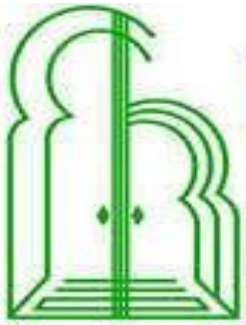


Université Mohammed V de Rabat

Ecole Supérieure de Technologie de Salé

ALGORITHMES DE CRYPTOGRAPHIE

Pr R.ALAOUI

Plan

- Introduction

- Texte en clair au texte chiffré (crypté)

- Algorithmes de substitution ou chiffrement simple
  Chiffrement de César
  Chiffrement de Vigenère

- Algorithmes symétriques ou à clef privée
Chiffrement par bloc :DES , AES...
Chiffrement par flot (ou en continu) :RC4, chacha20
Chiffrement hybride

- Algorithmes asymétriques ou à clef publique: RSA

- Protocole Diffie-Hellman pour l'echange de clé symétrique

- PKI et SSL/TLS: Authentification via certificats numériques

Introduction

L'histoire de la cryptographie est longue et remonte à l'Égypte antique en 1900 avant J.-C.

La cryptographie a pour but de **garantir la sécurité des communications** en **présence d'adversaires** .

La cryptographie est utilisée pour **protéger :**
la confidentialité,
l'intégrité
l'authenticité.

La cryptographie peut être définie comme **la pratique et l'étude de techniques** de **communication sécurisée** et de protection des données là où l'on s'attend à la présence d'adversaires et de tiers.

Les adversaires ne doivent pas pouvoir **divulguer ou modifier** le **contenu des messages**.

Applications de la cryptographie :

- Lorsque vous vous **connectez aux plateformes** (sites web…) , vos **informations d'identification** sont **cryptées** et envoyées au serveur afin que **personne ne puisse** les **récupérer en espionnant votre connexion**.

- Lorsque vous vous **connectez via SSH** , votre **Le client SSH** et **le serveur** établissent un **tunnel crypté** afin que **personne ne puisse espionner votre session**.

- Lorsque **vous effectuez** des **opérations bancaires en ligne**, votre **navigateur vérifie** le **certificat du serveur distant** pour **confirmer** que vous **communiquez avec le serveur de votre banque** et **non avec celui d'un attaquant**.

- Lorsque vous **téléchargez un fichier**, **comment vérifiez-vous** qu'il a été **téléchargé correctement** ? **La cryptographie** fournit une **solution via** des **fonctions de hachage** pour **confirmer que votre fichier** est **identique à l'original**.

Prenons le **cas d'une entreprise** souhaitant **gérer les informations** de **carte de crédit** et **traiter les transactions associées**.

Lors du traitement des cartes de crédit, **l'entreprise doit suivre** et **appliquer la norme de sécurité des données de l'industrie** des **cartes de paiement (PCI DSS)**.

Dans ce cas, la **norme PCI DSS garantit** un **niveau de sécurité minimum** pour **stocker, traiter et transmettre les données** liées aux **crédits de carte**.

Selon la **norme PCI DSS** pour les grandes organisations , vous apprendrez que les **données doivent être cryptées** à la fois **pendant leur stockage** (au repos) et **pendant leur transmission** (en mouvement).

Texte en clair au texte chiffré (crypté)

Le texte en clair est la **donnée lisible** ; il peut s'agir de n'importe quoi, d'un simple « bonjour », d'une photo de chat, d'informations de carte de crédit ou de dossiers médicaux.

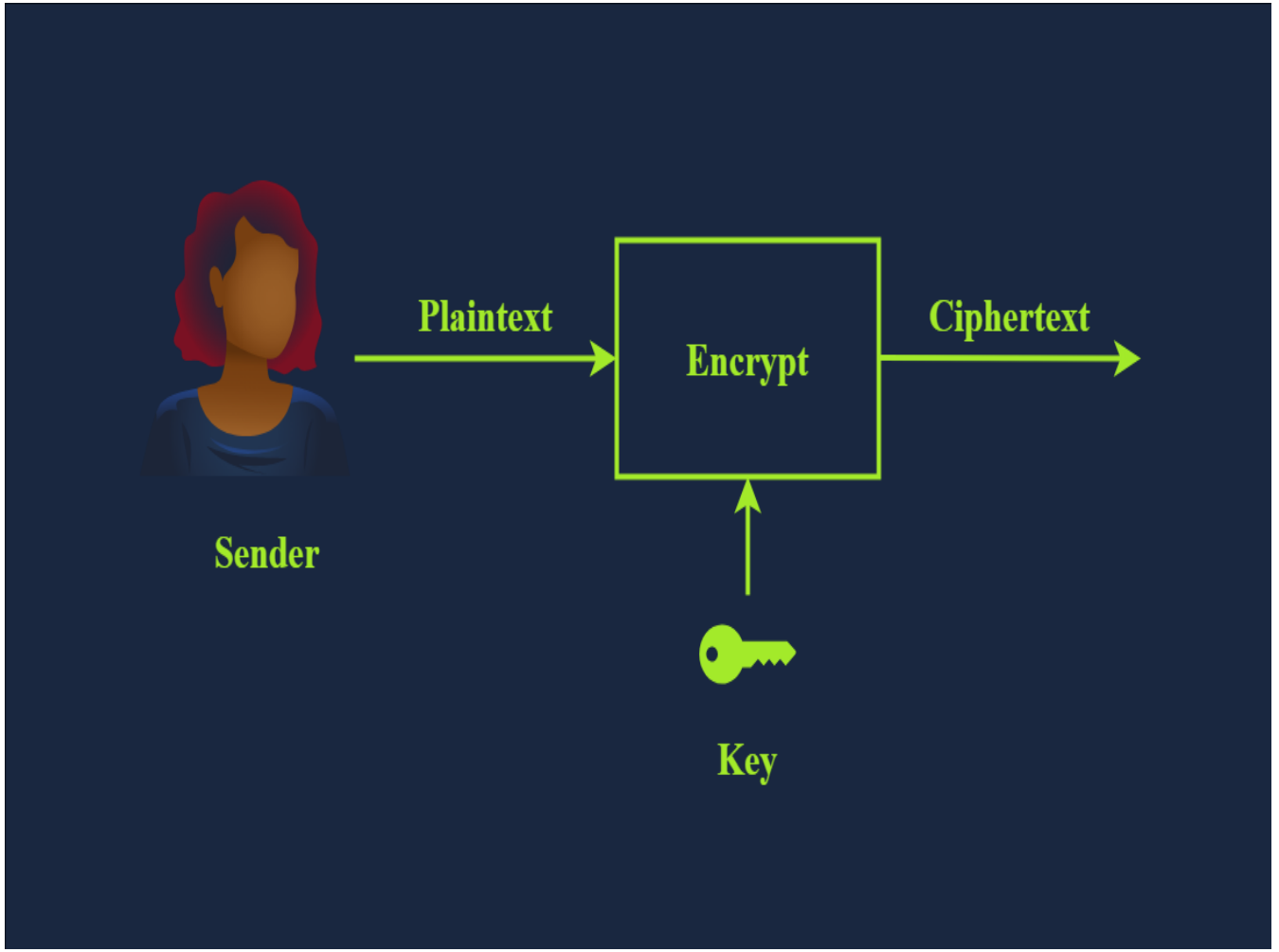
Du **point de vue de la cryptographie**, ce sont tous des messages en « **texte en clair** » qui **attendent d'être chiffrés**.

