Présentation

- Pascal PETIT
- 01 60 87 39 03 (peu joignable)
- petit@shayol.org (trop joignable)
- http://www.ibisc.fr/~petit
- Crypto: bibi
- Compression: N. Thibault

Chiffrement

- Auteur : P. Petit (pascal.petit@shayol.org)
- La bibliographie cite des sources/ressources utiles de deux catégories:
  - Des sources dont je me suis inspiré pour certains points (notamment support de cours de C. Laforest qui a assuré cet enseignement jusqu'en 2007-2008)
  - Des sources pour des approfondissements
- Les schémas/photos qui ne sont pas de moi ont en général étaient récupérés sur wikipedia

Définitions

- *Texte en clair* : un texte sous sa forme originale, compréhensible tel quel
- *Chiffrement* : transformer un texte en clair en un texte incompréhensible
- La transformation inverse quand on possède tous les éléments pour le faire s'appelle le *déchiffrement* (ne pas confondre avec décryptage)
- *Décrypter* un texte: faire de même sans avoir les éléments (clef, ...). Le correspondant légitime déchiffre le texte, l'attaquant le décrypte.

Définitions

- *cryptographie :* domaine scientifique et technique dont le but est de garder les messages secrets. Pratiquée par les cryptographes
- *cryptanalyse* : domaine scientifique et technique dont le but est de retrouver les messages en clair sans avoir tous les éléments pour le faire. Pratiquée par les cryptanalystes
- *cryptologie:* : regroupe cryptographie et cryptanalyse. Pratiquée par les cryptologues.
- *Stéganographie*: domaine scientifique et technique dont le but est de faire passer inaperçu un message dans un autre objet

Stéganographie : exemple (G. Sand à A. De Musset)

Je suis tres emue de vous dire que j'ai
bien compris l'autre soir que vous aviez
toujours une envie folle de me faire
danser. Je garde le souvenir de votre
baiser et je voudrais bien que ce soit
la une preuve que je puisse etre aimee
par vous. Je suis prete a montrer mon
affection toute desinteressee et sans cal-
cul, et si vous voulez me voir aussi
vous devoiler sans artifice mon ame
toute nue, venez me faire une visite.
Nous causerons en amis, franchement.
Je vous prouverai que je suis la femme
sincere, capable de vous offrir l'affection
la plus profonde comme la plus etroite
en amitie, en un mot la meilleure preuve
que vous puissiez rever, puisque votre
ame est libre. Pensez que la solitude ou j'ha-
bite est bien longue, bien dure et souvent
difficile. Ainsi, en y songeant j'ai l'ame
grosse. Accourez donc vite et venez me la
faire oublier par l'amour ou je veux me
mettre.

Définitions

- Soit M un message en clair à faire transiter de façon sûre entre Alfred et Bachir
- Soit E le processus de chiffrement
- Soit D le processus de déchiffrement
- Alfred calcule et transmet C=E(M)
- Bachir reçoit C et doit connaître D pour retrouver M=D(C)
- On doit avoir M=D(E(M))

Historique

- L'artisanat
  - Chiffrement de césar
  - Permutation et attaque statistique
- La technique
  - ENIGMA
- Masque jetable
- L'ère scientifique (actuelle)
- Principes actuels

Chiffrement de césar

k=2

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
c	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

salut les tepos

chiffrer (k=2)

ucnwv ngu vgrqu

Déchiffrer (k=2)

salut les tepos

Chiffrement de césar

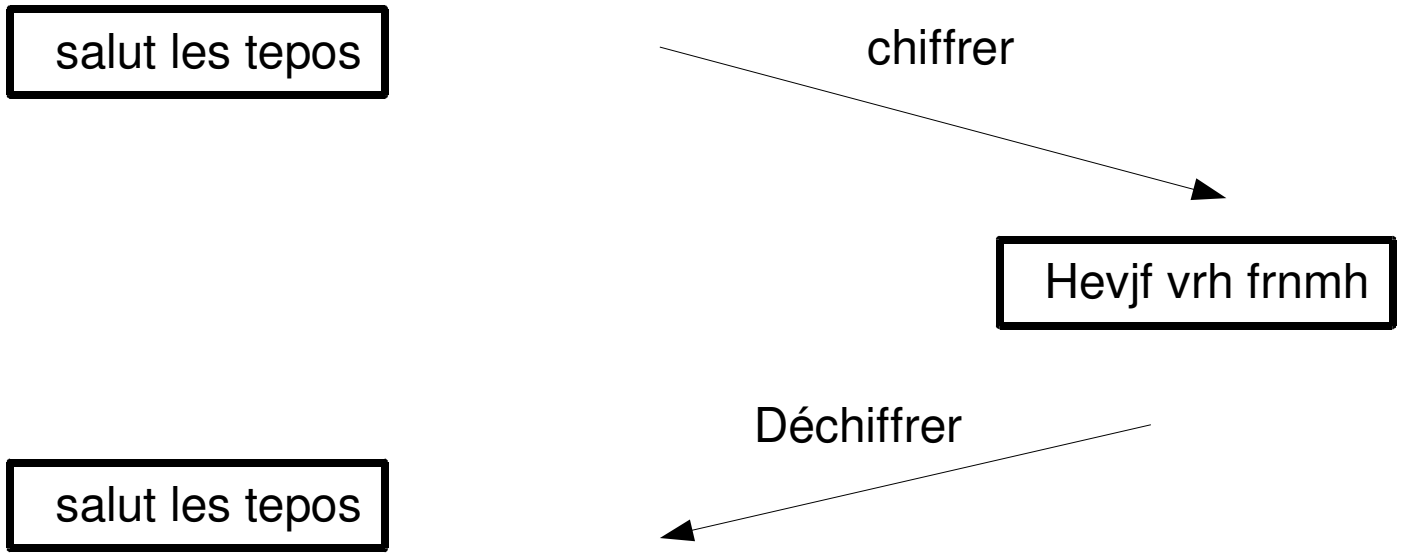
- Décaler chaque lettre de K caractères dans l'alphabet
- Exemple1: k=2 « salut les tepos » devient « ucnwv ngu vgrqu »
- Le déchiffrement se fait en décalant les lettres dans l'autre sens
- Exemple2: rot13: l'opération de codage et de décodage sont les mêmes (but: qu'un texte ne soit lu que par les gens qui souhaitent le lire. Utilisé couramment sur les forums USENET)

