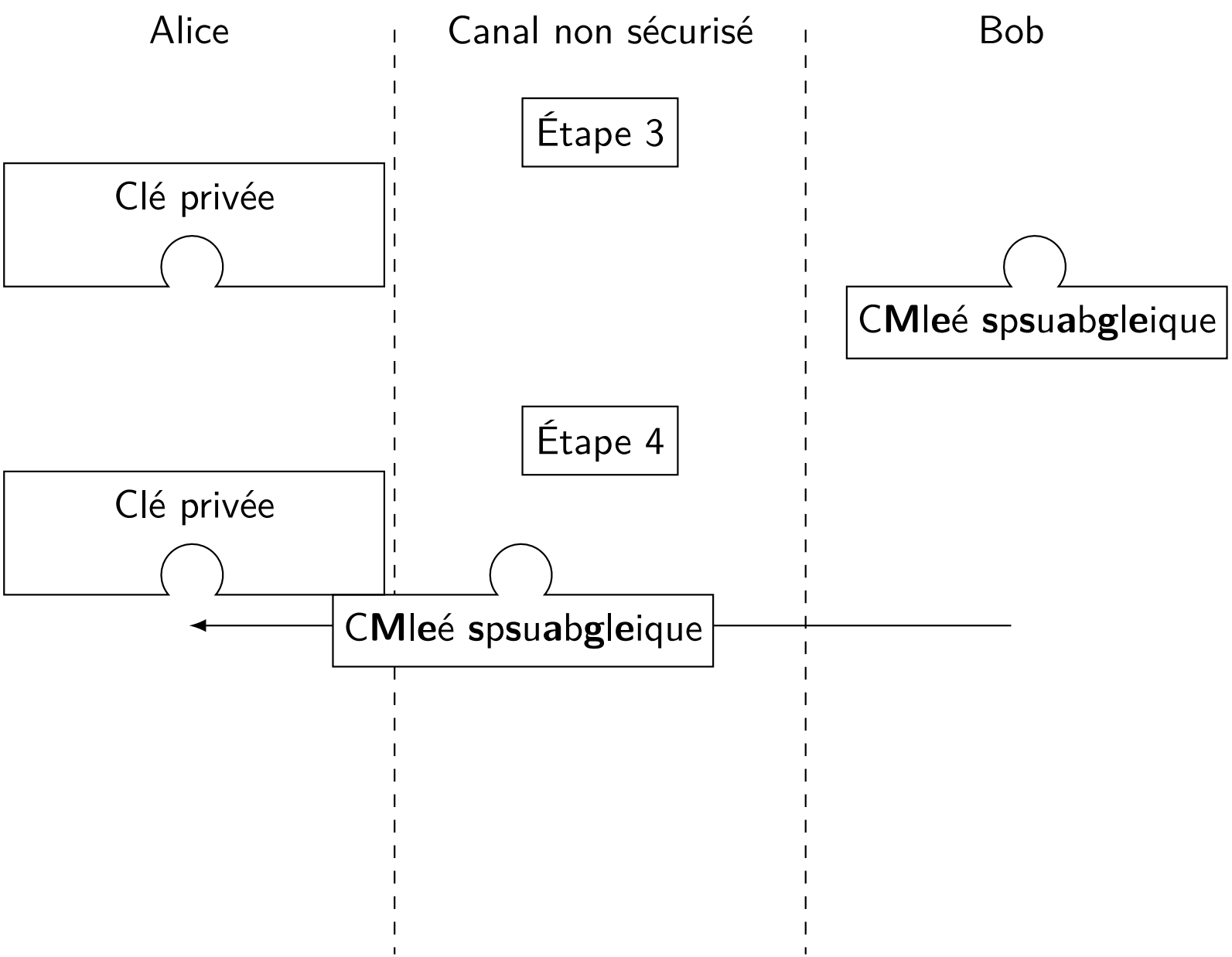
Sommaire

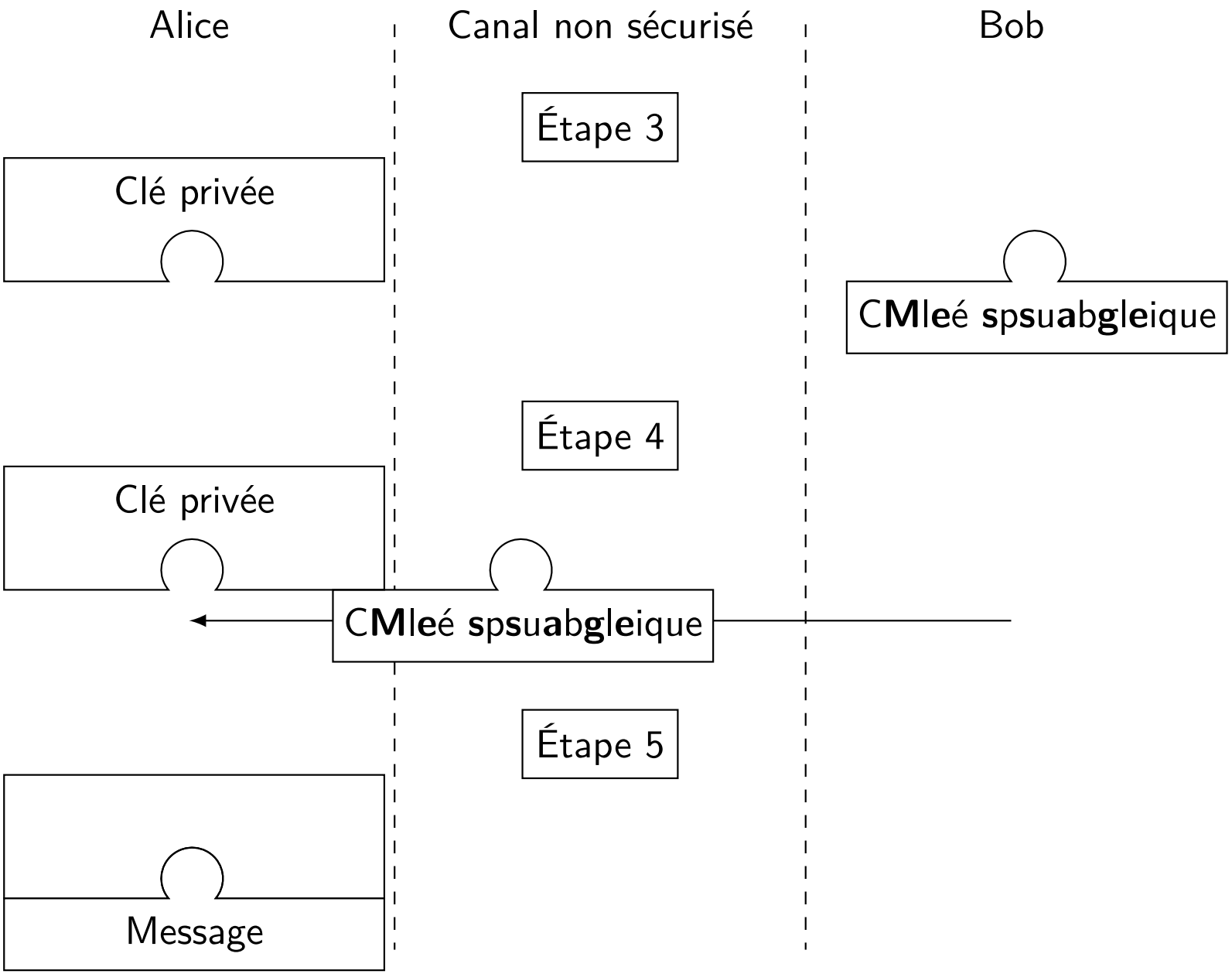
Description








Alice
Canal non sécurisé
Bob
Étape 3
Clé privée
CMleé spsuabgleique
Étape 4
Clé privée
CMleé spsuabgleique



