Introduction à la sécurité
Cours 1 – Cryptologie historique

Julien Lavauzelle

Université Paris 8

Licence 3 Informatique et Vidéoludisme

11/09/2023

Ces slides sont en partie inspirées de celles de Pablo Rauzy, concepteur initial du cours.

Voir sa page web : `pablo.rauzy.name/teaching/is`

Plan

Présentation

La **cryptologie** :

- étymologie : **science** du **secret**
- une discipline très **ancienne** : tablettes d'argile au 16ème siècle AEC (avant l'ère commune)
- historiquement, principalement réservée à des enjeux militaires
- à partir des années 60 : devient une discipline scientifique, universitaire

La **cryptologie** :

- étymologie : **science** du **secret**
- une discipline très **ancienne** : tablettes d'argile au 16ème siècle AEC (avant l'ère commune)
- historiquement, principalement réservée à des enjeux militaires
- à partir des années 60 : devient une discipline scientifique, universitaire

Enjeux (exemples) :

- **confidentialité**
- **authentification**
- **non-répudiation**
- **intégrité**
- **anonymat**, **preuve de connaissance**, **retrait confidentiel d'information**, ...

Présentation

La **cryptologie** :

- étymologie : **science** du **secret**
- une discipline très **ancienne** : tablettes d'argile au 16ème siècle AEC (avant l'ère commune)
- historiquement, principalement réservée à des enjeux militaires
- à partir des années 60 : devient une discipline scientifique, universitaire

Enjeux (exemples) :

- **confidentialité**
- **authentification**
- **non-répudiation**
- **intégrité**
- **anonymat**, **preuve de connaissance**, **retrait confidentiel d'information**, ...

Des **primitives cryptographiques** permettenr de satisfaire ces besoin. Exemples :

- le **chiffrement** pour la confidentialité
- la **signature** pour la l'intégrité, la non-répudiation, en partie l'authentification

Deux approches principales :

Présentation

Deux approches principales :

- la crypto**graphie** : **construction** de primitives

Deux approches principales :

- la crypto**graphie** : **construction** de primitives
- la crypt**analyse** : **étude de la sécurité** des primitives

Présentation

Deux approches principales :

- la crypto**graphie** : **construction** de primitives
- la crypt**analyse** : **étude de la sécurité** des primitives

Pour cela, il faudra définir :

Présentation

Deux approches principales :

- la crypto**graphie** : **construction** de primitives
- la crypt**analyse** : **étude de la sécurité** des primitives

Pour cela, il faudra définir :

- qu'est ce qu'une **bonne** primitive cryptographique ?

  critères de **sécurité**
  critères d'**efficacité** (en temps, en espace)
  facilités de **déploiement**, infrastructure nécessaire, ...

Présentation

Deux approches principales :

- la crypto**graphie** : **construction** de primitives
- la crypt**analyse** : **étude de la sécurité** des primitives

Pour cela, il faudra définir :

- qu'est ce qu'une **bonne** primitive cryptographique ?
  - critères de **sécurité**
  - critères d'**efficacité** (en temps, en espace)
  - facilités de **déploiement**, infrastructure nécessaire, ...
- à quel **attaquant** / quelle **attaque** veut-on faire face ?
  - à quoi l'attaquant a-t-il accès ?
  - que veut-il obtenir ?

Le chiffrement

Le **chiffrement** a pour but de rendre une donnée **confidentielle**, c'est-à-dire la rendre inintelligible à certains, et intelligible à d'autres.

Le chiffrement

Le **chiffrement** a pour but de rendre une donnée **confidentielle**, c'est-à-dire la rendre inintelligible à certains, et intelligible à d'autres.

Deux protagonistes :

- une personne qui **chiffre** un **texte clair** m, afin de le rendre inintelligible
- une personne qui **déchiffre** le **chiffré** c, afin de pouvoir lire le message m correspondant

Le chiffrement

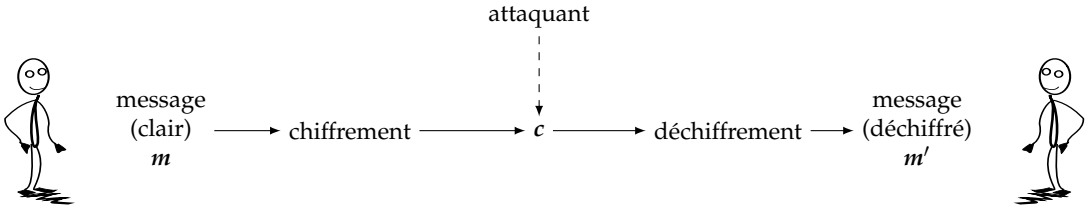
Le **chiffrement** a pour but de rendre une donnée **confidentielle**, c'est-à-dire la rendre inintelligible à certains, et intelligible à d'autres.

Deux protagonistes :

une personne qui **chiffre** un **texte clair** m, afin de le rendre inintelligible

une personne qui **déchiffre** le **chiffré** c, afin de pouvoir lire le message m correspondant

→ *Exemple pour le chiffrement :*



Le chiffrement

La connaissance d'une valeur secrète, appelée **clé de chiffrement**, permet d'effectuer ces actions de chiffrement et de déchiffrement. Cette clé peut...

- être **commune** aux protagonistes : on parle de **chiffrement symétrique**, ou **à clé secrète**
- **différer** selon le protagoniste et son rôle (chiffrement ou déchiffrement) : on parle de **chiffrement asymétrique**, ou **à clé publique**

Le chiffrement

La connaissance d'une valeur secrète, appelée **clé de chiffrement**, permet d'effectuer ces actions de chiffrement et de déchiffrement. Cette clé peut...