Le texte en clair est **transmis** à la **fonction de chiffrement** avec une **clé appropriée** ;

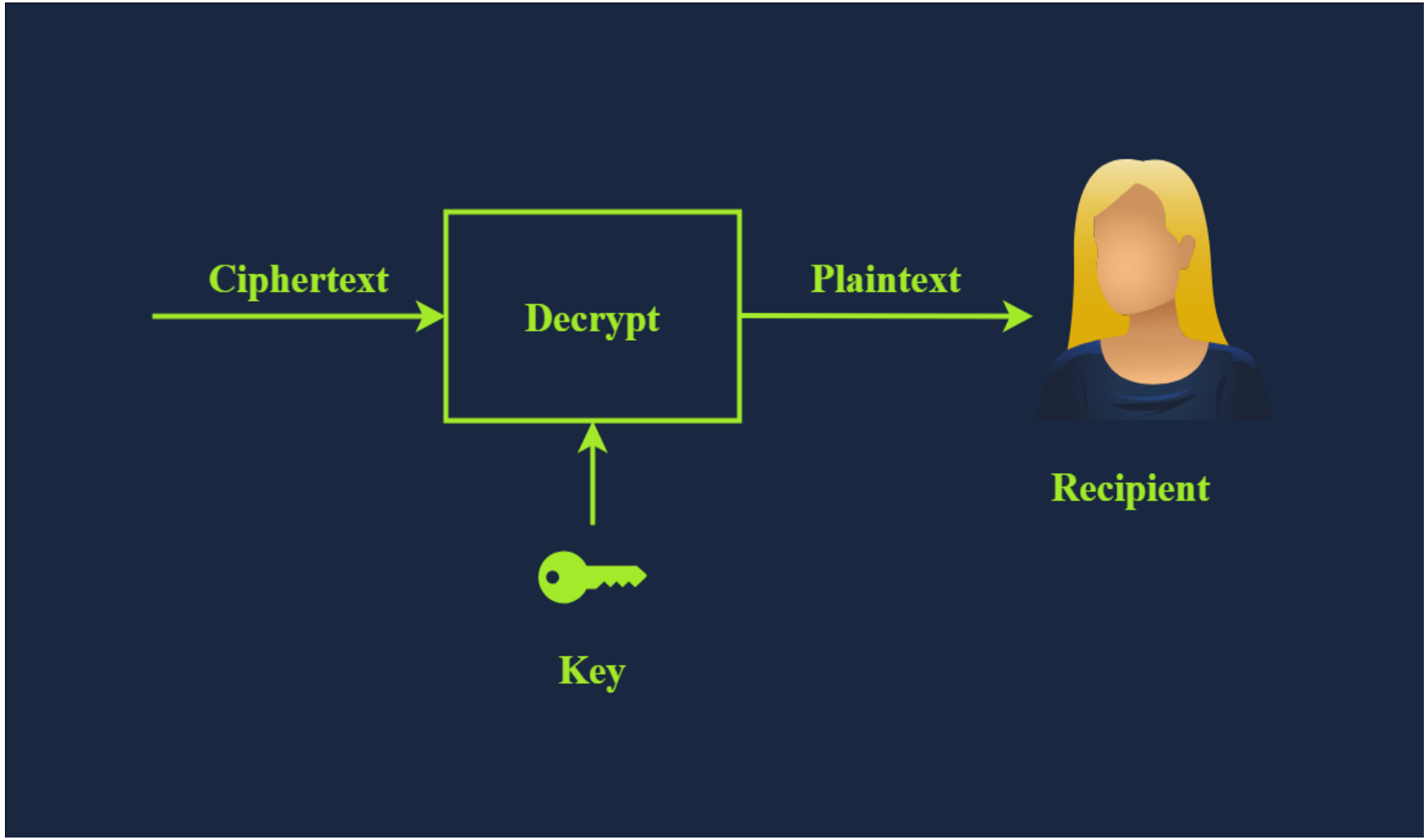
la fonction de chiffrement renvoie un texte chiffré .

La fonction de chiffrement fait partie du chiffrement ;

un chiffrement est un **algorithme permettant de convertir un texte en clair** en **texte chiffré** et vice versa.



Pour **récupérer le texte en clair**, nous devons **transmettre le texte chiffré** avec la **clé appropriée** via la **fonction de décryptage**, ce qui nous donnerait le **texte en clair d'origine** .

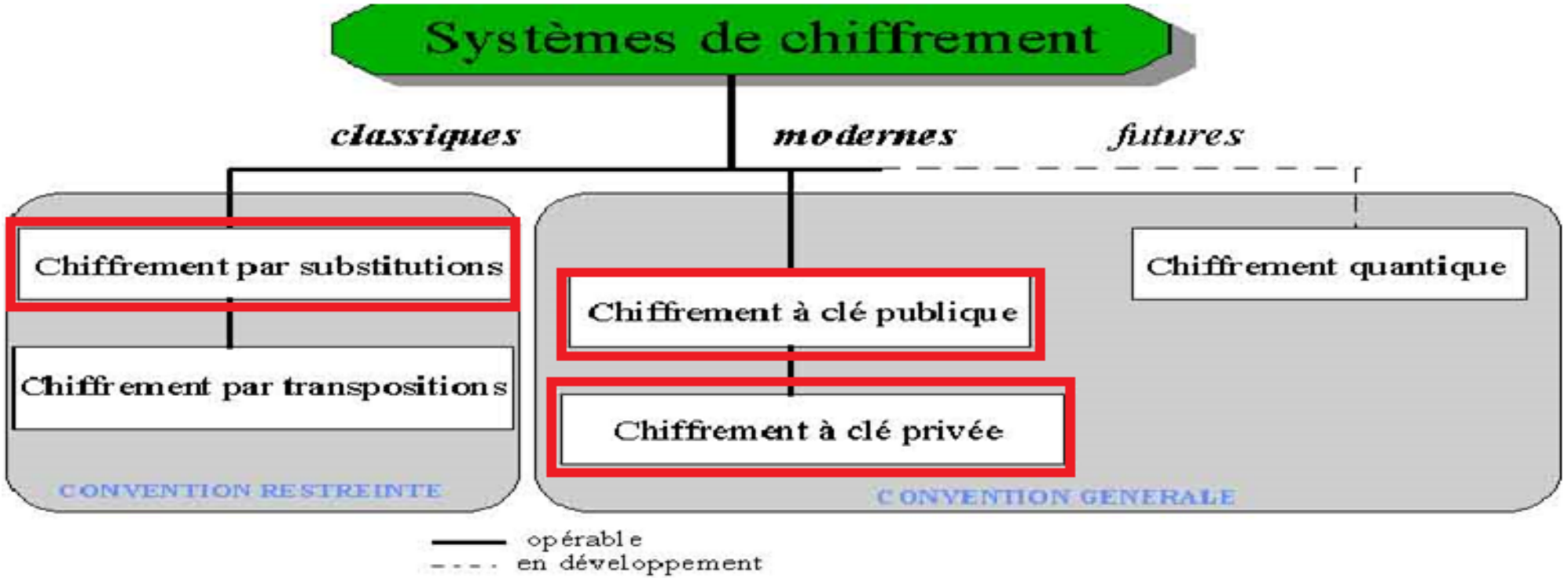


Il existe trois types d'algorithmes

1. **Algorithmes de substitution ou chiffrement simple**

2. **Algorithmes symétriques ou à clef privée**

3. **Algorithmes asymétriques ou à clef publique**

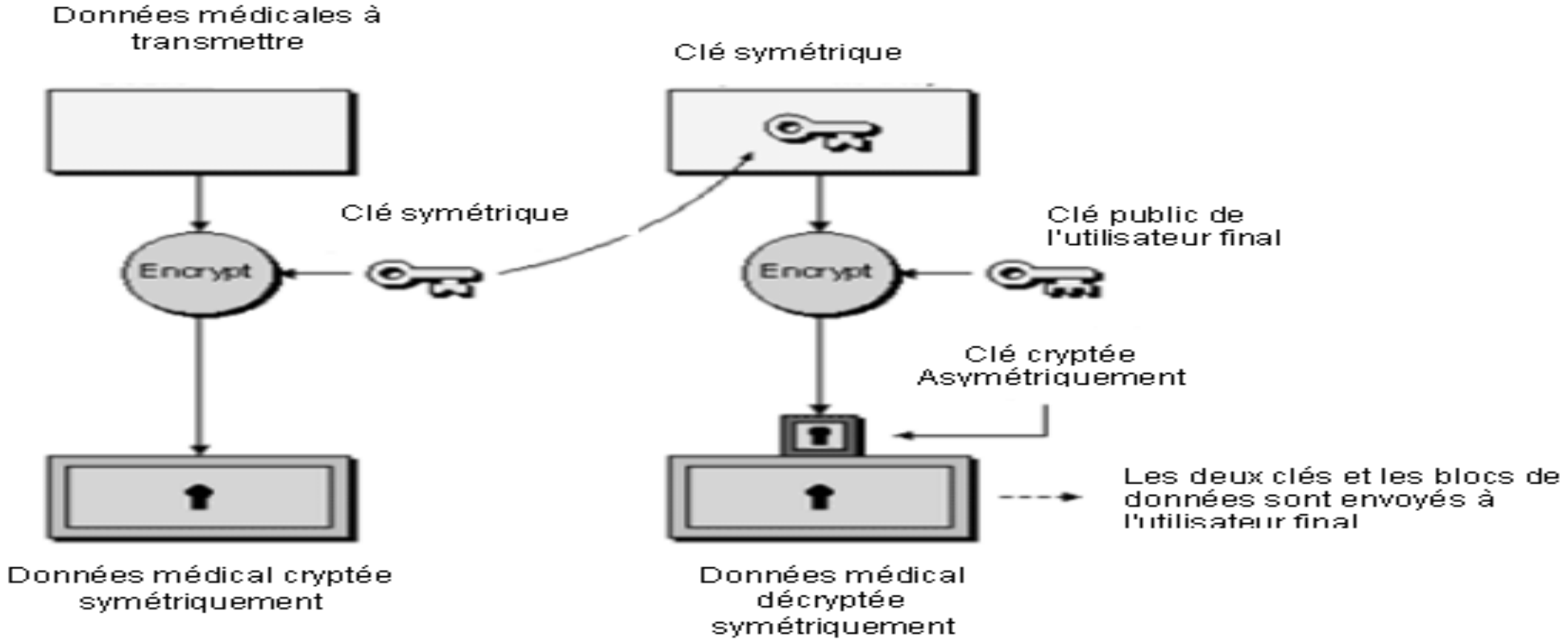


Le chiffrement symétrique est **beaucoup plus rapide** que le chiffrement asymétrique **mais a l'inconvénient**

de nécessiter le **partage** au préalable d'une **clé secrète**.