À retenir

Mathématiquement, la fonction respecte les règles suivantes :

- Il est impossible de deviner la clé privée en connaissant la clé publique.
- Il est impossible de deviner le message avec une seule des deux clés.

Sommaire

kid RSA

La création des clés suit un algorithme mathématique complexe. L'université de Rhode Island a produit une version simplifiée (à utilisation pédagogique) pour simuler le protocole.

Activité 1 :

1. Découvrir l'algorithme *kidrsa* sur la page https://tinyurl.com/rsakid
2. Écrire la fonction **`creer_nombre(a: int, b: int, a1: int, b1: int)`** $\rightarrow$ **`dict`** qui renvoie un dictionnaire contenant les clés privée et publique. Chaque clé sera un tuple.
3. Écrire la fonction **`chiffrer(message: int, publique: tuple)`** $\rightarrow$ **`int`** qui encode **`message`** avec la clé publique (e, n).
4. Écrire la fonction **`dechiffrer(message_chiffre: int, privee: tuple)`** $\rightarrow$ **`int`** qui déchiffre **`message_chiffre`** avec la clé privée (d, n).
5. Tester l'algorithme de chiffrage avec un entier (inférieur à n). Le message à chiffrer sera l'entier 538.

Correction


```python
def creer_nombre(a: int, b: int, a1: int, b1: int) -> dict:
    """
    crée un couple clé privée/publique
    Returns:
        dict: {"publique": (e, n), "privee":(d, n)}
    """
    M = a*b-1
    e = a1*M+a
    d = b1*M+b
    n = (e*d)//M
    return {"publique": (e, n), "privee": (d, n)}
```

Code 1 – Alice crée ses clés

Correction

```python
def chiffrer(message: int, publique: tuple) -> int:
    """
    Args:
        message (int)
        publique (tuple): (e, n)

    Returns:
        int: message chiffré
    """
    return (publique[0]*message) % publique[1]
```

Code 2 – Bob chiffre son message avec la clé publique d'Alice

Correction

```
1  def dechiffrer(message_secret: int, privee: tuple)
       -> int:
2      """
3      Args:
4          message_secret (int)
5          privee (tuple): (d, n)
6
7      Returns:
8          int: message déchiffré
9      """
10     return (privee[0]*message_secret) % privee[1]
```

Code 3 – Alice déchiffre le message de Bob avec sa clé privée

```
>>> cles = creer_nombre(9, 11, 5, 8)
>>> cles
{'publique': (499, 4048), 'privee': (795, 4048)}

>>> s = chiffrer(538, cles["publique"])
>>> s
1294

>>> e = dechiffrer(s, cles["privee"])
>>> e
538
```

Pour l'instant, l'algorithme RSA ne fait rien de plus que celui de Diffie-Hellman. Le problème de l'authentification n'est toujours pas résolu.

Sommaire

Authentification des participants

À retenir

Pour certifier l'identité des individus, il faut qu'un **tiers de confiance** (Nestor) intervienne.

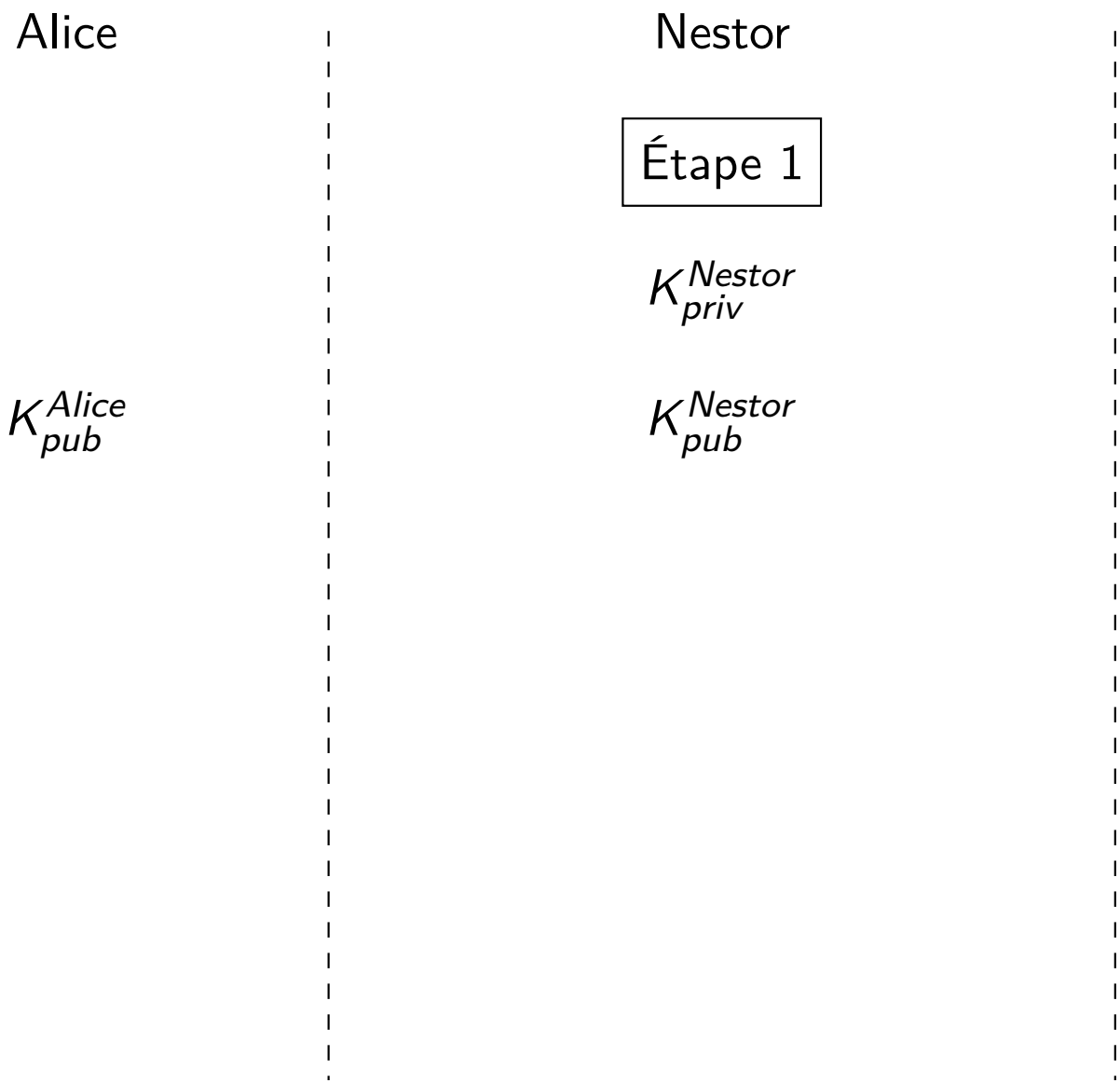


FIGURE 1 – Le tiers de confiance crée un couple de clés privée/publique.