- être **commune** aux protagonistes : on parle de **chiffrement symétrique**, ou **à clé secrète**
- **différer** selon le protagoniste et son rôle (chiffrement ou déchiffrement) : on parle de **chiffrement asymétrique**, ou **à clé publique**

Ces deux grandes familles de chiffrement ont des avantages et inconvénients selon le **contexte d'application**.

Le chiffrement

La connaissance d'une valeur secrète, appelée **clé de chiffrement**, permet d'effectuer ces actions de chiffrement et de déchiffrement. Cette clé peut...

- être **commune** aux protagonistes : on parle de **chiffrement symétrique**, ou **à clé secrète**
- **différer** selon le protagoniste et son rôle (chiffrement ou déchiffrement) : on parle de **chiffrement asymétrique**, ou **à clé publique**

Ces deux grandes familles de chiffrement ont des avantages et inconvénients selon le **contexte d'application**.

Les **protocoles** et **algorithmes** mis en place vont ensuite différer selon de type de chiffrement choisi.

La signature

Qu'est ce qu'une **signature numérique** ?

La signature

Qu'est ce qu'une **signature numérique** ?

On souhaite **imiter**, voire **améliorer**, certaines propriétés des signatures manuscrites :

- **Intégrité** : on peut vérifier si le message a été modifié ou non.
- **Authenticité** : on peut associer un message à un émetteur.
- **Non-répudiation** : on ne peut pas nier avoir émis une signature valide.
- **Infalsifiabilité** : une autre personne ne peut pas prétendre avoir émis la signature.
- **Non-réutilisation :** on ne peut pas utiliser une même signature sur deux messages différents.

La signature

Qu'est ce qu'une **signature numérique** ?

On souhaite **imiter**, voire **améliorer**, certaines propriétés des signatures manuscrites :

- **Intégrité** : on peut vérifier si le message a été modifié ou non.
- **Authenticité** : on peut associer un message à un émetteur.
- **Non-répudiation** : on ne peut pas nier avoir émis une signature valide.
- **Infalsifiabilité** : une autre personne ne peut pas prétendre avoir émis la signature.
- **Non-réutilisation :** on ne peut pas utiliser une même signature sur deux messages différents.

La signature va s'**apposer** au message, et devra dépendre explicitement de lui.

La signature

Qu'est ce qu'une **signature numérique** ?

On souhaite **imiter**, voire **améliorer**, certaines propriétés des signatures manuscrites :

- **Intégrité** : on peut vérifier si le message a été modifié ou non.
- **Authenticité** : on peut associer un message à un émetteur.
- **Non-répudiation** : on ne peut pas nier avoir émis une signature valide.
- **Infalsifiabilité** : une autre personne ne peut pas prétendre avoir émis la signature.
- **Non-réutilisation :** on ne peut pas utiliser une même signature sur deux messages différents.

La signature va s'**apposer** au message, et devra dépendre explicitement de lui.

Exemples de ce qu'on souhaite signer **numériquement** : des emails, du code (mise à jour de logiciels), des transactions bancaires, de la communication publique (sites web), des clés de chiffrement, des certificats, etc.

D'autres primitives cryptographiques

Il existe beaucoup d'autres **primitives cryptographiques** :

- les **fonctions de hachage** cryptographiques, dont le but est de condenser une message long en une courte séquence, appelée **haché**, de sorte que :
  - deux messages différents (même proches) donnent deux hachés différents
  - étant donné un haché, il est impossible de retrouver un message correspondant

D'autres primitives cryptographiques

Il existe beaucoup d'autres **primitives cryptographiques** :

- les **fonctions de hachage** cryptographiques, dont le but est de condenser une message long en une courte séquence, appelée **haché**, de sorte que :
  - deux messages différents (même proches) donnent deux hachés différents
  - étant donné un haché, il est impossible de retrouver un message correspondant
- les **schémas d'engagement** (*commitment schemes*), dont le but est de permettre à un utilisateur de s'**engager** sur une valeur sans la révéler aux autres utilisateurs

D'autres primitives cryptographiques

Il existe beaucoup d'autres **primitives cryptographiques** :

- les **fonctions de hachage** cryptographiques, dont le but est de condenser une message long en une courte séquence, appelée **haché**, de sorte que :
  - deux messages différents (même proches) donnent deux hachés différents
  - étant donné un haché, il est impossible de retrouver un message correspondant
- les **schémas d'engagement** (*commitment schemes*), dont le but est de permettre à un utilisateur de s'**engager** sur une valeur sans la révéler aux autres utilisateurs
- le **retrait confidentiel d'information** (*private information retrieval*), dont le but est d'**accéder** à une information distante **sans révéler** l'identité de l'élément cherché

D'autres primitives cryptographiques

Il existe beaucoup d'autres **primitives cryptographiques** :

- les **fonctions de hachage** cryptographiques, dont le but est de condenser une message long en une courte séquence, appelée **haché**, de sorte que :
  - deux messages différents (même proches) donnent deux hachés différents
  - étant donné un haché, il est impossible de retrouver un message correspondant
- les **schémas d'engagement** (*commitment schemes*), dont le but est de permettre à un utilisateur de s'**engager** sur une valeur sans la révéler aux autres utilisateurs
- le **retrait confidentiel d'information** (*private information retrieval*), dont le but est d'**accéder** à une information distante **sans révéler** l'identité de l'élément cherché
- les **preuves cryptographiques**, qui peuvent être de toute sorte : preuve de **travail**, preuve de **stockage**, preuve de **connaissance**, preuve d'extraction, etc.