En pratique, on utilise d'abord un **chiffrement asymétrique** pour échanger la clé secrète et ensuite

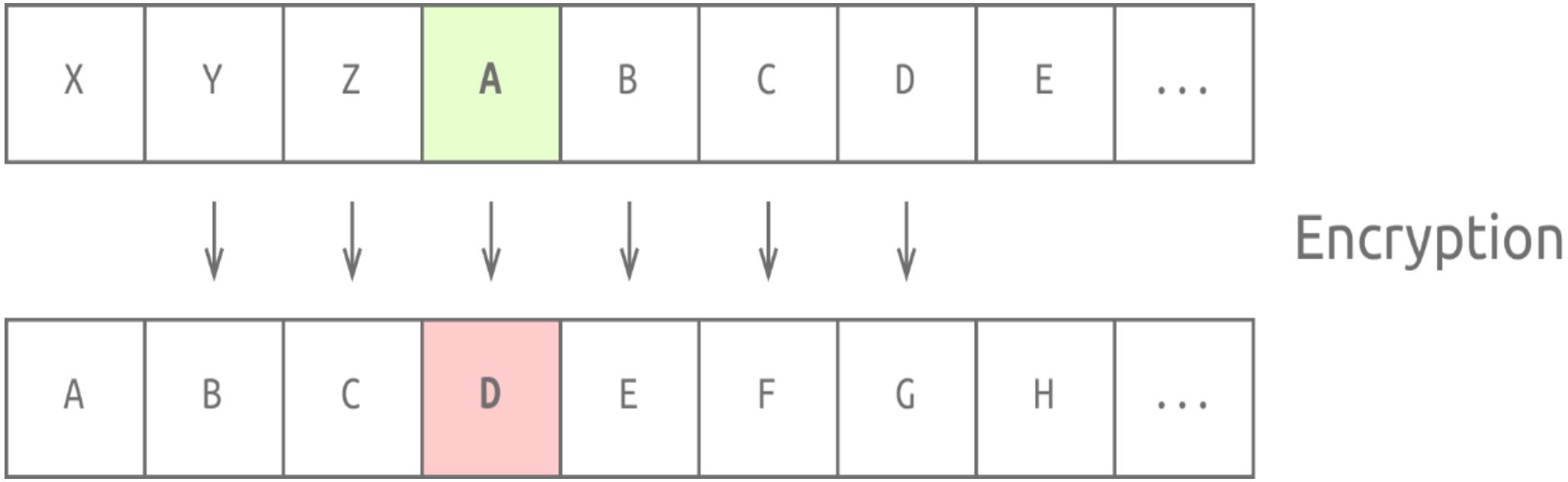
un **chiffrement symétrique** pour l'échange des données.



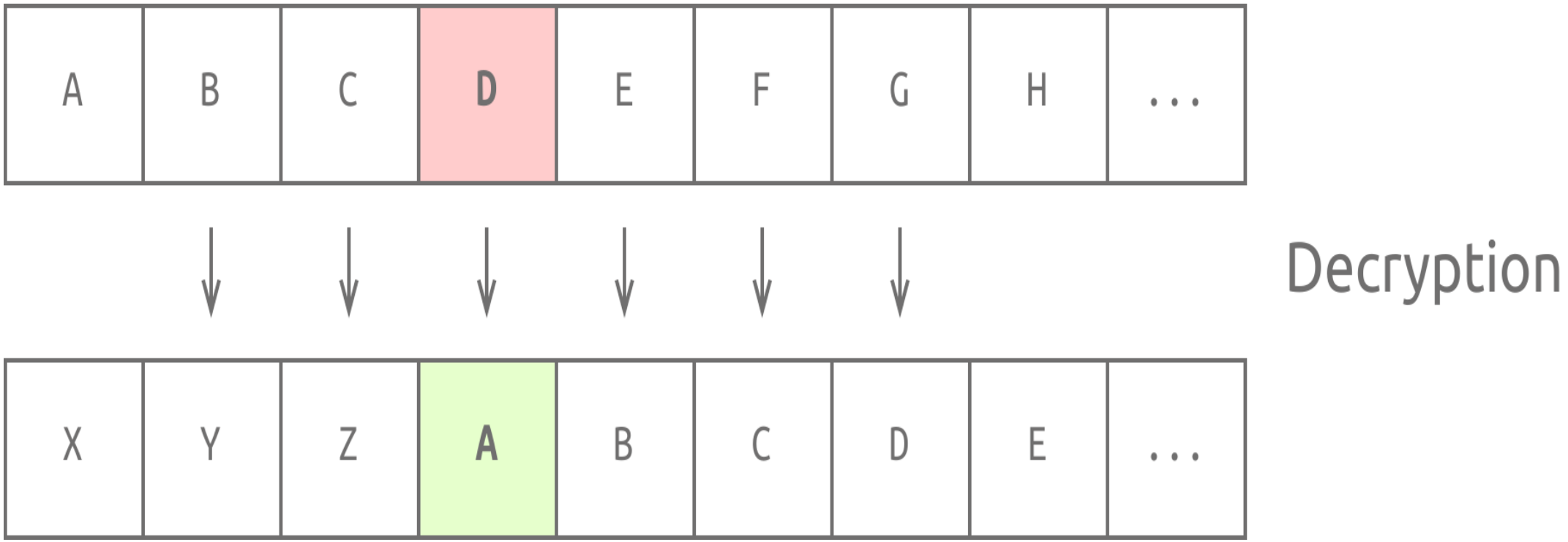
Chiffrement de César

L'un des chiffrements les **plus simples** est le **chiffrement de César**, utilisé il y a plus de 2000 ans.

Le chiffrement de César **décale la lettre** d'un **nombre fixe** de positions vers la gauche ou vers la droite.
Exemple: Cas d'un **décalage de 3 positions** vers la **droite** pour crypter



Le destinataire doit savoir que le texte a été décalé de 3 vers la droite pour récupérer le message d'origine.



En utilisant la même clé pour crypter « TRY HACK ME », nous obtenons « WUB KDFN PH ».

Le chiffrement de César peut utiliser une **clé** comprise **entre 1 et 25**.
Avec une clé de 1, chaque lettre est décalée d'une position,
où A devient B et
Z devient A.

Avec une clé de 25, chaque lettre est décalée de 25 positions, où A devient Z et B devient A.