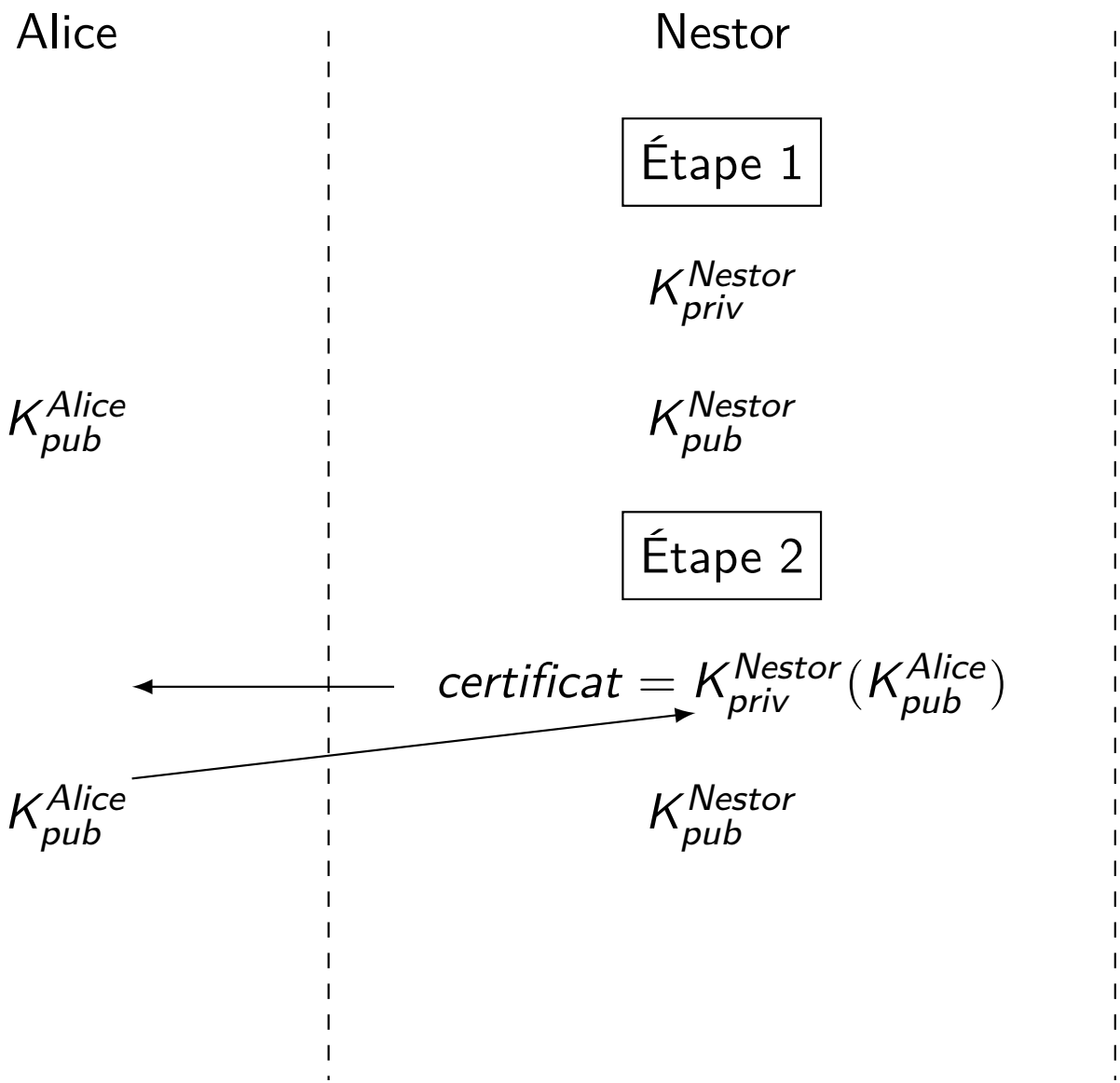


FIGURE 2 – Nestor chiffre la clé publique d'Alice avec sa clé privée : il crée un **certificat**.