Chiffrement de césar

- Une fois le mécanisme est connu, la sécurité repose sur la connaissance de k
- 25 valeurs possibles pour k. Il suffit de les essayer toutes jusqu'à trouver un texte qui a un sens
- Attaque en
- brute: on essaie tout ou partie des clefs
- Ici, l'espace des clefs est trop petit

Permutation alphabétique

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
e	t	c	g	r	u	i	o	q	a	x	v	w	s	m	n	b	d	h	f	j	k	l	y	z	p



Permutation alphabétique

- On utilise une permutation quelconque de l'alphabet
- Avec un alphabet de n caractères, n ! permutations possibles
- L'espace des clefs devient très grand (26! ~ 2^88). L'attaque par force brute devient trop longue

Permutation alphabétique

- Problème: chaque lettre est toujours remplacée par la même lettre. Si on a réussit à la décoder une fois, on saura la décoder partout
- 3 attaques complémentaires possibles :
  - Parfois, de notre connaissance de la structure du message, on peut en déchiffrer certaines parties
    - On peut déchiffrer les lettres correspondantes partout dans le message
  - Attaque statistique: connaissant la langue dans laquelle le message a été écrit, on peut s'appuyer sur des statistiques d'occurences des caractères dans cette langue

Permutation alphabétique: exemple

- Sur la page WeB de ressources du cours, on vous propose un fichier chiffré à l'aide d'une permutation alphabétique
- Voir http://www.ibisc.fr/~petit
- Déchiffrez le.

Chiffrement de vigenère: substitution polyalphabétique

- Principe: chaque lettre du message est chiffré à l'aide d'un chiffrement de César avec un décalage différent;
- Pour simplifier la transmission des tables, on indique la version chiffrée de A (et on en déduit le décalage et donc les versions chiffrées des autres lettres);
- La clef est un texte indiquant la suite des versions chiffrées de la lettre A
- Un fois arrivé au bout de la clef, on repart au début

Vigénère: exemple

- Clef: tepos
- Texte à chiffrer: « ilfai tfaim »

```
i l f a i t f a i m
t e p o s t e p o s
---------------------
B P U O A M J P W E
```