Une clé de 0 signifie qu'il n'y a aucun changement ;

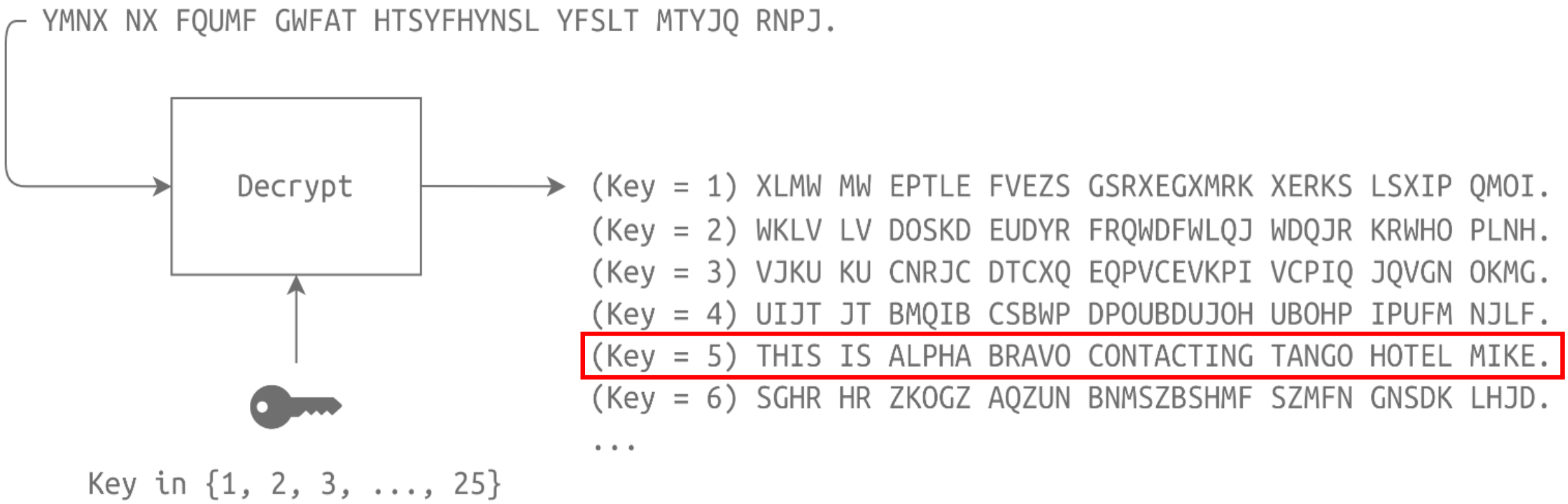
de plus, une **clé de 26 n'entraînera aucun changement** car elle entraînerait une **rotation complète**.

Conclusion:
le **chiffrement de César** a un **espace de clés de 25** ;
il existe **25 clés différentes** parmi lesquelles **l'utilisateur peut choisir**.

Considérez le cas où vous avez intercepté un message chiffré à l'aide du chiffrement César :
« YMNX NX FQUMF GWFAT HTSYFHYNSL YFSLT MTYJQ RNPJ ».

On nous demande de le déchiffrer sans connaître la clé.
Nous pouvons tenter cela en utilisant la force brute, c'est-à-dire que nous pouvons essayer toutes les clés possibles et voir laquelle est la plus logique.

Dans la figure suivante, nous avons remarqué que la clé 5 est la plus logique :
« C'EST ALPHA BRAVO CONTACTANT TANGO HOTEL MIKE ».



Algorithmes

Les ordinateurs ont révolutionné la cryptographie et surtout le décryptage d'un message intercepté. Nous montrons ici, à l'aide du langage `Python` comment programmer et attaquer le chiffrement de César. Tout d'abord la fonction de chiffrement se programme en une seule ligne :

Code 1 (*cesar.py (1)*).

```
def cesar_chiffre_nb(x,k):
    return (x+k)%26
```

Ici `x` est un nombre de $\{0, 1, \ldots, 25\}$ et `k` est le décalage. `(x+k)%26` a pour valeur le reste modulo 26 de la somme `(x+k)`. Pour le décryptage, c'est aussi simple :

Code 2 (*cesar.py (2)*).

```
def cesar_dechiffre_nb(x,k):
    return (x-k)%26
```

Le chiffrement de César est considéré comme un chiffrement par substitution car chaque lettre de l'alphabet est remplacée par une autre.

Un autre type de chiffrement est appelé chiffrement par transposition , qui crypte le message en changeant l'ordre des lettres.

Chiffrement de Vigenère

C’est un chiffrement par **substitution polyalphabétique** classique **qui utilise une clé** pour **déterminer quel alphabet de substitution** est **appliqué à chaque lettre**.

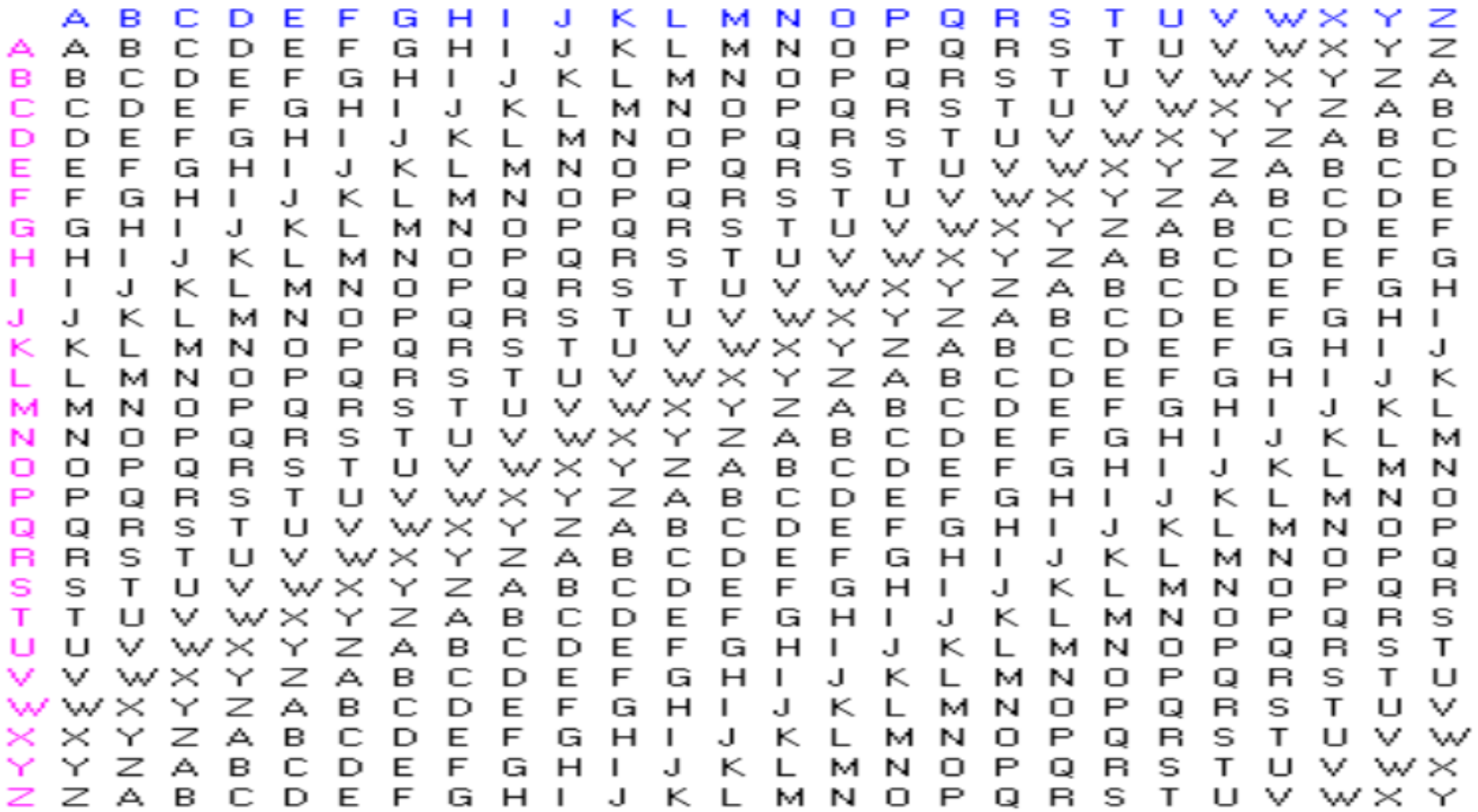
Principe du chiffrement de Vigenère

1.Message clair (plaintext) : Le texte que l'on veut chiffrer.

2.Clé (key) : Une chaîne de lettres (souvent un mot ou une phrase) **répétée ou étendue à la longueur du message clair**.

3. Tableau de Vigenère (ou décalage alphabétique) :

On utilise un **tableau contenant les 26 alphabets décalés**, où chaque ligne correspond à une substitution différente.