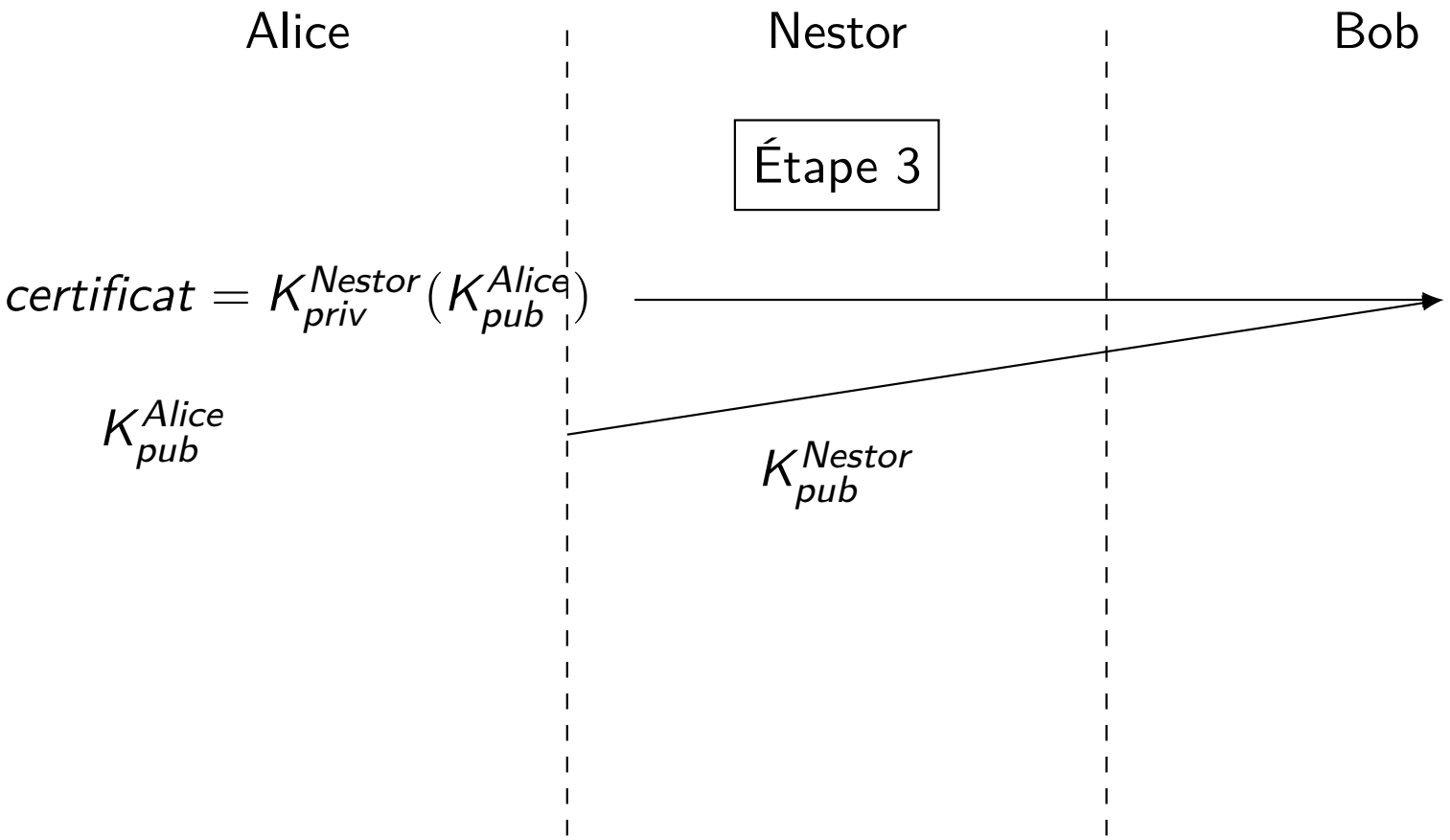


FIGURE 3 – Alice envoie le certificat et sa clé publique en clair.

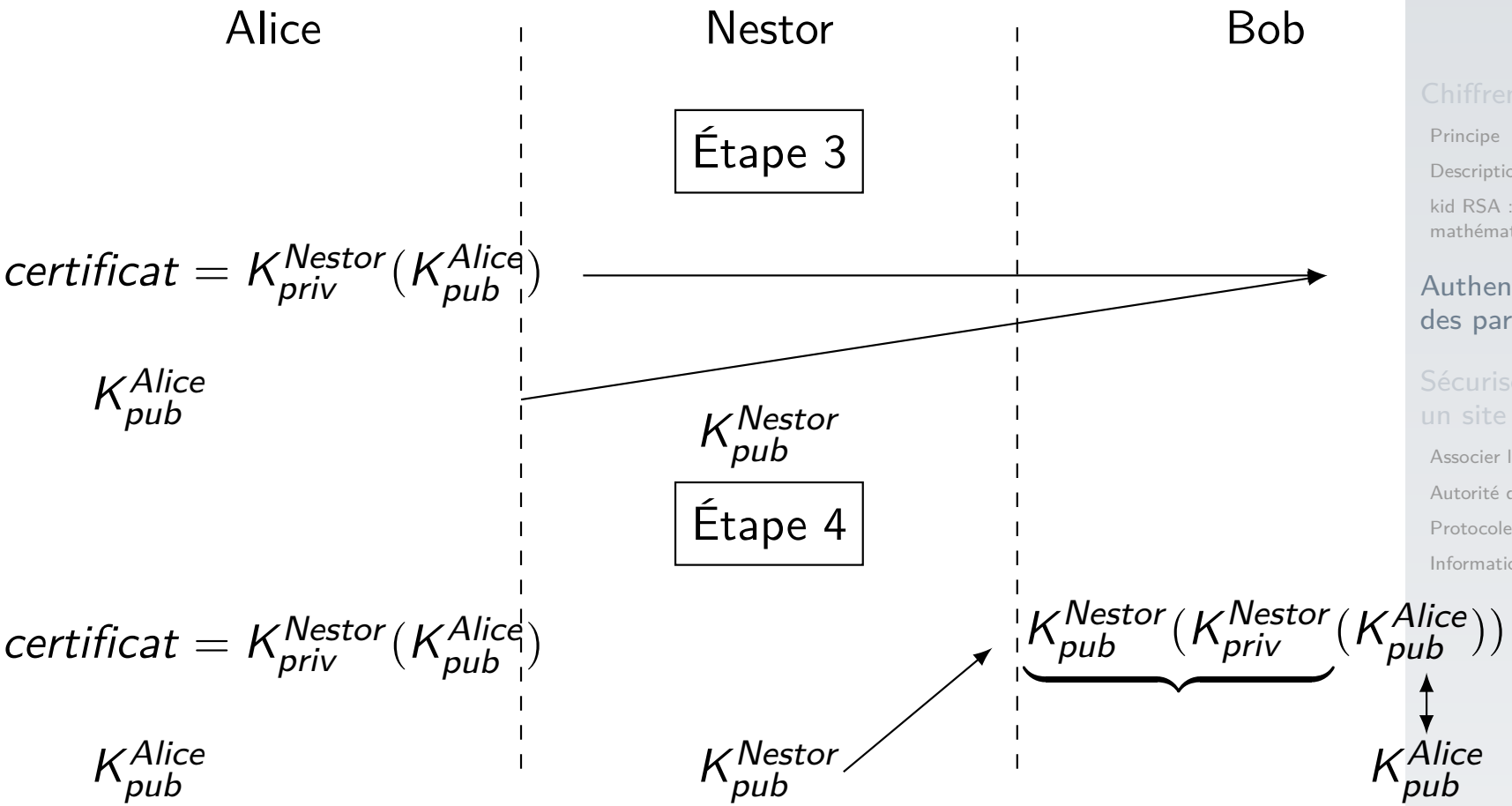


FIGURE 4 – À l'aide de la clé publique de Nestor, Bob déchiffre la clé publique d'Alice (le certificat) et la compare à la clé publique fournie en clair.

Sommaire

Sécuriser l'accès à un site web

L'algorithme RSA :

- **avantage :** permet de sécuriser les données

Sécuriser l'accès à un site web

L'algorithme RSA :

- **avantage :** permet de sécuriser les données
- **avantage :** permet d'authentifier les participants.

Sécuriser l'accès à un site web

L'algorithme RSA :

- **avantage :** permet de sécuriser les données
- **avantage :** permet d'authentifier les participants.
- **inconvénient :** est très coûteux en temps de calcul.

À retenir

On mettra à profit les avantages de chaque type de chiffrement :

- Le chiffrement symétrique, rapide, sera utilisé pour chiffrer les données avec une *clé de chiffrement symétrique*.
- Le chiffrement asymétrique, permettant d'authentifier les participants, sera utilisé pour transmettre la clé de chiffrement symétrique.