- Version chiffrée: BPUOA MJPWE

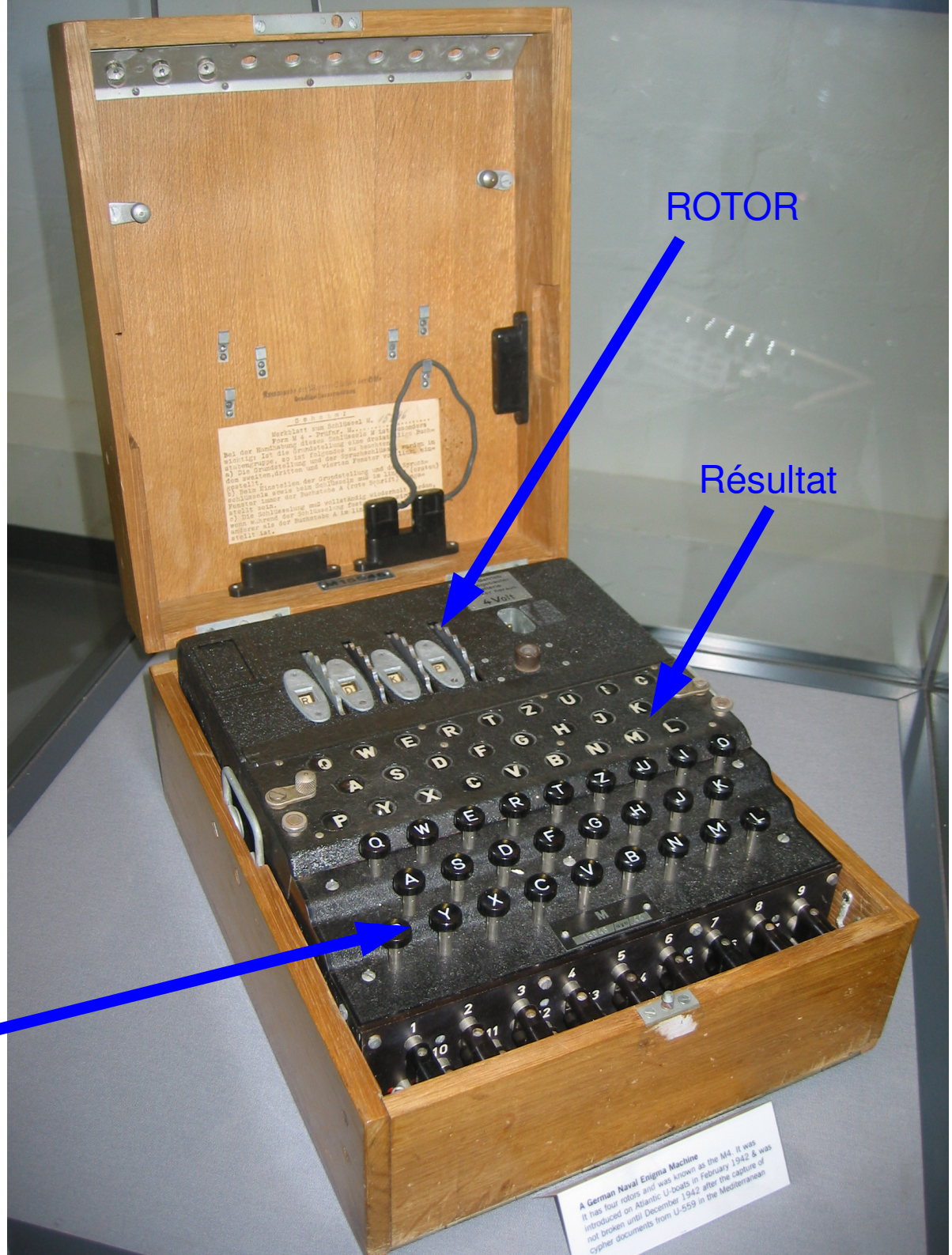
A	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*I*	*J*	*K*	*L*	*M*	*N*	*O*	*P*	*Q*	*R*	*S*	*T*	*U*	*V*	*W*	*X*	*Y*	*Z*
B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z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

Transposition

- Transposition: changer l'ordre des lettres d'un texte
- Exemple:
  - on range le texte du message dans un tableau rectangulaire ligne par ligne
  - Le message chiffré s'obtient en lisant le tableau colonne par colonne