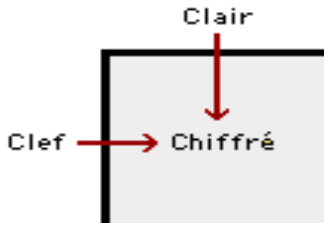


Application

On veut coder le texte "**CRYPTOGRAPHIE DE VIGENERE**" avec la clé "**MATHWEB**".
On commence par écrire la clef sous le texte à coder :

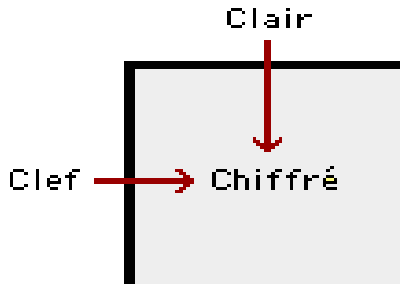
C	R	Y	P	T	O	G	R	A	P	H	I	E		D	E		V	I	G	E	N	E	R	E
M	A	T	H	W	E	B	M	A	T	H	W	E		B	M		A	T	H	W	E	B	M	A

Pour coder la lettre C, la clé est donnée par la lettre M.



On regarde dans le tableau l'intersection de la ligne donnée par le C,
et de la colonne donnée par le M
On trouve O.
Puis on continue.
On trouve : **ORRWPSHDAIOEI EQ VBNARFDE**

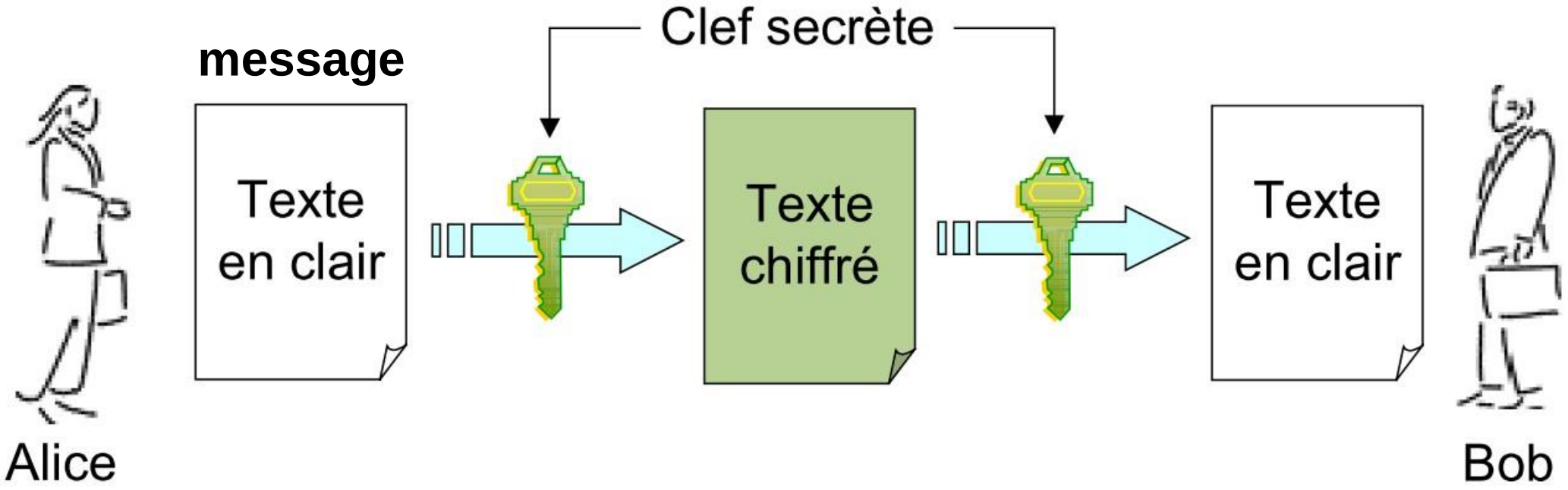
Pour décoder, il suffit de faire la même chose dans l'autre sens:



Chiffrement symétrique (ou *chiffrement à clé privée*)

Définition :

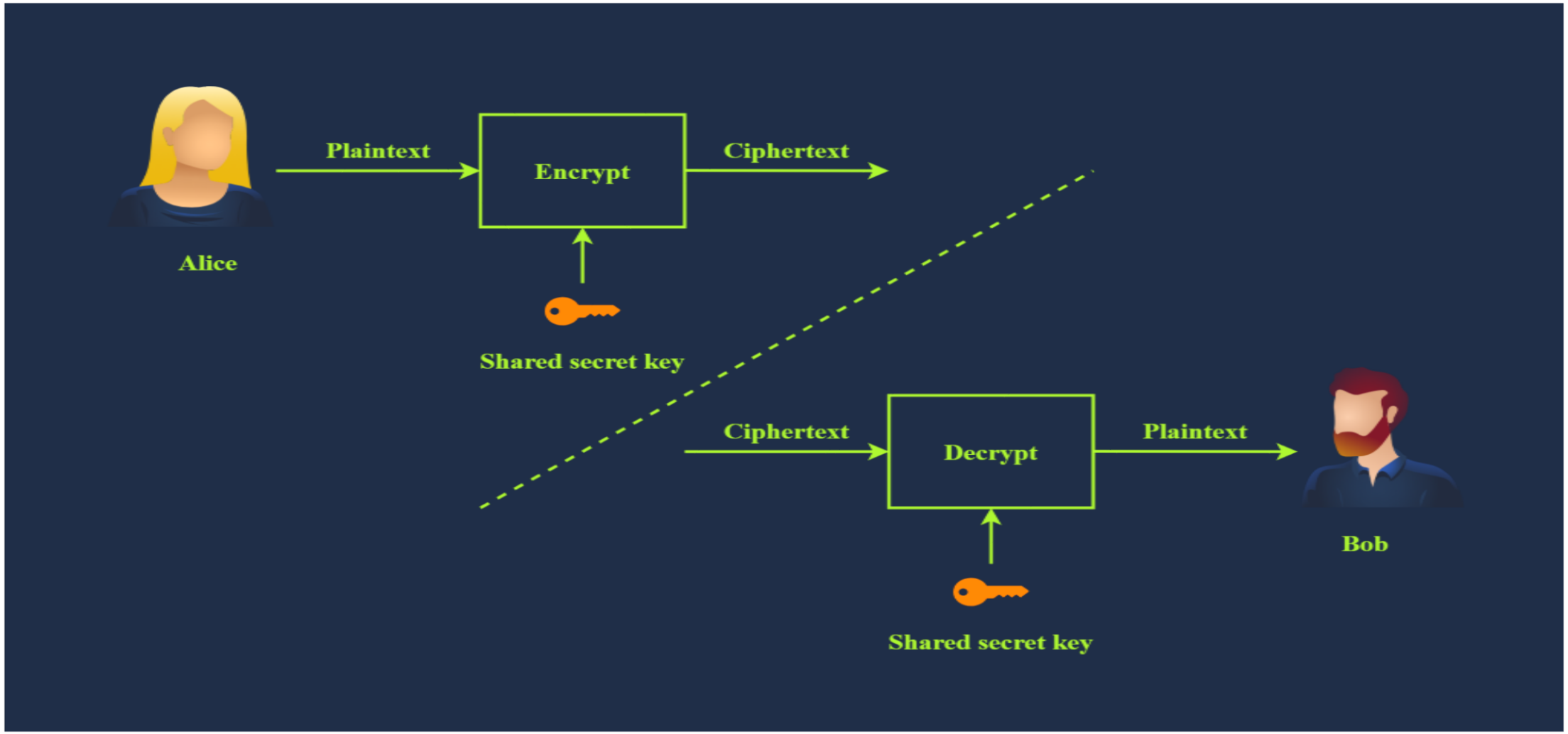
Le chiffrement symétrique consiste à utiliser la **même clé** pour le **chiffrement et le déchiffrement**.



Principe :

Le chiffrement symétrique consiste à appliquer une opération (algorithme) sur les données à chiffrer à l'aide de la clé privée, afin de les rendre inintelligibles.

On donne le message à quelqu'un et on lui fournit la clé privée, pour qu'il puisse déchiffrer le



Types d'algorithmes de chiffrement symétriques :

1.Chiffrement par bloc :

Divise les données en blocs de **taille fixe** **avant de les chiffrer**.

Exemples :

1. **DES (Data Encryption Standard)** : Ancien standard, aujourd'hui considéré comme peu sûr.
2. **AES (Advanced Encryption Standard)** : Actuellement très utilisé, avec des clés de 128, 192, ou 256 bits.
3. **Blowfish** : Rapide et flexible.
4. **Triple DES (3DES)** : Version améliorée de DES.

2. Chiffrement par flot (ou en continu) :

Chiffre les données **bit par bit** ou **caractère par caractère**.

Exemples :

1. **RC4** : Utilisé autrefois dans les protocoles comme WEP, mais vulnérable.
2. **ChaCha20** : Une alternative moderne et sécurisée.