Sommaire

Autorité de certification

Une autorité de certification peut être :

- un état,
- une entreprise spécialisée,
- une association à but non lucratif (Let's Encrypt).

Les navigateurs possèdent une copie des clés publiques de ces autorités de certification.

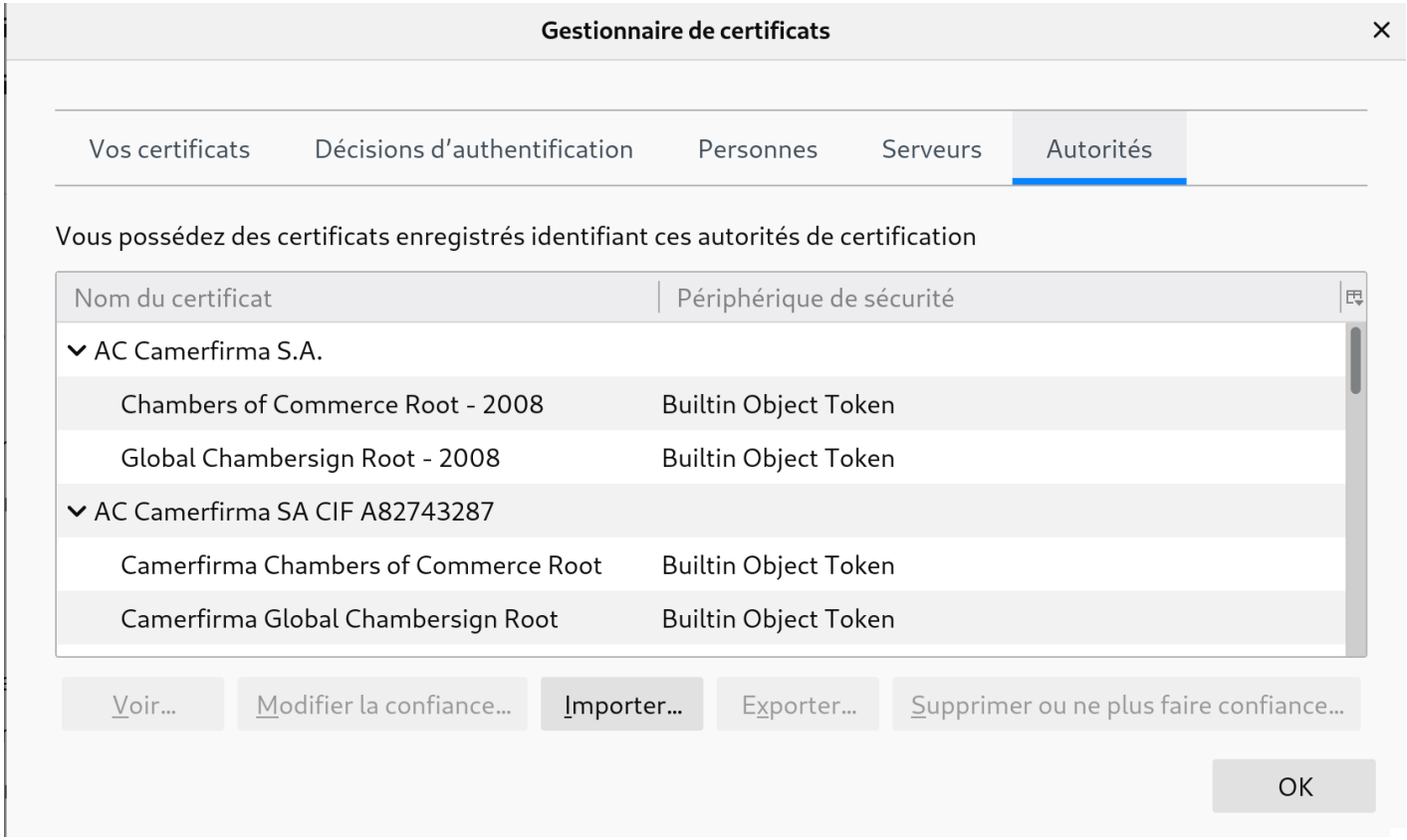


FIGURE 5 – Firefox/préférences/vie privée et sécurité/certificats

Hors programme

En pratique, l'autorité de certification ne signe pas la clé publique entière du site (2048 ou 4096 bits) mais sa somme de contrôle calculée (256 bits) par une fonction de hachage (souvent sha256).

Sommaire

Protocole https

Le protocole *https* ajoute une couche *TLS (Transport Layer Security)* au protocole *http* existant.