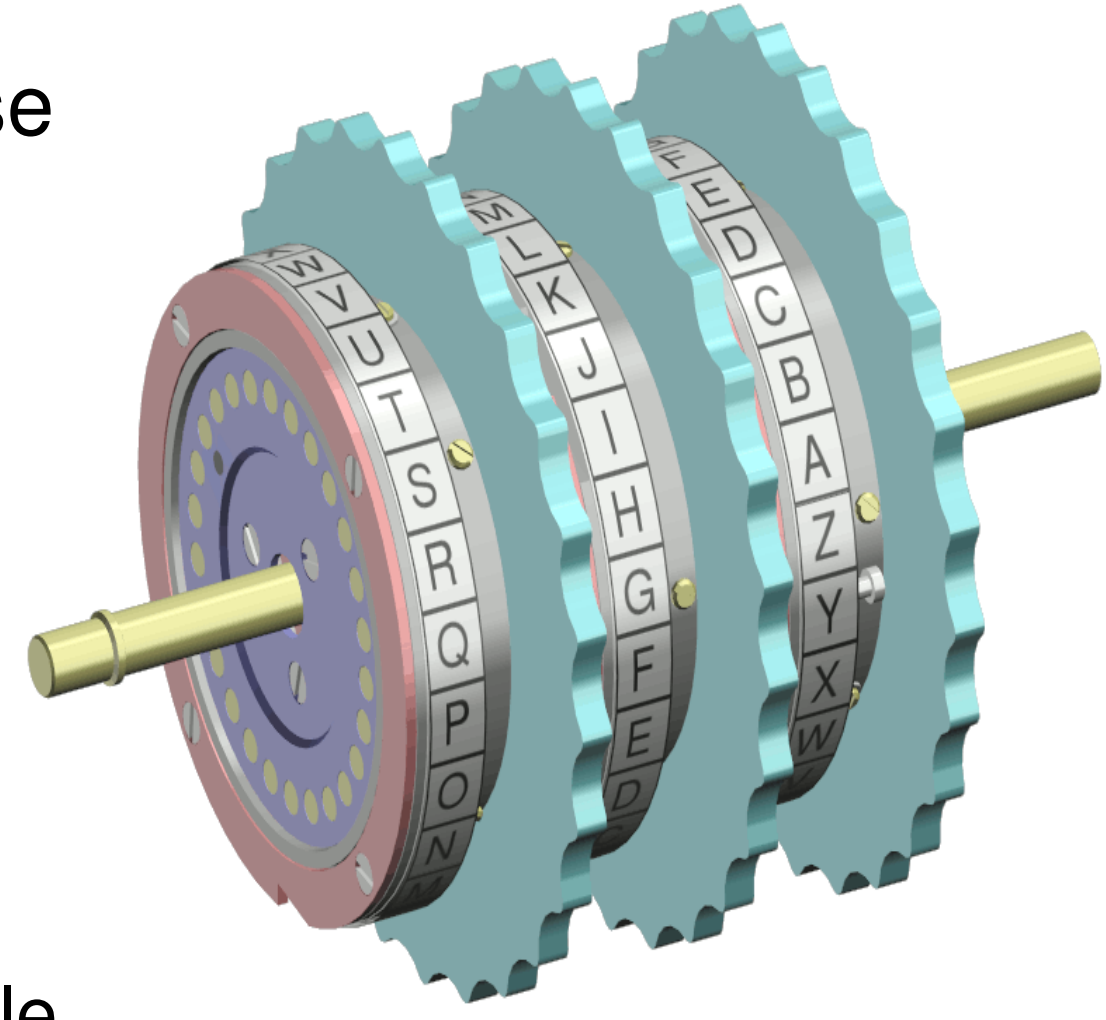
ENIGMA

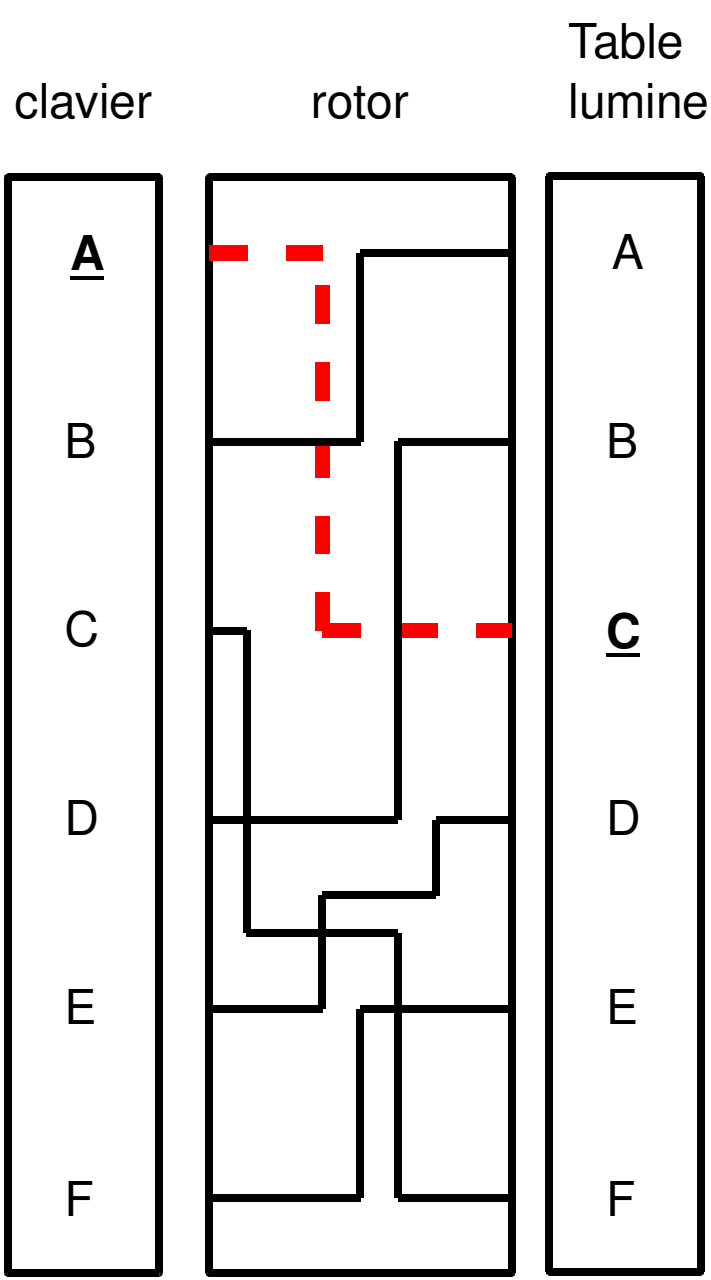
- Utilisé par les allemands durant la seconde guerre mondiale
- déchiffré par les anglais sur de longue périodes