3. Chiffrement hybride (utilisant symétrique avec d'autres approches) :

Combine les algorithmes symétriques avec des méthodes asymétriques pour l'échange sécurisé de clés.

Problèmes :

Un algorithme de chiffrement symétrique utilise la même clé pour le chiffrement et le déchiffrement. Par conséquent, les parties communicantes doivent se mettre d'accord sur une clé secrète avant de pouvoir échanger des messages.

- **Longueur de la clé privée** :

  Selon *Claude Shannon* (années 40)

  pour être totalement sûr, les systèmes à clefs privées doivent utiliser **des clefs d'une longueur au moins égale à celle du message à chiffrer**.

- **Canal sécurisé** :

  Le chiffrement symétrique impose d'avoir un **canal sécurisé pour l'échange de la clé**, ce qui dégrade sérieusement l'intérêt d'un tel système de chiffrement.

Imaginez le cas simple où vous avez **créé** un **document protégé par mot de passe** pour le **partager avec votre collègue**.

Vous pouvez facilement envoyer le **document crypté à votre collègue** par **courrier électronique**,

Mais il est **fort probable** que vous **ne puissiez pas lui envoyer** le **mot de passe par courrier électronique**.

En effet, toute personne ayant accès à sa boîte aux lettres accéderait à la fois au document protégé par mot de passe et à son mot de passe.

Vous devez donc réfléchir à un autre moyen, c'est-à-dire à un canal, pour partager le mot de passe.

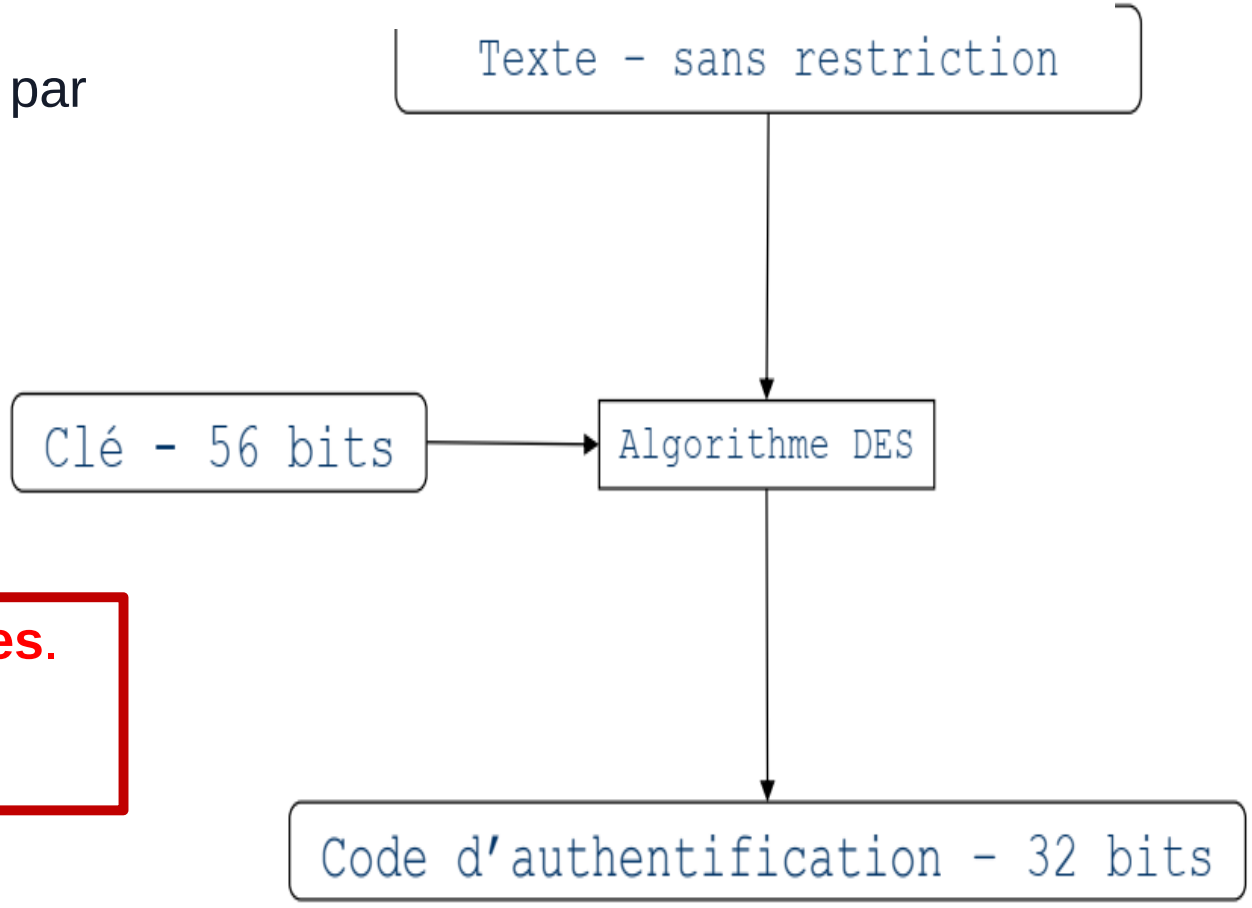
À moins que vous ne pensiez à un canal sécurisé et accessible, une solution consisterait à le rencontrer en personne et à lui communiquer le mot de passe.

DES (Data Encryption Standard)

Le National Institute of Standard and Technology (NIST) a publié la **norme de chiffrement des données DES** en **1977**.

DES est un **algorithme de chiffrement symétrique** qui utilise une **taille de clé de 56 bits**.
En 1997, un défi visant à déchiffrer un message chiffré à l'aide de DES a été résolu.

il a été démontré : **Possible d'utiliser** une recherche par **force brute** pour **trouver la clé et déchiffrer un message chiffré** à l'aide de **DES** .



En 1998, une **clé DES** a été **déchiffrée en 56 heures**.
Ces cas ont montré que **DES ne pouvait plus être considéré comme sûr**.

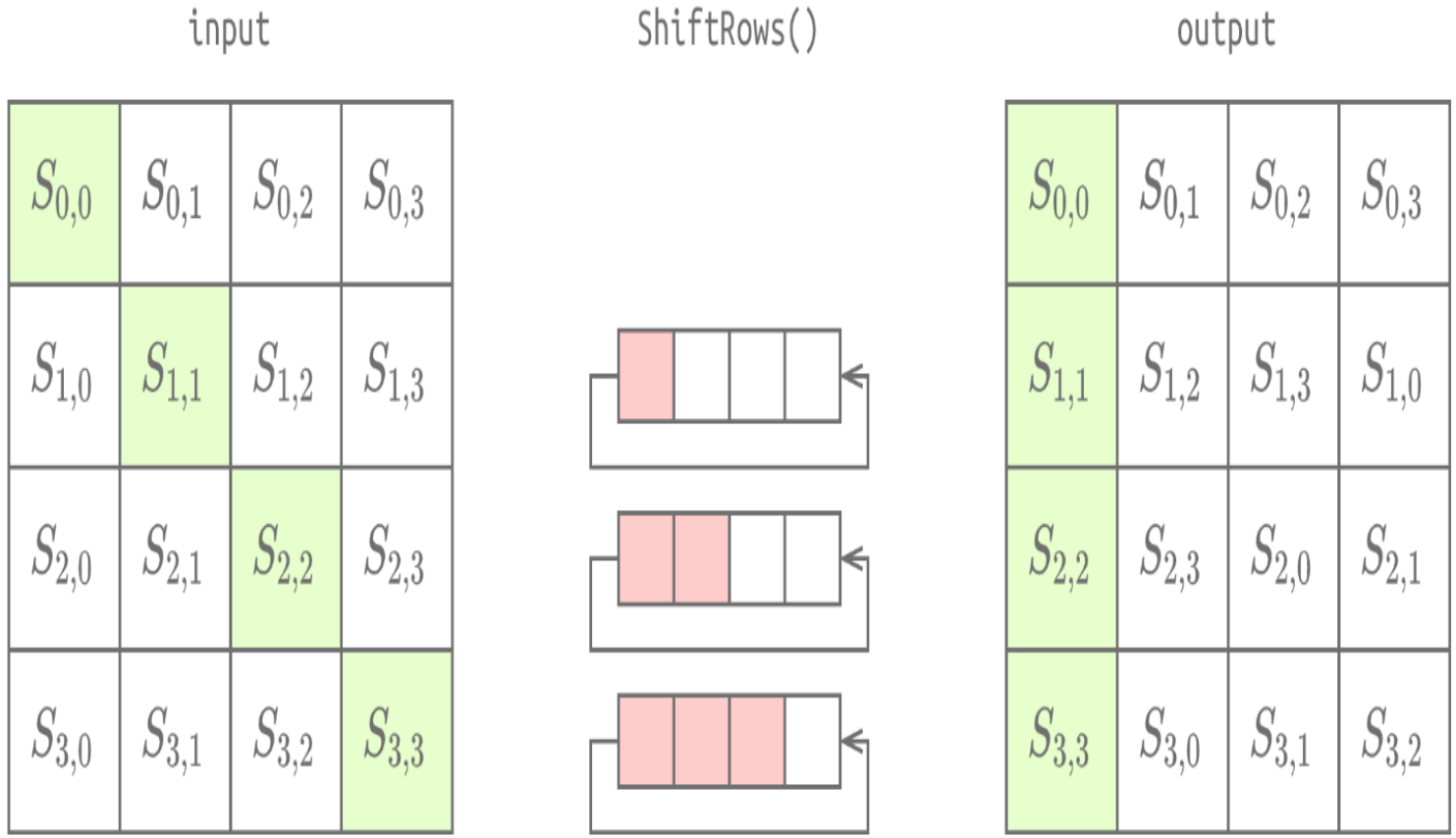
AES (Advanced Encryption Standard)