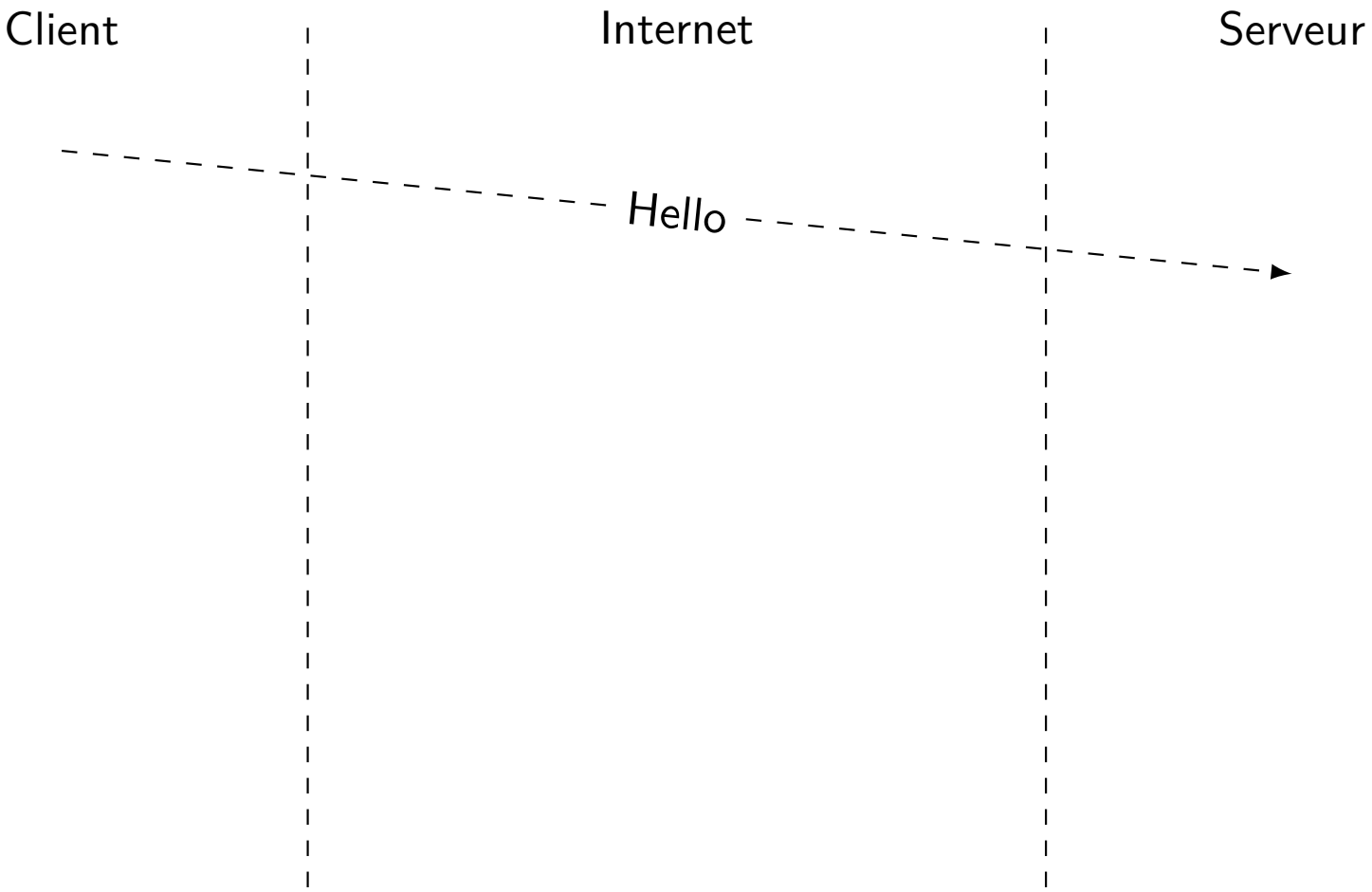


FIGURE 6 – **Hello :** Le navigateur envoie son intention de se connecter et diverses informations techniques.

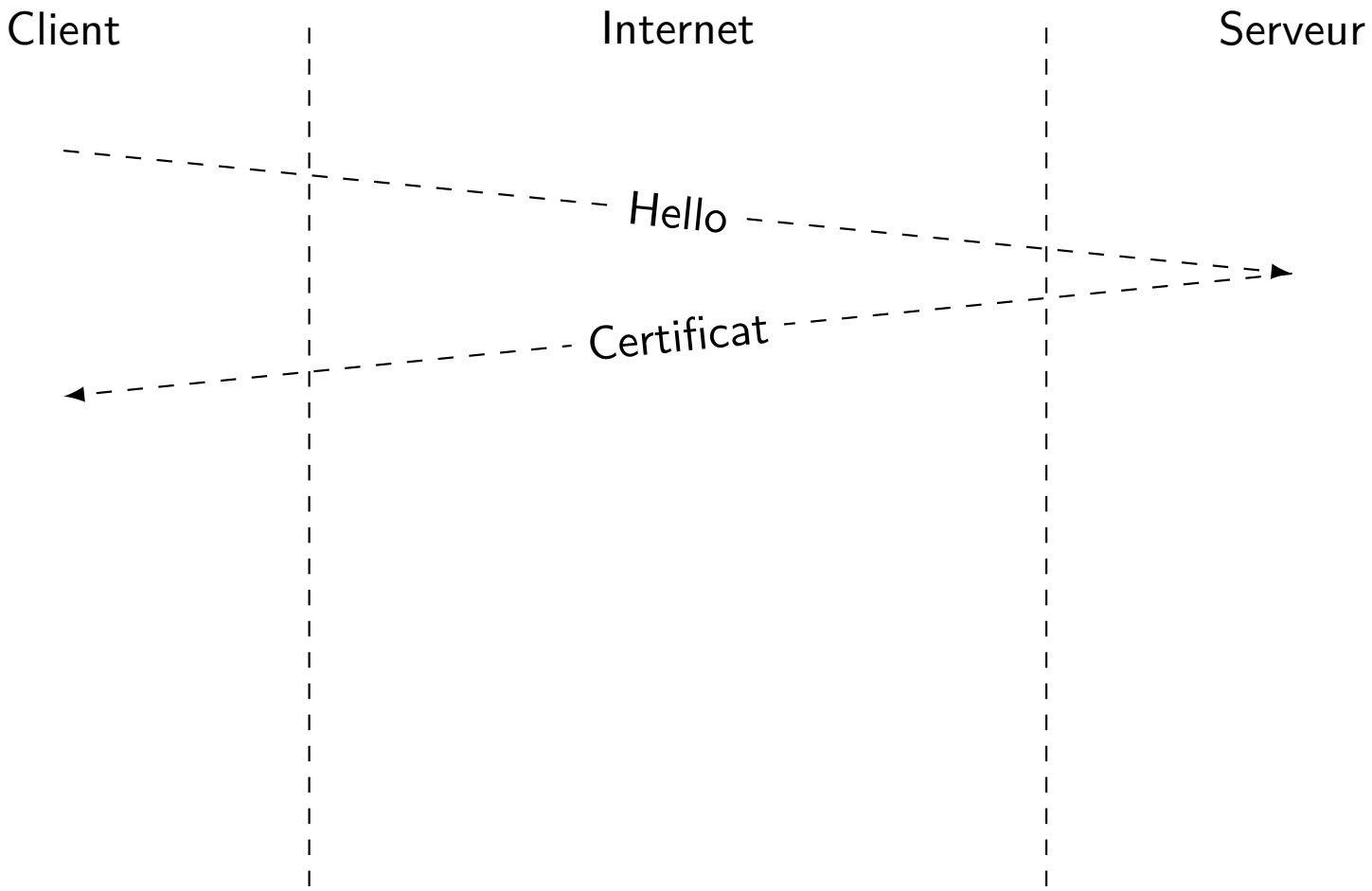


FIGURE 7 – **Certificat :** Le serveur envoie son certificat (sa clé publique signée par la clé privée d'une autorité).