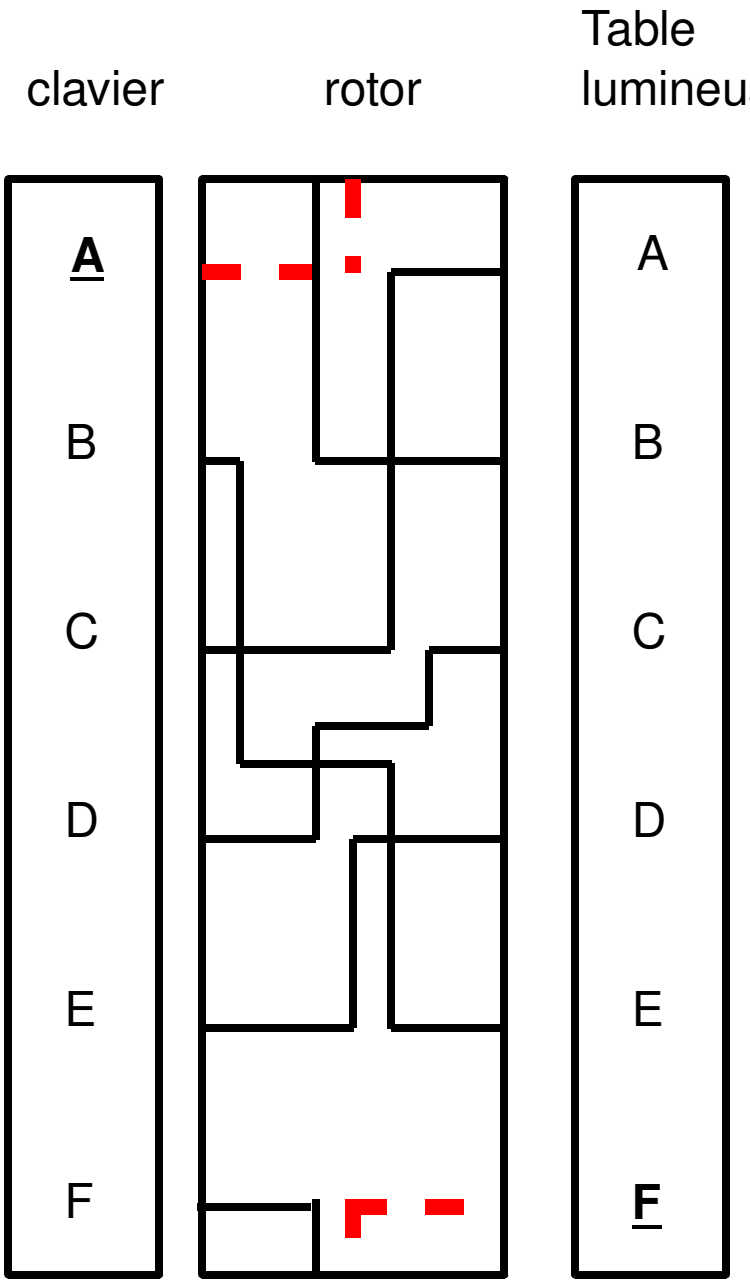
ENIGMA

- Chaque rotor réalise une substitution alphabétique
- À chaque codage d'un caractère,
  - le premier rotor tourne
  - Quand le premier rotor a fait un tour, le second rotor tourne
  - ...





Enigma



Enigma après avancement du rotor d'un cran

Enigma

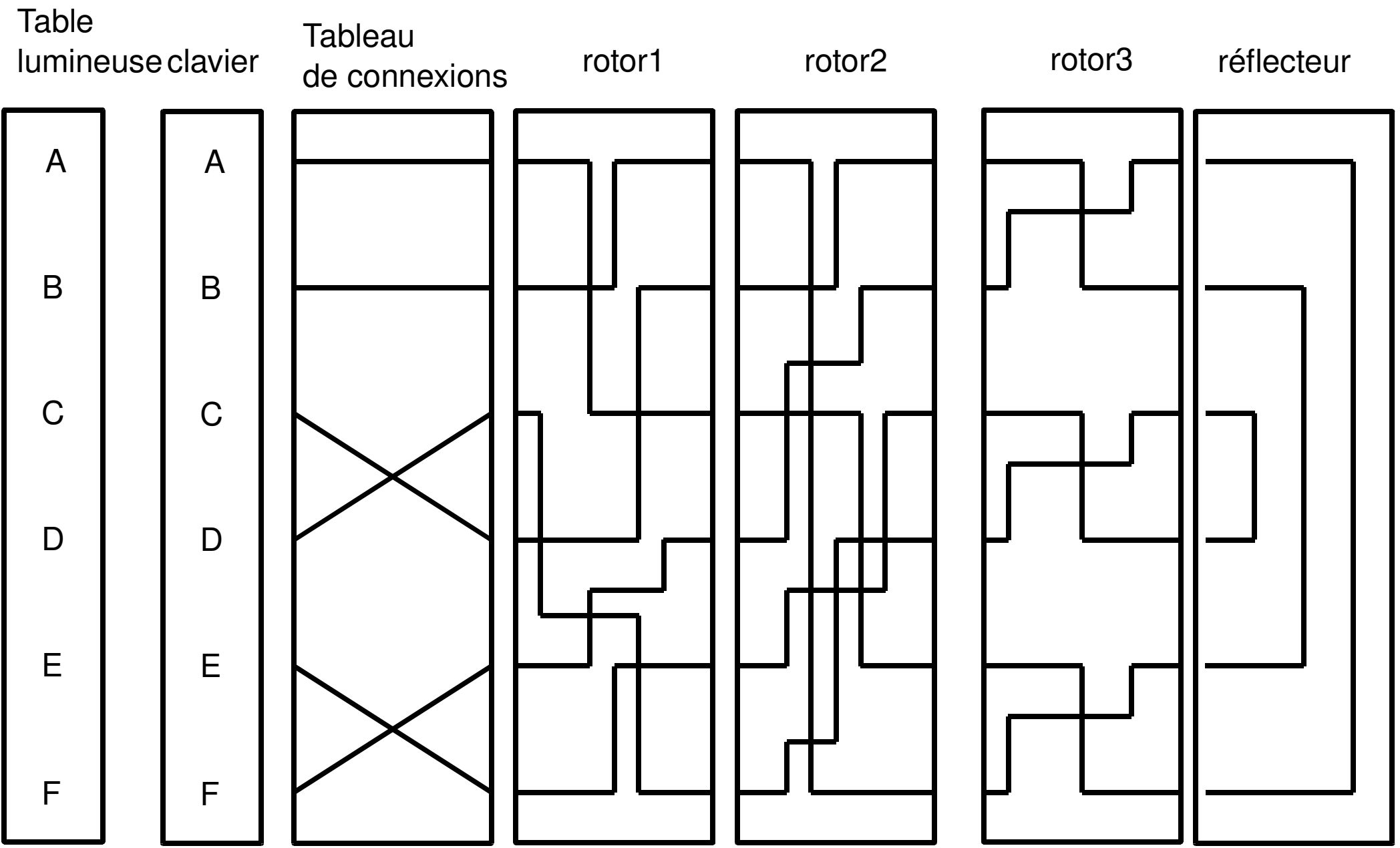
- Conséquence :
  - Les substituions changent donc à chaque caractère codé
  - Il faut que tous les rotors aient fait un tour complet pour qu'on retrouve la même transformation
    - Avec un rotor alphabétique, il faudrait 26 mouvements pour revenir à la transformations initiale
    - Avec deux rotors, 26*26
    - Avec n rotor 26^n
  - Les attaques sur vigenere ont montré la faiblesse que constituait de la répétition des transformations
  - Là, on les limite à défaut de les supprimer