Le NIST a publié l'Advanced Encryption Standard **(AES) en 2001**.
Comme DES , il s'agit d'un **algorithme de chiffrement symétrique** ;

AES utilise une **taille de clé** de **128, 192 ou 256 bits**, et il est **toujours considéré** comme **sécurisé et utilisé aujourd'hui**.
AES répète plusieurs fois les **quatre transformations** suivantes :

1. SubBytes (état) :

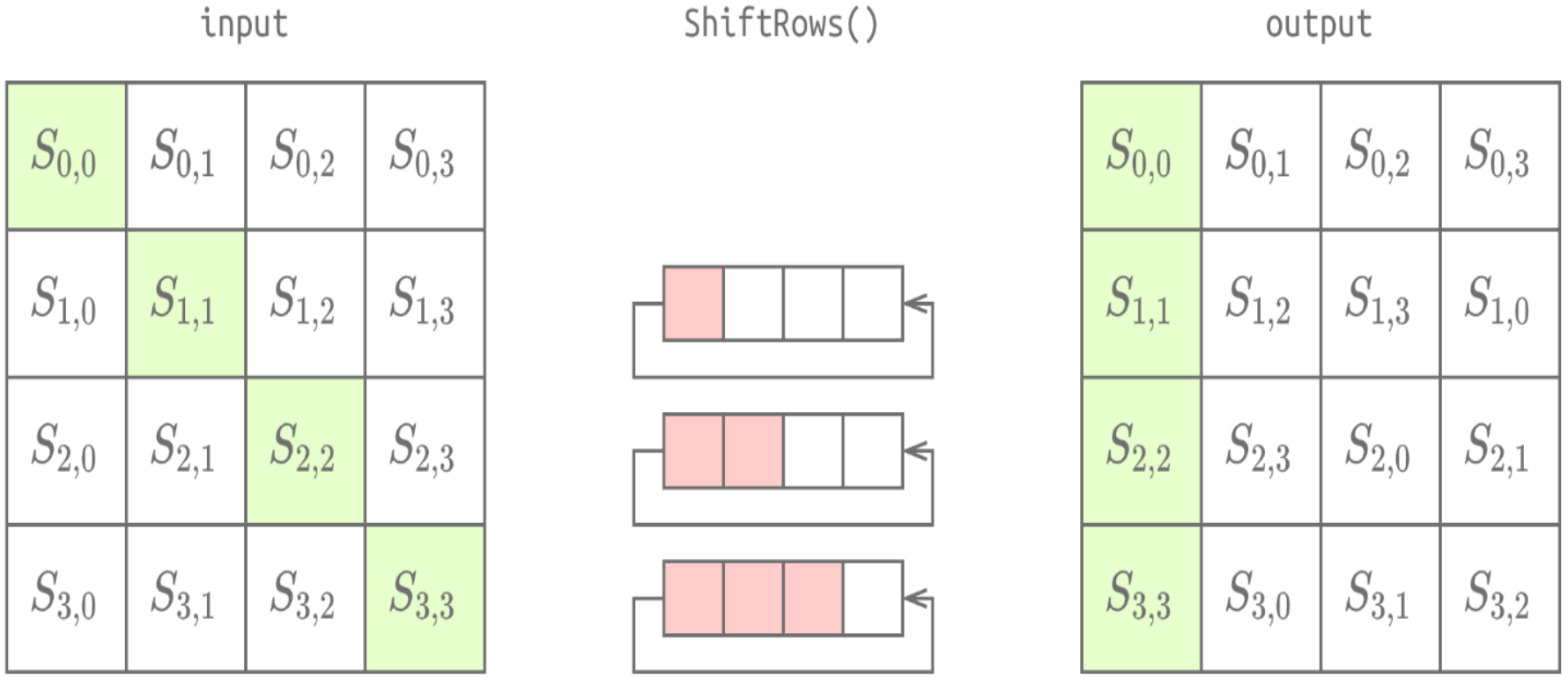
Cette transformation <u>recherche chaque octet</u> dans une **table de substitution donnée (S-box)** et
le remplace par la valeur correspondante.
L' état est de 16 octets, soit 128 bits (16x8=128), enregistrés dans un tableau 4 x 4.



2. ShiftRows (état) :
La deuxième ligne est décalée d'une position ,
la troisième ligne est décalée de deux positions et
la quatrième ligne est décalée de trois positions . montré dans la figure ci-dessous .

3. MixColumns (état) : Chaque colonne est multipliée par une matrice fixe (tableau 4 par 4).

4. AddRoundKey (état) : Une clé ronde est ajoutée à l'état à l'aide de l' opération XOR .



Le nombre total de **cycles de transformation dépend** de la **taille de la clé**.

Ne vous inquiétez pas si vous trouvez cela cryptique, car c'est le cas !

Notre objectif
n'est pas d'apprendre les détails du fonctionnement d'**AES**
ni de l'implémenter en tant que **bibliothèque de programmation** ;

le but est d'apprécier la **différence de complexité entre les algorithmes** de **chiffrement anciens et modernes**.

Si vous êtes curieux de plonger dans les détails, vous pouvez consulter les spécifications AES, y compris le pseudo-code et les exemples dans sa norme publiée, FIPS PUB 197 .

AES , de nombreux autres algorithmes de **chiffrement symétrique** sont **considérés comme sûrs**.

Voici une liste des algorithmes de chiffrement symétrique pris en charge par GPG (GnuPG) 2.37.7, par exemple :

Algorithme	Type	Blocs	Clé (bits)	Remarques
AES	Symétrique	128 bits	128, 192, 256	Standard actuel, très sécurisé et rapide
IDEA	Symétrique	64 bits	128	Sécurisé mais moins utilisé aujourd'hui
3DES	Symétrique	64 bits	Jusqu'à 168	Plus lent que AES, encore utilisé dans les anciens systèmes
CAST5	Symétrique	64 bits	40 à 128	Utilisé dans PGP, moins courant que Blowfish
Blowfish	Symétrique	64 bits	Jusqu'à 448	Rapide, remplacé par Twofish dans certains cas
Twofish	Symétrique	128 bits	Jusqu'à 256	Alternative à AES, très sécurisé
Camellia	Symétrique	128 bits	128, 192, 256	Similaire à AES, utilisé dans certains équipements

Exemple pratique pour AES

Tous les algorithmes déjà mentionnés sont des algorithmes de chiffrement symétrique par bloc.

Un algorithme de chiffrement par bloc convertit l'entrée (texte en clair) en blocs et chiffre chaque bloc.

Un bloc est généralement de 128 bits.

Texte clair : "TANGO HOTEL MIKE"

Le texte est composé de **16 caractères** :

T A N G O (espace) H O T E L (espace) M I K E

Chaque caractère est converti en code ASCII, qui est ensuite représenté en **hexadécimal** (base 16).
Par exemple :

- **T** est **0x54** (en hexadécimal).
- **A** est **0x41**.
- **N** est **0x4E**.
- Et ainsi de suite pour chaque caractère.