D'autres primitives cryptographiques

Il existe beaucoup d'autres **primitives cryptographiques** :

- les **fonctions de hachage** cryptographiques, dont le but est de condenser une message long en une courte séquence, appelée **haché**, de sorte que :
  - deux messages différents (même proches) donnent deux hachés différents
  - étant donné un haché, il est impossible de retrouver un message correspondant
- les **schémas d'engagement** (*commitment schemes*), dont le but est de permettre à un utilisateur de s'**engager** sur une valeur sans la révéler aux autres utilisateurs
- le **retrait confidentiel d'information** (*private information retrieval*), dont le but est d'**accéder** à une information distante **sans révéler** l'identité de l'élément cherché
- les **preuves cryptographiques**, qui peuvent être de toute sorte : preuve de **travail**, preuve de **stockage**, preuve de **connaissance**, preuve d'extraction, etc.
- le **calcul multipartite** sécurisé et distribué, pour lequel on souhaite déporter des calculs massifs sur des serveurs distants tout en préservant la confidentialité des données

D'autres primitives cryptographiques

Il existe beaucoup d'autres **primitives cryptographiques** :

- les **fonctions de hachage** cryptographiques, dont le but est de condenser une message long en une courte séquence, appelée **haché**, de sorte que :
  - deux messages différents (même proches) donnent deux hachés différents
  - étant donné un haché, il est impossible de retrouver un message correspondant
- les **schémas d'engagement** (*commitment schemes*), dont le but est de permettre à un utilisateur de s'**engager** sur une valeur sans la révéler aux autres utilisateurs
- le **retrait confidentiel d'information** (*private information retrieval*), dont le but est d'**accéder** à une information distante **sans révéler** l'identité de l'élément cherché
- les **preuves cryptographiques**, qui peuvent être de toute sorte : preuve de **travail**, preuve de **stockage**, preuve de **connaissance**, preuve d'extraction, etc.
- le **calcul multipartite** sécurisé et distribué, pour lequel on souhaite déporter des calculs massifs sur des serveurs distants tout en préservant la confidentialité des données
- etc.

Définir une notion de sécurité

Comment estimer la **sécurité** d'un système cryptographique ?

Définir une notion de sécurité

Comment estimer la **sécurité** d'un système cryptographique ?

Deux approches :

1. sécurité **inconditionnelle** : peu importe la capacité de calcul de l'attaquant, il ne peut pas obtenir d'information sur le secret,
2. sécurité **calculatoire** : casser le système nécessite au moins une certaine capacité de calcul.

Définir une notion de sécurité

Comment estimer la **sécurité** d'un système cryptographique ?

Deux approches :

1. sécurité **inconditionnelle** : peu importe la capacité de calcul de l'attaquant, il ne peut pas obtenir d'information sur le secret,
2. sécurité **calculatoire** : casser le système nécessite au moins une certaine capacité de calcul.

En pratique, une bonne sécurité calculatoire est **suffisante**.

Définir une notion de sécurité

Comment estimer la **sécurité** d'un système cryptographique ?

Deux approches :

1. sécurité **inconditionnelle** : peu importe la capacité de calcul de l'attaquant, il ne peut pas obtenir d'information sur le secret,
2. sécurité **calculatoire** : casser le système nécessite au moins une certaine capacité de calcul.

En pratique, une bonne sécurité calculatoire est **suffisante**. Actuellement,

- « très bonne sécurité » : $> 2^{128}$ opérations,
- « mauvaise sécurité » : $\leq 2^{80}$ opérations.

Plan

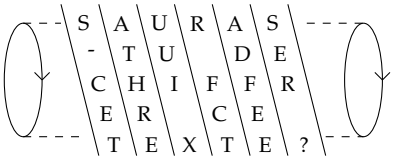
La scytale

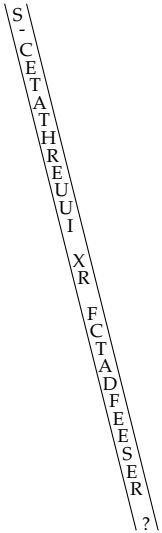
La **scytale** a été utilisée par les Grecs, à partir du 10ème siècle AEC.

La scytale

La **scytale** a été utilisée par les Grecs, à partir du 10ème siècle AEC.

Fonctionnement :

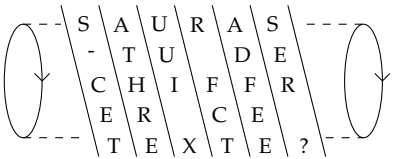




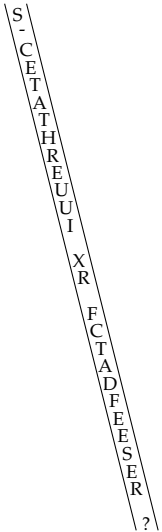
La scytale

La **scytale** a été utilisée par les Grecs, à partir du 10ème siècle AEC.

Fonctionnement :



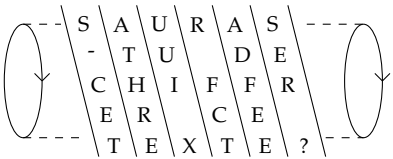
Ici, la **clef secrète** est le rayon du bâton, qui définit les dimensions du tableau ci-dessus.

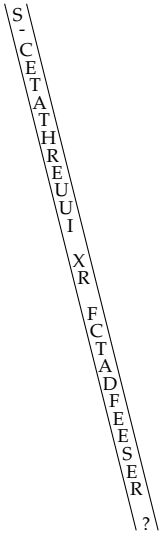


La scytale

La **scytale** a été utilisée par les Grecs, à partir du 10ème siècle AEC.

Fonctionnement :





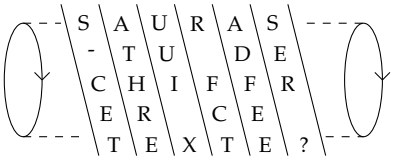
Ici, la **clef secrète** est le rayon du bâton, qui définit les dimensions du tableau ci-dessus.