ENIGMA: nature et nombre des clefs

- Clef:
  - Le choix de la position des rotors (un rotor = une permutation): 6 possibilités
  - La position initiale des rotors => 26^3=17576 clefs
- Ça fait beaucoup pour des attaques manuelles
- Plus tard: jusqu'à 8 rotors + choix des rotors
- Un mécanisme de permutations de 6 fois 2 lettres multiplie le nombre de possibilités par 100 391 791 500
- => espace de clef très grand = #10^{16}

ENIGMA: transpositions

- Pour compliquer :
  - ajout d'un étage de transposition permettant d'échanger deux lettres: ENIGMA avait 6 mécanismes permettant l'échange de 2 lettres au choix
  - Ajoute 100 391 791 500 possibilités = C(26,12)*11*9*7*5*3*1
    - C(26,12): choisir 12 lettres parmi 26 sans tenir compte de l'ordre
    - Ensuite, on choisir la lettre qu'on associe à la première: 11 possibilités.
    - Il reste 10 lettres disponible.On fait de même pour la première de ces 10 lettres: reste 9 possiblités
    - ...



Permutations: 2 permutations de 2 lettres dans notre exemple

Réflecteur: codage et décodage sont une seule et même opération

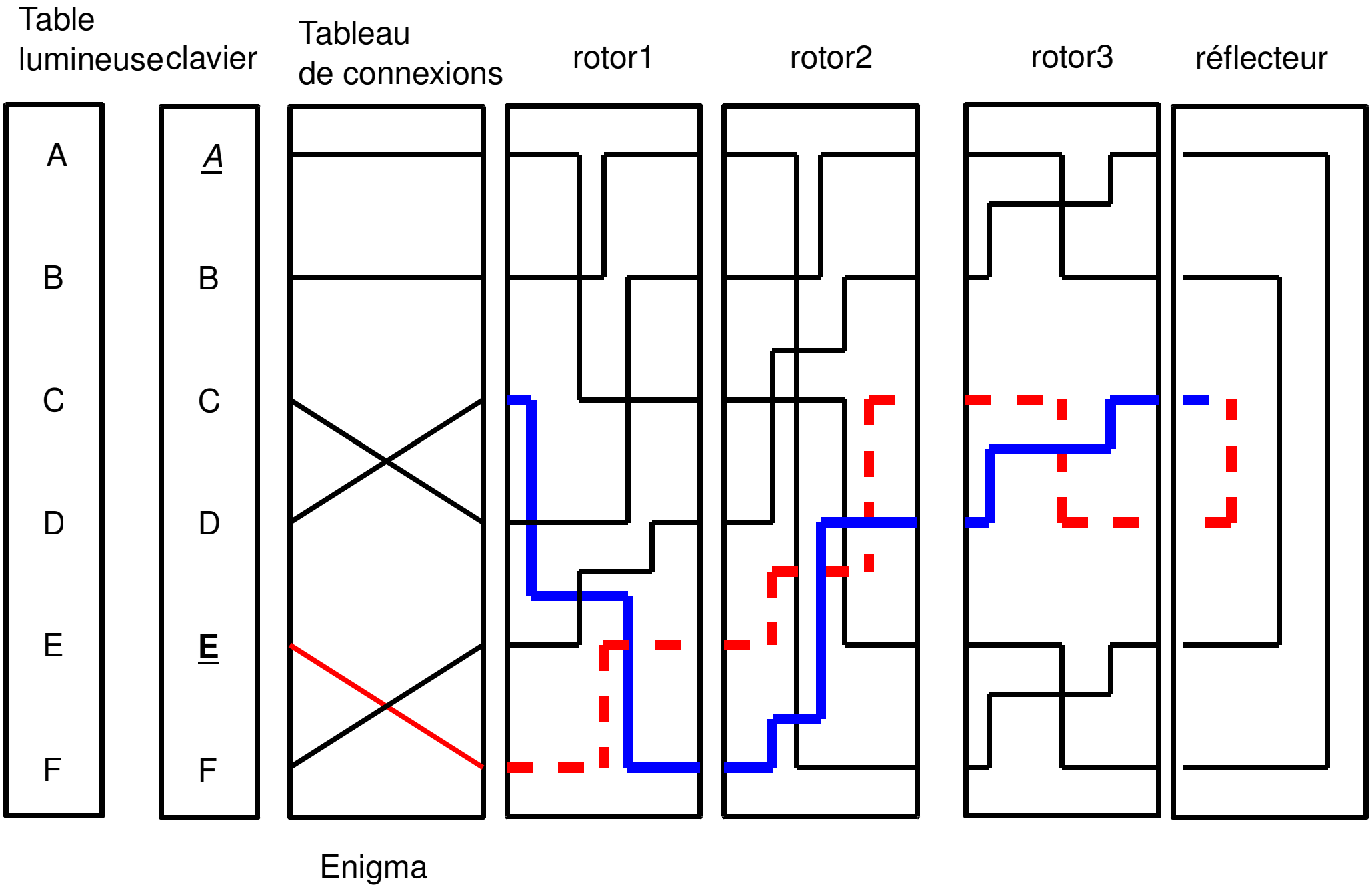


Table lumineuse
clavier
Tableau de connexions
rotor1
rotor2
rotor3
réflecteur
A
B
C
D
E
F
A
B
C
D
E
F
Enigma

Masque jetable

- Principe du tel. Rouge entre URSS et USA
- Soit M un message binaire=b1b2b3...bl (avec l le nombre de bits du message)
- Soit K=k1k2k3...kl une suite de bits impossibles à prédire (on utilise une suite tirée aléatoirement)
- Pour obtenir le message chiffré, on fait un où exclusif entre chaque bit du message et celui de la clef
  - $c_i = b_i$ XOR k_i

Masque jetable

- Table du XOR: 0 XOR 0=0, 1 XOR 0 = 0 XOR 1 = 1 et 1 XOR 1 = 0
- Avantages du masque jetable:
  - Simple à mettre en oeuvre, rapide
  - Sûr si la clef est imprévisible
- Défauts :
  - Transport de la clef: les deux extrémités doivent l'avoir mais pas l'espion. Le transfert sûr de la clef est aussi difficile que celui du message.
  - La génération de la clef qui ne peut être réutilisée d'un message à un autre

Masque jetable

- Connaissant le message et sa version chiffrée, on peut en déduire facilement la clef K
- C= M XOR K
- K = M XOR C
- Conséquence: il faut une nouvelle clef pour chaque message
- Méthode du masque jetable ou « One Time Pad »

Principes de Kerckhoffs

- Auguste Kerckhoffs (Hollande, 1883)
  - Seule une donnée de petite taille (clef) doit suffire à assurer la sécurité
  - La sécurité d'un mécanisme ne doit pas reposer sur son caractère secret