Donc, le texte complet se transforme en une suite de valeurs hexadécimales comme ceci :

54 41 4E 47 4F 20 48 4F 54 45 4C 20 4D 49 4B 45

Création du tableau 4x4 (Projection de 54 41 4E 47 4F 20 48 4F 54 45 4C 20 4D 49 4B 45)

Un **bloc de 128 bits** (ou 16 octets) est représenté dans une grille de **4 lignes et 4 colonnes**.
On remplit cette grille **colonne par colonne** avec les valeurs hexadécimales, dans l'ordre donné.

Texte clair représenté en tableau :

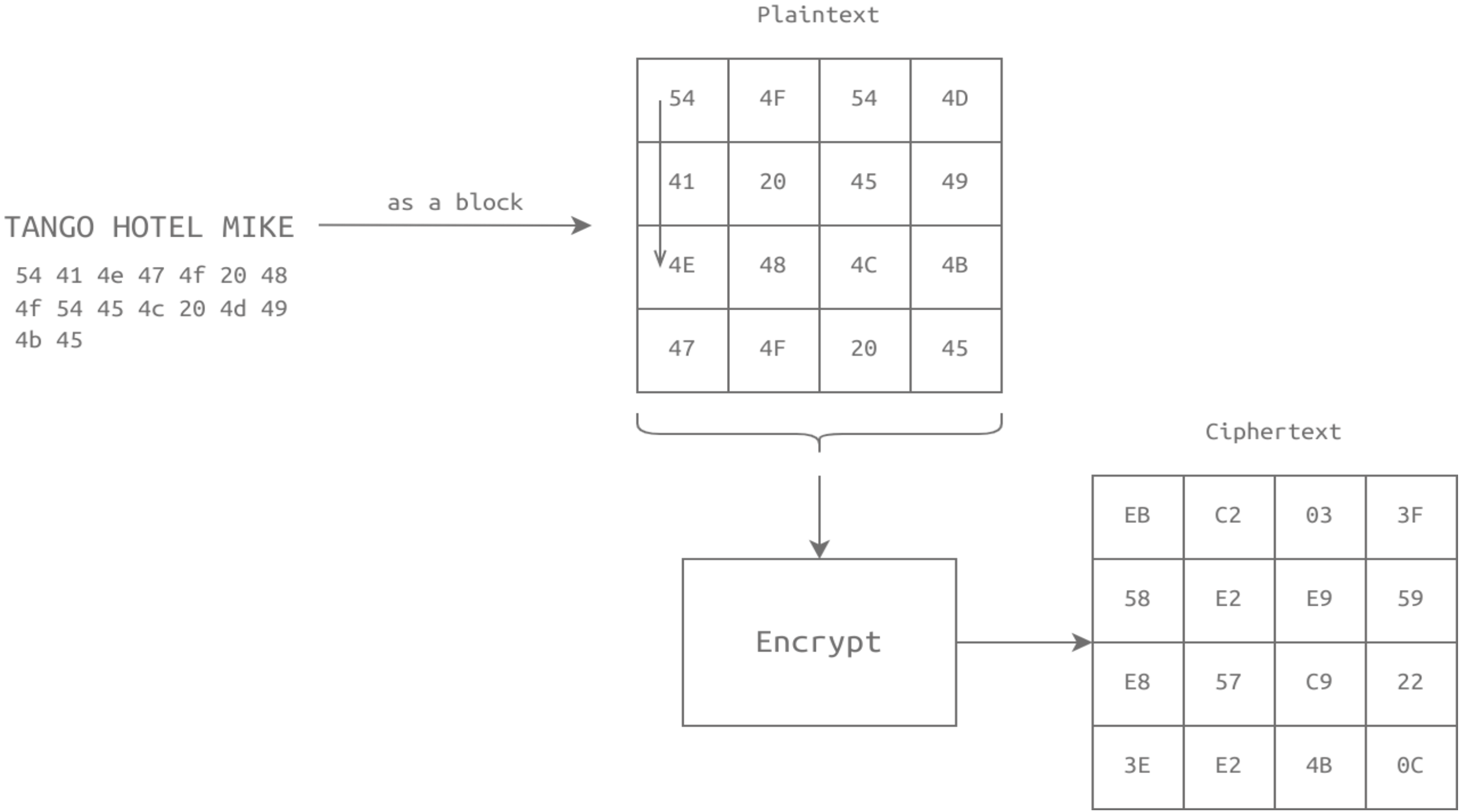
	Colonne 1	Colonne 2	Colonne 3	Colonne 4
Ligne 1	54	4F	54	4D
Ligne 2	41	20	45	49
Ligne 3	4E	48	4C	4B
Ligne 4	47	4F	20	45

Une fois ce tableau 4x4 formé, il est envoyé à **l'algorithme de chiffrement par blocs**, comme **AES (Advanced Encryption Standard)**.
Cet algorithme :
1.Prend **ce bloc de 128 bits** comme une unité.
2.Applique des transformations complexes (substitution, permutation, mélange des colonnes, etc.) en utilisant une clé secrète.

Résultat :

- Le tableau des données chiffrées (texte chiffré) est produit.
- Par exemple, après chiffrement avec une clé donnée, on obtient :

	Colonne 1	Colonne 2	Colonne 3	Colonne 4
Ligne 1	EB	C2	03	3F
Ligne 2	S8	E2	E9	59
Ligne 3	E8	57	C9	22
Ligne 4	3E	E2	4B	0C


TANGO HOTEL MIKE
54 41 4e 47 4f 20 48
4f 54 45 4c 20 4d 49
4b 45
as a block
Plaintext
54 4F 54 4D
41 20 45 49
4E 48 4C 4B
47 4F 20 45
Encrypt
Ciphertext
EB C2 03 3F
58 E2 E9 59
E8 57 C9 22
3E E2 4B 0C

Programmes pour le chiffrement symétrique

Deux d'entre eux, est largement utilisés :

. GNU Privacy Guard

. OpenSSL Project

Voir Lab Crypto 1 - -----Introduction to Cryptography

E:\lecteurDD\formations\formation office echange2024\securite\cours2025\hackme tutorials\Cryptography\Introduction to Cryptography

Le chiffrement par flux

Le **chiffrement par flux** (ou **chiffrement par flot**) est une **technique de cryptographie symétrique** où les données sont chiffrées **bit par bit** ou **caractère par caractère**, plutôt que par blocs de taille fixe (comme dans le chiffrement par bloc).

Fonctionnement du chiffrement par flux :

1. Clé initiale et générateur pseudo-aléatoire :
Le chiffrement par flux utilise une **clé secrète** combinée à un **générateur pseudo-aléatoire** (PRG) pour produire un flux de bits appelé **flux de clé**.
2. Combinaison avec les données en clair :
Chaque bit ou caractère du **flux de clé** est **combiné** (généralement par une opération XOR) avec le texte en clair pour produire le texte chiffré.
3. Déchiffrement :
Le déchiffrement est réalisé en appliquant le même flux de clé au texte chiffré (l'opération XOR étant réversible).

Exemples d'algorithmes de chiffrement par flux :
•**RC4** : L'un des algorithmes par flux les plus connus, utilisé dans des protocoles comme WEP et SSL (aujourd'hui peu sûr à cause de vulnérabilités).
•**ChaCha20** : Un algorithme moderne, rapide et sécurisé, utilisé dans des protocoles comme TLS.

Applications de chiffrement par flux

1. Communications en temps réel 🗣️

- **Voix sur IP (VoIP)** : Protocole de **communication audio** comme **Skype ou WhatsApp**.
    - Avantage : Le chiffrement par flux permet de protéger les données tout en maintenant une transmission fluide et rapide.
- **Diffusions en direct (live streaming)** : Protéger les flux **vidéo/audio** en **temps réel**.

2. Télécommunications mobiles 📱

- **SMS sécurisés** : Certains protocoles de messagerie mobile utilisent le chiffrement par flux pour transmettre des messages textuels de manière sécurisée.
- **Réseaux sans fil** : Historiquement, le chiffrement WEP (Wi-Fi) utilisait l'algorithme RC4 (un chiffrement par flux).

3. Dispositifs embarqués et IoT 🚘

- Les objets connectés (IoT) et systèmes embarqués utilisent souvent des algorithmes par flux pour sécuriser des communications dans des environnements où les ressources sont limitées (faible puissance de calcul, mémoire réduite).
    - Exemples : Domotique, capteurs intelligents, montres connectées.

4. Systèmes militaires et de défense 🛡️

- Les transmissions radio sécurisées ou les communications cryptées en temps réel utilisent le chiffrement par flux pour garantir la confidentialité et la rapidité.

TANGO HOTEL MIKE

54 41 4e 47 4f 20 48
4f 54 45 4c 20 4d 49
4b 45

as a stream