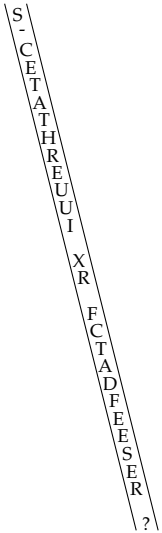
C'est un **chiffrement par transposition** (ou permutation) : l'ordre d'apparition des lettres est modifiée, mais ce sont les mêmes lettres dans le texte clair et dans le chiffré.

La scytale

La **scytale** a été utilisée par les Grecs, à partir du 10ème siècle AEC.

Fonctionnement :





Ici, la **clef secrète** est le rayon du bâton, qui définit les dimensions du tableau ci-dessus.

C'est un **chiffrement par transposition** (ou permutation) : l'ordre d'apparition des lettres est modifiée, mais ce sont les mêmes lettres dans le texte clair et dans le chiffré.

Question. Si l'on connaît le procédé de chiffrement, mais pas la valeur du rayon r de la scytale, est-il toujours possible de casser le système ?

Autre exemple

Chiffrement ***rail fence*** (palissade), aussi appelé **chiffrement zigzag**.

Autre exemple

Chiffrement ***rail fence*** (palissade), aussi appelé **chiffrement zigzag**.

Fonctionnement. On définit une hauteur de palissade h, qui est la clé de chiffrement. Puis, pour chiffrer un message, on place les lettres en zigzag (voir ci-dessous). Le chiffré correspond aux lettres lues ligne par ligne.

Autre exemple

Chiffrement ***rail fence*** (palissade), aussi appelé **chiffrement zigzag**.

Fonctionnement. On définit une hauteur de palissade h, qui est la clé de chiffrement. Puis, pour chiffrer un message, on place les lettres en zigzag (voir ci-dessous). Le chiffré correspond aux lettres lues ligne par ligne.

Exemple pour une palissade de hauteur $h = 5$. Si le message est :

`SALUT TOUT LE MONDE`

Autre exemple

Chiffrement ***rail fence*** (palissade), aussi appelé **chiffrement zigzag**.

Fonctionnement. On définit une hauteur de palissade h, qui est la clé de chiffrement. Puis, pour chiffrer un message, on place les lettres en zigzag (voir ci-dessous). Le chiffré correspond aux lettres lues ligne par ligne.

Exemple pour une palissade de hauteur $h = 5$. Si le message est :

```
SALUT TOUT LE MONDE
```

On obtient la palissade :

```
S_______U_______N__
_A_____O_T_____O_D_
__L___T___ ___M___E
___U_ _____L_ _____
____T_______E______
```

Autre exemple

Chiffrement ***rail fence*** (palissade), aussi appelé **chiffrement zigzag**.

Fonctionnement. On définit une hauteur de palissade h, qui est la clé de chiffrement. Puis, pour chiffrer un message, on place les lettres en zigzag (voir ci-dessous). Le chiffré correspond aux lettres lues ligne par ligne.

Exemple pour une palissade de hauteur $h = 5$. Si le message est :

```
SALUT TOUT LE MONDE
```

On obtient la palissade :

```
S_______U_______N__
_A_____O_T_____O_D_
__L___T___ ___M___E
___U_ _____L_ _____
____T_______E______
```

Et le chiffré est donc :

```
SUNAOTODLT MEU L TE
```

Autre exemple

Chiffrement ***rail fence*** (palissade), aussi appelé **chiffrement zigzag**.

Fonctionnement. On définit une hauteur de palissade h, qui est la clé de chiffrement. Puis, pour chiffrer un message, on place les lettres en zigzag (voir ci-dessous). Le chiffré correspond aux lettres lues ligne par ligne.

Exemple pour une palissade de hauteur $h = 5$. Si le message est :

```
SALUT TOUT LE MONDE
```

On obtient la palissade :

```
S_______U_______N__
_A_____O_T_____O_D_
__L___T___ ___M___E
___U_ _____L_ _____
____T_______E______
```

Et le chiffré est donc :

```
SUNAOTODLT MEU L TE
```

Exercices de programmation. Implanter ces deux chiffrements (et leur déchiffrement).

Plan

Chiffrement par substitution

Dans un chiffrement par **transposition** (vu précédemment), on **permute** l'ordre des lettres d'un texte.

Chiffrement par substitution

Dans un chiffrement par **transposition** (vu précédemment), on **permute** l'ordre des lettres d'un texte.

On pourrait également changer la **valeur** de ces lettres : on parle de chiffrement **par substitution**.

Chiffrement par substitution

Dans un chiffrement par **transposition** (vu précédemment), on **permute** l'ordre des lettres d'un texte.

On pourrait également changer la **valeur** de ces lettres : on parle de chiffrement **par substitution**.

Plusieurs manières :

- substitution **monoalphabétique** : à **chaque lettre** du message, on associe une autre est remplacée par une autre ;

Chiffrement par substitution

Dans un chiffrement par **transposition** (vu précédemment), on **permute** l'ordre des lettres d'un texte.

On pourrait également changer la **valeur** de ces lettres : on parle de chiffrement **par substitution**.

Plusieurs manières :

- substitution **monoalphabétique** : à **chaque lettre** du message, on associe une autre est remplacée par une autre ;
- substitution **polyalphabétique** : à **chaque groupe** de lettres de taille fixe, on associe un autre groupe de lettre de la même taille ;

Chiffrement par substitution

Dans un chiffrement par **transposition** (vu précédemment), on **permute** l'ordre des lettres d'un texte.

On pourrait également changer la **valeur** de ces lettres : on parle de chiffrement **par substitution**.