- Il est beaucoup plus facile de garder secret un clef connue d'une personne qu'un procédé mis au point par plusieurs personnes.

L'ère scientifique:

- 1976: Diffie et Hellman découvrent la crypographie à clef publique en mettant au point un algorithme d'échange de clefs ne supposant aucun échange de secret préalable
- 1977: DES (standard américain de chiffrement symétrique)
- 1978: Rivest, Shamir et Alderman (re)découvrent RSA (système dechiffrement à clef publique)

algorithme de chiffrement symétrique (à clefs privées)

- Chiffrement symétrique:
  - la même clef sert au chiffrement et au déchiffrement
  - C'est le principe du coffre fort:
    - Akira et Barack connaissent tous deux la combinaison du coffre fort :
      - Akira dépose un message dans le coffre fort en utilisant la combinaison : elle chiffre le message avec la clef partagée
      - Barack récupère le message en ouvrant le coffre avec la même combinaison: il déchiffre le message avec la clef partagée
      - Si une autre personne réussit à avoir la combinaison, il peut lui aussi ouvrir le coffre et lire ou déposer des messages: elle peut chiffrer de nouveaux messages ou déchiffrer les messages transmis

Chiffrement symétrique

- les algo classiques sont rapides
- Pb:
  - comme faire en sorte que Bob et Alice connaissent la clef ?
  - Problème loin d'être négligeable qui se coûtait des sommes importantes aux services secrets du monde entier pour diffuser et renouveler les clefs secrètes envoyées à leur agents :
    - Renouveler suffisamment souvent les clefs
    - Avoir une clefs spécifique pour chaque contexte/agent pour éviter les conséquences de la récupération d'une clef par l'ennemi

algorithme de chiffrement

- chiffrement asymétrique:
  - Chaque participant x
    - a une clef publique Px qu'il peut diffuser à tous
    - A une clef secrète Qx qu'il doit être seul à connaître
  - Le chiffrement se fait avec la clef publique
  - Le déchiffrement se fait avec la clef privée
  - Principe de la boîte aux lettres :
    - Alice peut déposer des messages dans la boîte aux lettres: elle chiffre un message avec la clef publique
    - Bob est le seul à avoir la clef de la boîte aux lettre et donc le seul à pouvoir lire les messages qui y ont été déposés: il est seul à pouvoir déchiffrer des messages avec la clef privée.
  - Exemple: si Alice veut envoyer un message à Bob
    - Elle chiffre le message avec la clef publique de Bob
    - Bob reçoit le message chiffré et le déchiffre avec sa clef privée
  - les algo classiques sont lents