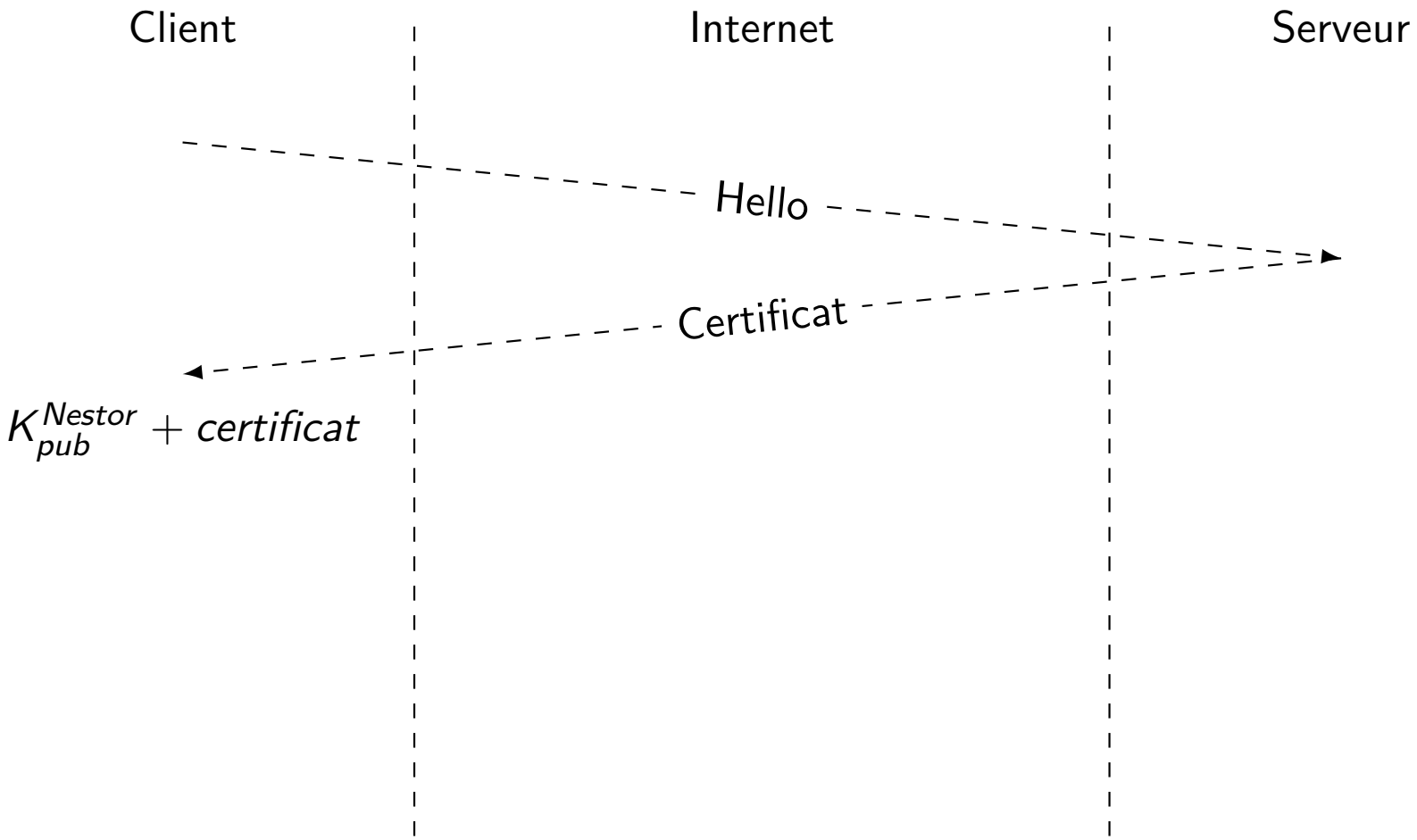


FIGURE 8 – **Authentification :** Le client utilise la clé publique de l'autorité pour déchiffrer le certificat et compare le résultat avec la clé publique du site.

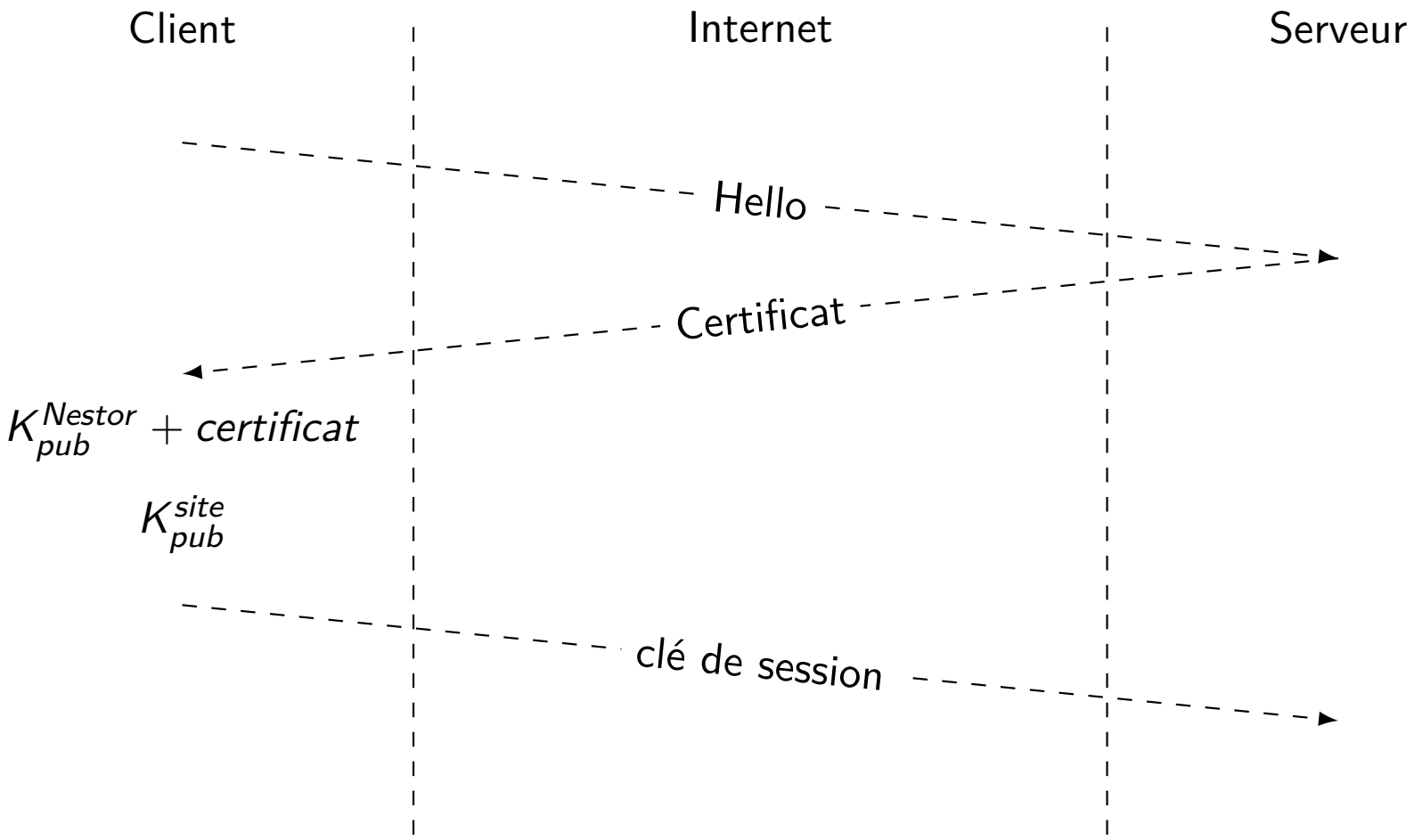


FIGURE 9 – **Clé de session :** Le client et le serveur se mettent d'accord sur un protocole d'échange (symétrique, Diffie-Hellman) : le client peut communiquer sa clé de manière sécurisée grâce à la clé publique authentifiée du site.