Plusieurs manières :

- substitution **monoalphabétique** : à **chaque lettre** du message, on associe une autre est remplacée par une autre ;
- substitution **polyalphabétique** : à **chaque groupe** de lettres de taille fixe, on associe un autre groupe de lettre de la même taille ;
- substitution par **polygramme** : même idée, mais avec des groupes de **taille variable**.

Exemples historiques

Premiers **exemples** :

- **Atbash** (5ème siècle AEC, chez les Hébreux) : la première lettre est remplacée par la dernière (dans l'ordre alphabétique), la deuxième par l'avant-dernière, etc.

Exemples historiques

Premiers **exemples** :

- **Atbash** (5ème siècle AEC, chez les Hébreux) : la première lettre est remplacée par la dernière (dans l'ordre alphabétique), la deuxième par l'avant-dernière, etc.
- **Cesar** (1er siècle AEC, chez les Romains) : la première lettre est remplacée par la quatrième, la seconde par la cinquième, etc.

Exemples historiques

Premiers **exemples** :

- **Atbash** (5ème siècle AEC, chez les Hébreux) : la première lettre est remplacée par la dernière (dans l'ordre alphabétique), la deuxième par l'avant-dernière, etc.
- **Cesar** (1er siècle AEC, chez les Romains) : la première lettre est remplacée par la quatrième, la seconde par la cinquième, etc.

Remarques.

– Dans ces deux cas, la i-ème lettre de l'alphabet est remplacée par la $f(i)$-ème lettre, où f est une **fonction affine**. On parle de **chiffrement affine**.

Exemples historiques

Premiers **exemples** :

- **Atbash** (5ème siècle AEC, chez les Hébreux) : la première lettre est remplacée par la dernière (dans l'ordre alphabétique), la deuxième par l'avant-dernière, etc.
- **Cesar** (1er siècle AEC, chez les Romains) : la première lettre est remplacée par la quatrième, la seconde par la cinquième, etc.

Remarques.

- Dans ces deux cas, la i-ème lettre de l'alphabet est remplacée par la $f(i)$-ème lettre, où f est une **fonction affine**. On parle de **chiffrement affine**.
- Définis comme tels, ni Atbash, ni Cesar ne nécessite une **clé** secrète.

Exemples historiques

Premiers **exemples** :

- **Atbash** (5ème siècle AEC, chez les Hébreux) : la première lettre est remplacée par la dernière (dans l'ordre alphabétique), la deuxième par l'avant-dernière, etc.
- **Cesar** (1er siècle AEC, chez les Romains) : la première lettre est remplacée par la quatrième, la seconde par la cinquième, etc.

Remarques.

- Dans ces deux cas, la i-ème lettre de l'alphabet est remplacée par la $f(i)$-ème lettre, où f est une **fonction affine**. On parle de **chiffrement affine**.
- Définis comme tels, ni Atbash, ni Cesar ne nécessite une **clé** secrète.
- Le chiffrement de Cesar peut être **généralisé** à un décalage de D lettres (au lieu de $D = 3$ fixé). Dans ce cas, la valeur de D est la clé.

Table de substitution

Pour définir un chiffrement par substitution monoalphabétique, il suffit de préciser la **table de substitution**. Exemples :

```
         A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z
cesar_3  D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z|A|B|C
atbash   Z|Y|X|W|V|U|T|S|R|Q|P|O|N|M|L|K|J|I|H|G|F|E|D|C|B|A
```

Table de substitution

Pour définir un chiffrement par substitution monoalphabétique, il suffit de préciser la **table de substitution**. Exemples :

```
         A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z
cesar_3  D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z|A|B|C
atbash   Z|Y|X|W|V|U|T|S|R|Q|P|O|N|M|L|K|J|I|H|G|F|E|D|C|B|A
```

On peut également associer aux lettres des **symboles** (au lieu d'autres lettres) ou des nombres.

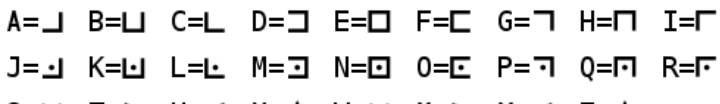
Table de substitution

Pour définir un chiffrement par substitution monoalphabétique, il suffit de préciser la **table de substitution**. Exemples :

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
cesar_3	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
atbash	Z	Y	X	W	V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A

On peut également associer aux lettres des **symboles** (au lieu d'autres lettres) ou des nombres.

Exemple : le chiffrement **Pigpen** utilisé par les franc-maçons :



Carré de Polybe

Le **carré de Polybe**, utilisé par les Grecs dans l'Antiquité, désigne une manière d'assigner à chaque lettre (ou presque) un couple d'entiers.

Carré de Polybe

Le **carré de Polybe**, utilisé par les Grecs dans l'Antiquité, désigne une manière d'assigner à chaque lettre (ou presque) un couple d'entiers.

Le carré de Polybe **standard** est :

	1	**2**	**3**	**4**	**5**
1	*A*	*B*	*C*	*D*	*E*
2	*F*	*G*	*H*	*I*/*J*	*K*
3	*L*	*M*	*N*	*O*	*P*
4	*Q*	*R*	*S*	*T*	*U*
5	*V*	*W*	*X*	*Y*	*Z*

Carré de Polybe

Le **carré de Polybe**, utilisé par les Grecs dans l'Antiquité, désigne une manière d'assigner à chaque lettre (ou presque) un couple d'entiers.

Le carré de Polybe **standard** est :

	1	**2**	**3**	**4**	**5**
1	*A*	*B*	*C*	*D*	*E*
2	*F*	*G*	*H*	*I*/*J*	*K*
3	*L*	*M*	*N*	*O*	*P*
4	*Q*	*R*	*S*	*T*	*U*
5	*V*	*W*	*X*	*Y*	*Z*

Pour **chiffrer**, on choisit une disposition des lettres différente du carré standard (sinon, tout le monde pourrait déchiffrer).

Carré de Polybe

Le **carré de Polybe**, utilisé par les Grecs dans l'Antiquité, désigne une manière d'assigner à chaque lettre (ou presque) un couple d'entiers.

Le carré de Polybe **standard** est :

	1	**2**	**3**	**4**	**5**
1	*A*	*B*	*C*	*D*	*E*
2	*F*	*G*	*H*	*I*/*J*	*K*
3	*L*	*M*	*N*	*O*	*P*
4	*Q*	*R*	*S*	*T*	*U*
5	*V*	*W*	*X*	*Y*	*Z*

Pour **chiffrer**, on choisit une disposition des lettres différente du carré standard (sinon, tout le monde pourrait déchiffrer).

Historiquement :

- la **clé** est un mot gardé secrètement,
- on dispose, de haut en bas et de gauche à droite, chaque lettre du mot dans le carré selon leur ordre d'apparition,
- on finit de remplir le carré avec les lettres manquantes dans l'ordre alphabétique.

Carré de Polybe : exemple

Exemple avec la clé formée par le mot `INFORMATIQUE`.

Le carré devient :

	1	**2**	**3**	**4**	**5**
1	*I/J*	*N*	*F*	*O*	*R*
2	*M*	*A*	*T*	*Q*	*U*
3	*E*	*B*	*C*	*D*	*G*
4	*H*	*K*	*L*	*P*	*S*
5	*V*	*W*	*X*	*Y*	*Z*

Puis, si l'on souhaite chiffrer le mot `LOGICIEL` on obtient :

43 14 35 11 33 11 31 43

Cryptanalyse historique

Les chiffrements par **substitution monoalphabétique** ont un inconvénient majeur : la **distribution d'apparition des lettres** est similaire dans le texte clair et dans le chiffré.

Les chiffrements par **substitution monoalphabétique** ont un inconvénient majeur : la **distribution d'apparition des lettres** est similaire dans le texte clair et dans le chiffré.

En français, le `E` est la lettre la plus fréquente :

A	B	C	D	**E**	F	G	H	I	J	K	L	M
9,42	1,02	2,64	3,39	**15,87**	0,95	1,04	0,77	8,41	0,89	0,00	5,34	3,24
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
7,15	5,14	2,86	1,06	6,46	7,90	7,26	6,24	2,15	0,00	0,30	0,24	0,3

Les chiffrements par **substitution monoalphabétique** ont un inconvénient majeur : la **distribution d'apparition des lettres** est similaire dans le texte clair et dans le chiffré.

En français, le `E` est la lettre la plus fréquente :

A	B	C	D	**E**	F	G	H	I	J	K	L	M
9,42	1,02	2,64	3,39	**15,87**	0,95	1,04	0,77	8,41	0,89	0,00	5,34	3,24
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
7,15	5,14	2,86	1,06	6,46	7,90	7,26	6,24	2,15	0,00	0,30	0,24	0,3

Avec le chiffrement d'atbash, si le texte chiffré est en français, alors on doit donc observer une proportion importante de lettres `V` dans le chiffré.

Les chiffrements par **substitution monoalphabétique** ont un inconvénient majeur : la **distribution d'apparition des lettres** est similaire dans le texte clair et dans le chiffré.

En français, le `E` est la lettre la plus fréquente :

A	B	C	D	**E**	F	G	H	I	J	K	L	M
9,42	1,02	2,64	3,39	**15,87**	0,95	1,04	0,77	8,41	0,89	0,00	5,34	3,24
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
7,15	5,14	2,86	1,06	6,46	7,90	7,26	6,24	2,15	0,00	0,30	0,24	0,3

Avec le chiffrement d'atbash, si le texte chiffré est en français, alors on doit donc observer une proportion importante de lettres `V` dans le chiffré.

Attention : la distribution des symboles à chiffrer varie selon la langue employée, le type de document à chiffrer (une image, un script `Lisp`, ...)

Cette **non-uniformité** peut permettre de retrouver la clé secrète d'un chiffrement monoalphabétique.

Cette **non-uniformité** peut permettre de retrouver la clé secrète d'un chiffrement monoalphabétique.

Exemple pour le chiffrement de Cesar de décalage Δ.

Cryptanalyse historique

Cette **non-uniformité** peut permettre de retrouver la clé secrète d'un chiffrement monoalphabétique.

Exemple pour le chiffrement de Cesar de décalage Δ.

On peut deviner le décalage Δ en cherchant la lettre la plus fréquente :

- $\rightarrow$ en effet, la lettre `E` $= L_4$ est chiffrée en $L_{4+\Delta \mod 26}$
- $\rightarrow$ la lettre la plus fréquente dans le chiffré est donc $L_{4+\Delta \mod 26}$; en la retrouvant on obtient la valeur de Δ

Pour un **chiffrement affine** quelconque, on va chercher les deux lettres les plus fréquentes.

Chiffrement de Vigenère

Le **chiffrement de Vigenère** est un **chiffrement polyalphabétique**.

Chiffrement de Vigenère

Le **chiffrement de Vigenère** est un **chiffrement polyalphabétique**.

- La clé est un **mot** `K[0]K[1]...K[l-1]` ; on note ℓ sa longueur (en nombre de lettres).
- Pour chiffrer un texte `T = T[0]T[1]...T[n]` :
  1. la première lettre `T[0]` est chiffrée en la décalant du rang (dans l'alphabet) de la première lettre `K[0]` de la clé ;
  2. la deuxième lettre `T[1]` est chiffrée en la décalant du rang de la deuxième lettre `K[1]` de la clé ;
  3. quand on arrive à la $(\ell + 1)$-ème lettre du texte, on repart de la première lettre de la clé : `T[`ℓ`]` est décalée suivant `K[0]`, `T[`ℓ`+1]` suivant `K[1]`, etc.