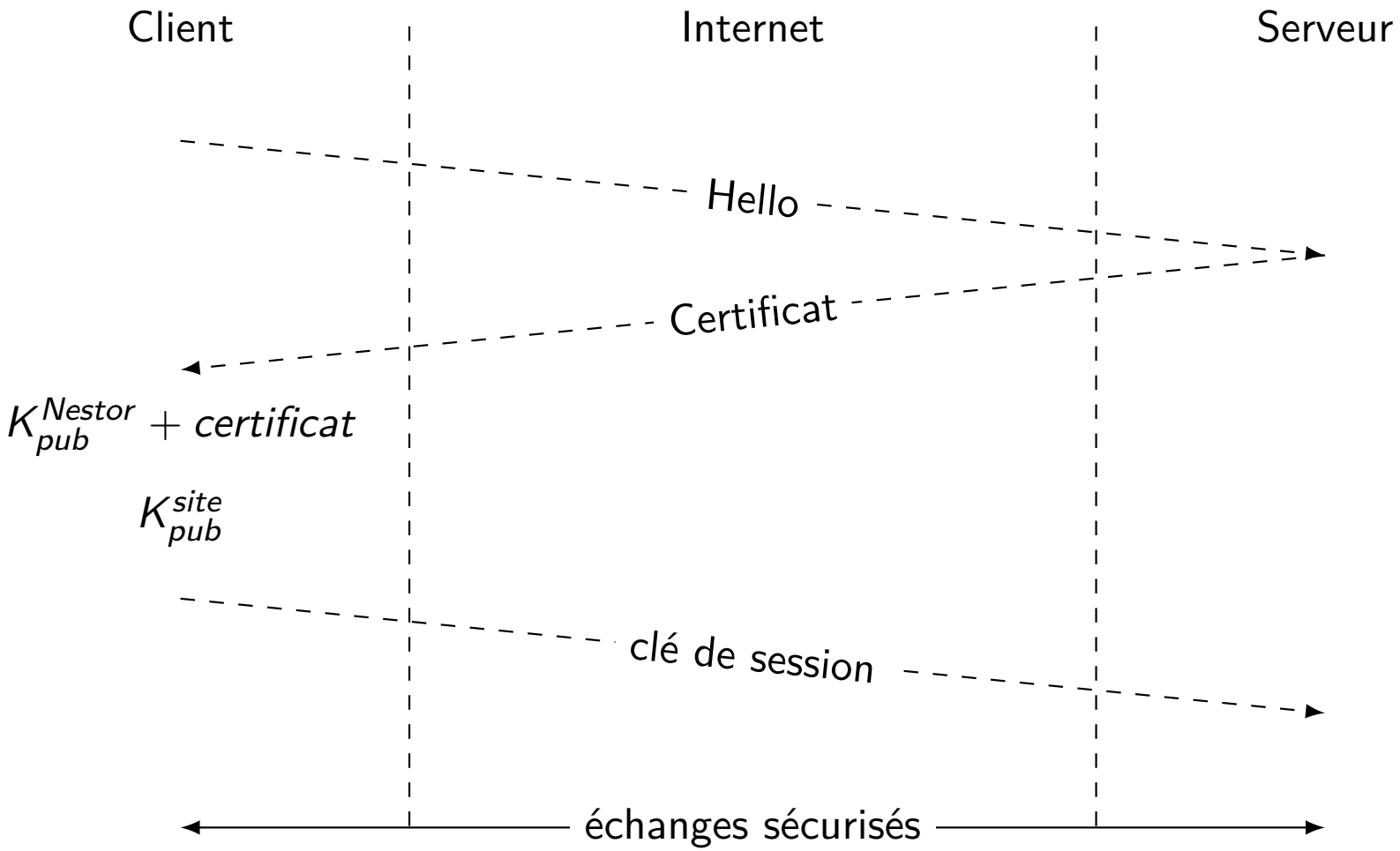


FIGURE 10 – **Échanges :** Le client et le serveur échangent de manière sécurisée.

Sommaire

Informations techniques

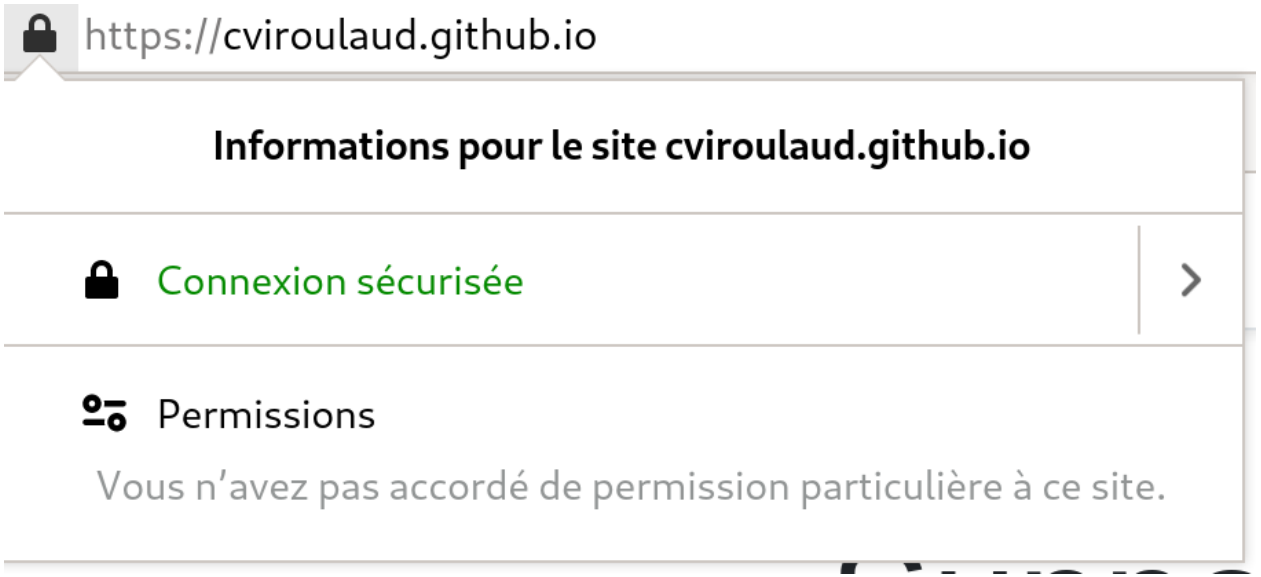


FIGURE 11 – Le cadenas atteste des échanges sécurisés

Détails techniques

Connexion chiffrée (clés TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384, 256 bits, TLS 1.2)

La page actuellement affichée a été chiffrée avant d'avoir été envoyée sur Internet.

TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384, 256bits, TLS 1.2

- Protocole **TLS** (Transport Layer Security)
- Algorithme d'échange de clés : **ECDHE** (Elliptic Curve Diffie-Hellman Ephemeral)
- Algorithme d'authentification : **RSA**
- Algorithme de chiffrement (symétrique) par bloc : **AES 256bits** (Advanced Encryption Standard) en mode GCM (Galois/Counter Mode)
- Algorithme de code d'authentification de message : **SHA384** (création de la somme de contrôle de la clé)