Chiffrement de Vigenère

Le **chiffrement de Vigenère** est un **chiffrement polyalphabétique**.

- La clé est un **mot** `K[0]K[1]...K[l-1]` ; on note ℓ sa longueur (en nombre de lettres).
- Pour chiffrer un texte `T = T[0]T[1]...T[n]` :
  1. la première lettre `T[0]` est chiffrée en la décalant du rang (dans l'alphabet) de la première lettre `K[0]` de la clé ;
  2. la deuxième lettre `T[1]` est chiffrée en la décalant du rang de la deuxième lettre `K[1]` de la clé ;
  3. quand on arrive à la $(\ell+1)$-ème lettre du texte, on repart de la première lettre de la clé : `T[`ℓ`]` est décalée suivant `K[0]`, `T[`ℓ`+1]` suivant `K[1]`, etc.

Exemple : avec la clé « `CRYPTO` » et le texte « `LE CHIFFREMENT DE VIGENERE` », on obtient le chiffré :

`NV AWBTHICBXBV UC KBUGECGX`

Chiffrement de Vigenère

Le **chiffrement de Vigenère** est un **chiffrement polyalphabétique**.

- La clé est un **mot** `K[0]K[1]...K[l-1]` ; on note ℓ sa longueur (en nombre de lettres).
- Pour chiffrer un texte `T = T[0]T[1]...T[n]` :
  1. la première lettre `T[0]` est chiffrée en la décalant du rang (dans l'alphabet) de la première lettre `K[0]` de la clé ;
  2. la deuxième lettre `T[1]` est chiffrée en la décalant du rang de la deuxième lettre `K[1]` de la clé ;
  3. quand on arrive à la $(\ell+1)$-ème lettre du texte, on repart de la première lettre de la clé : `T[`ℓ`]` est décalée suivant `K[0]`, `T[`ℓ`+1]` suivant `K[1]`, etc.

Exemple : avec la clé « `CRYPTO` » et le texte « `LE CHIFFREMENT DE VIGENERE` », on obtient le chiffré :

`NV AWBTHICBXBV UC KBUGECGX`

On observe que les 7 « `E` » du texte n'ont pas toujours la même valeur chiffrée.

Au premier abord, cette technique permet d'éviter l'attaque directe vue sur le chiffrement de César, car le `E` est chiffré différemment suivant sa position dans le texte.

Au premier abord, cette technique permet d'éviter l'attaque directe vue sur le chiffrement de César, car le `E` est chiffré différemment suivant sa position dans le texte.

Néanmoins, on peut essayer de **se ramener** à cette attaque :

- supposons que l'on connaisse la longueur ℓ de la clé (on verra après comment faire)

Au premier abord, cette technique permet d'éviter l'attaque directe vue sur le chiffrement de César, car le `E` est chiffré différemment suivant sa position dans le texte.

Néanmoins, on peut essayer de **se ramener** à cette attaque :

- supposons que l'on connaisse la longueur ℓ de la clé (on verra après comment faire)
- on découpe le chiffré `C = C[0]C[1]...C[n]` en ℓ morceaux

$$\begin{aligned}
\texttt{C}_0 &= \texttt{C}[0]\texttt{C}[\ell]\texttt{C}[2*\ell]... \\
\texttt{C}_1 &= \texttt{C}[1]\texttt{C}[1+\ell]\texttt{C}[1+2*\ell]... \\
&... \\
\texttt{C}_{\ell-1} &= \texttt{C}[\ell-1]\texttt{C}[\ell-1+\ell]\texttt{C}[\ell-1+2*\ell]...
\end{aligned}$$

Au premier abord, cette technique permet d'éviter l'attaque directe vue sur le chiffrement de César, car le `E` est chiffré différemment suivant sa position dans le texte.

Néanmoins, on peut essayer de **se ramener** à cette attaque :

- supposons que l'on connaisse la longueur ℓ de la clé (on verra après comment faire)
- on découpe le chiffré `C = C[0]C[1]...C[n]` en ℓ morceaux

$$
\begin{aligned}
\texttt{C}_0 &= \texttt{C}[0]\texttt{C}[\ell]\texttt{C}[2*\ell]... \\
\texttt{C}_1 &= \texttt{C}[1]\texttt{C}[1+\ell]\texttt{C}[1+2*\ell]... \\
&... \\
\texttt{C}_{\ell-1} &= \texttt{C}[\ell-1]\texttt{C}[\ell-1+\ell]\texttt{C}[\ell-1+2*\ell]...
\end{aligned}
$$

- chaque lettre du morceau C_i a été chiffré avec la même lettre $K[i]$ de la clé K ; on peut donc utiliser l'attaque par fréquence sur le chiffrement de Cesar.

Exemple avec les commandes bash

Avec des commandes `bash`, et une longueur de clé connue $\ell = 6$ (par exemple), on va effectuer :

```
fold -w 6 <filename> | cut -b <i> | sort | uniq -c
```

pour retrouver la i-ème lettre de la clé.

Exemple avec les commandes bash

Avec des commandes `bash`, et une longueur de clé connue $\ell = 6$ (par exemple), on va effectuer :

```
fold -w 6 <filename> | cut -b <i> | sort | uniq -c
```

pour retrouver la i-ème lettre de la clé.

Explication :

- `fold -w 6` permet d'afficher un fichier par lignes de taille 6
- `cut -b <i>` permer de sélectionner la colonne i du texte précédent
- `sort` permet de trier le texte précédent
- `uniq -c` affiche le nombre d'apparition de lignes consécutives (ici une ligne = une lettre)

Retrouver la longueur de la clé

Pour terminer l'attaque, il reste à **retrouver la longueur de la clé**...

Retrouver la longueur de la clé

Pour terminer l'attaque, il reste à **retrouver la longueur de la clé**...

Pour $1 \leq i \leq 26$, on note M_i le nombre de fois que la lettre numéro i apparaît dans le texte T.

Retrouver la longueur de la clé

Pour terminer l'attaque, il reste à **retrouver la longueur de la clé**...

Pour $1 \leq i \leq 26$, on note M_i le nombre de fois que la lettre numéro i apparaît dans le texte T.

En 1920, Friedman introduit l'**indice de coincidence** d'un texte T :

$$\mathrm{IC}(T) = \frac{\sum_{i=1}^{26} M_i(M_i - 1)}{N(N-1)}$$

de sorte qu'une grande disparité dans les fréquences d'apparition des lettres donne un grand indice de coincidence.

Retrouver la longueur de la clé

Pour terminer l'attaque, il reste à **retrouver la longueur de la clé**...

Pour $1 \leq i \leq 26$, on note M_i le nombre de fois que la lettre numéro i apparaît dans le texte T.

En 1920, Friedman introduit l'**indice de coincidence** d'un texte T :

$$\mathrm{IC}(T) = \frac{\sum_{i=1}^{26} M_i(M_i - 1)}{N(N-1)}$$

de sorte qu'une grande disparité dans les fréquences d'apparition des lettres donne un grand indice de coincidence.

Si le texte T :

- est formé de lettres tirées **uniformément** et indépendemment : $\mathrm{IC} = \frac{1}{26} \simeq 0.038$
- est tiré du **français** : $\mathrm{IC} \simeq 0.078$;
- est tiré de l'**anglais** : $\mathrm{IC} \simeq 0.067$.

Retrouver la longueur de la clé

Pour terminer l'attaque, il reste à **retrouver la longueur de la clé**...

Pour $1 \leq i \leq 26$, on note M_i le nombre de fois que la lettre numéro i apparaît dans le texte T.

En 1920, Friedman introduit l'**indice de coincidence** d'un texte T :

$$\mathrm{IC}(T) = \frac{\sum_{i=1}^{26} M_i(M_i - 1)}{N(N-1)}$$

de sorte qu'une grande disparité dans les fréquences d'apparition des lettres donne un grand indice de coincidence.

Si le texte T :

- est formé de lettres tirées **uniformément** et indépendemment : $\mathrm{IC} = \frac{1}{26} \simeq 0.038$
- est tiré du **français** : $\mathrm{IC} \simeq 0.078$;
- est tiré de l'**anglais** : $\mathrm{IC} \simeq 0.067$.

Comment retrouver la longueur de la clé ?

Comment en déduire la longueur de la clé ? L'idée est de vérifier si les morceaux C_i définis précédemment (selon une longueur ℓ) ont un indice de coincidence proche de celui du français.

Comment retrouver la longueur de la clé ?

Comment en déduire la longueur de la clé ? L'idée est de vérifier si les morceaux C_i définis précédemment (selon une longueur ℓ) ont un indice de coincidence proche de celui du français.

On obtient finalement l'algorithme suivant.

Algorithme 2 : Calcul de la longueur d'une clé de Vigenère

Data : un chiffré C par une clé K de longueur ℓ
Result : la longueur ℓ

```
test ← 0
ℓ ← 0
while test < 0.075 do
    ℓ ← ℓ + 1
    T ← C[0]C[ℓ]C[2ℓ]C[3ℓ]...
    IC ← IC(T)
end
return ℓ
```

Vers une cryptographie moderne

Les chiffrements vus précedemment souffrent d'un **problème majeur** : le chiffré contient des **biais statistiques** qui révèlent de l'information sur la clé et/ou le clair.

Vers une cryptographie moderne

Les chiffrements vus précedemment souffrent d'un **problème majeur** : le chiffré contient des **biais statistiques** qui révèlent de l'information sur la clé et/ou le clair.

Historiquement, grâce à des **progrès techniques** en automatique, des mécanismes plus complexes sont apparus, afin de rendre l'analyse fréquentielle plus difficile :

- **chiffre de Hill** (années 1920)
- machine **Enigma** (seconde guerre mondiale)

Vers une cryptographie moderne

Les chiffrements vus précedemment souffrent d'un **problème majeur** : le chiffré contient des **biais statistiques** qui révèlent de l'information sur la clé et/ou le clair.

Historiquement, grâce à des **progrès techniques** en automatique, des mécanismes plus complexes sont apparus, afin de rendre l'analyse fréquentielle plus difficile :

- **chiffre de Hill** (années 1920)
- machine **Enigma** (seconde guerre mondiale)

Cependant, l'arrivée de l'**informatique** a multiplié la puissance des attaques sur ces systèmes, en permettant des recherches exhaustives et des calculs plus efficaces.

Vers une cryptographie moderne

Les chiffrements vus précedemment souffrent d'un **problème majeur** : le chiffré contient des **biais statistiques** qui révèlent de l'information sur la clé et/ou le clair.

Historiquement, grâce à des **progrès techniques** en automatique, des mécanismes plus complexes sont apparus, afin de rendre l'analyse fréquentielle plus difficile :

- **chiffre de Hill** (années 1920)
- machine **Enigma** (seconde guerre mondiale)

Cependant, l'arrivée de l'**informatique** a multiplié la puissance des attaques sur ces systèmes, en permettant des recherches exhaustives et des calculs plus efficaces.

On s'est donc dirigé vers une **cryptographie moderne**, dont l'objectif est double :

1. avoir des chiffrements pouvant traiter rapidement **n'importe quelle donnée**, notamment des fichiers informatiques ;
2. obtenir des **preuves**/certitudes de l'invulnérablité des chiffrements, notamment en éliminant les biais statistiques des chiffrements par substitution.