algorithmes classiques

- symétriques:
  - DES (1976): standard américain (1977), clef de 56 bits sur des blocs de 64 bits. dépassé de nos jours.
  - triple DES (1978): variante, triple application de DES, clefs entre 128 et 192 bits sur des blocs de 64 bits.
  - RC2, RC4, RC5 (1994) et RC6:
  - IDEA (1992): clef 128 bits sur des blocs de 64 bits
  - blowfish: clef 32 à 448 bits sur des blocs de 64 bits. Algo très analysé,considéré comme solide. utilisation libre.
  - AES (1998): clefs 128, 192 ou 256 bits sur blocs de 128 bits. standard américain. utilisation libre.

algorithmes classiques

- asymétriques:
  - RSA s'appuyant sur la factorisation de nombres premiers
  - Diffie-Hellman et El Gamal s'appuyant sur le calcul des logarithmiques discrets
  - des algorithmes nouveaux s'appuyant sur les courbes elliptiques

Système hybrides

- On génère une clef de session Ks
- On utilise un chiffrement à clef publique pour la transmettre à son correspondant
- On chiffre le reste de la communication avec un algorithme symétrique (donc rapide) utilisant la clef Ks

L'espion

- L'espion est un personnage classique de la cryptographie. Le but initial de la cryptopgraphie était de protéger des communications de ses actions :
  - Il peut vouloir écouter et déchiffrer un message
  - Il peut vouloir déchiffrer tous les messages échangés entre Ahmed et Bernard
  - Il peut vouloir modifier les messages entre Arthur et Bérénice (on parle alors d'écoute active)

L'espion (2)

- Il peut se faire passer pour Anas auprès de Bérénice: on parle d'usurpation d'identité
- Il peut se faire passer pour Alex auprès de Bertrand et pour Bertrand auprès d'Alex : attaque « Man In the Middle »
- Bertrand peut refuser de reconnaître être l'auteur d'un message (reconnaissance de dette, ...) qu'il a pourtant envoyé à Aïcha.

Services s'appuyant sur de la cryptographie

- Confidentialité
- Intégrité
- Authentification:
- Non répudiation

Confidentialité

- protection des données contre une divulgation non autorisée
- 2 moyens techniques complémentaires
  - protéger l'accès aux données (implique authentification et contrôle d'accès). Ex.: authentification windows + ACL NTFS
  - les chiffrer
- intégrité, confidentialité : des contraintes opposées
  - intégrité : multiplier les sauvegardes notamment hors site
  - confidentialité: limiter les lieux de stockage pour faciliter le contrôle d'accès

Intégrité

- certifier que les données n'ont pas été altérées de façon intentionnelle ou accidentelle
- la modification peut avoir lieu
  - lors du transfert des données (corruption, écoute active)
  - lors du stockage des données
  - lors de leur traitement (bogues des logiciels applicatifs, des OS).
- Implications:
  - légales, plantage des applications et perte d'activité
  - perte d'image

Identification et authentification

- **identification**: définir l'identité de l'utilisateur
- **authentification**: permet de vérifier l'identité fournie (authentification simple vs authentification forte)
  - via un élément que l'utilisateur connait (mot de passe, ...)
  - via un élément que l'utilisateur possède (carte à puce, certificat, ...)
  - via biometrie

authentification

- élément clef pour assurer :
  - la confidentialité et l'intégrité des données via un contrôle d'accès: seules les personnes identifiées, authentifiées et habilités à le faire peuvent accèder/modifier les données
  - la non-répudiation et l'imputabilité (preuve d'une transaction, ...)
- Authentification unique (SSO: Single Sign On)
  - l'utilisateur s'authentifie une fois
  - il a accès à toutes les ressources du réseau
  - cf partie technique (keberos, ...)

non répudiation

- **non répudiation** : ne pouvoir nier qu'un événement a eu lieu
- **imputabilité**: on sait qui a réalisé une action
- **traçabilité**: mémoriser des événements imputables
- **auditabilité**: pouvoir réaliser une analyse ultérieure d'un événement. Ex.: en cas d'intrusion.
- moyens: utilisation de journaux
  - de taille limitée
  - éventuellement hors site (intrusion)

Signature électronique

- Définition: la *signature électronique* a pour objectif de permettre à une personne d'attacher son identité à un message.
- Problèmes:
  - Le message doit être protégé contre les modifications sinon que certifie-t-on en le signant ?
  - La signature ne doit pas pouvoir être utilisée pour signer un autre message

Signature électronique: exemple à ne pas suivre

- Supposons que signer se résume à ajouter son nom à la fin d'un fichier
  - Tout le monde peut imiter une telle signature (pb de non répudiation)
  - Le document signé peut-être modifié (ajoutons un zéro sur la somme citée dans un chèque et changeons le bénéficiaire)
  - Ajouter de l'information à la fin d'un fichier peut le corrompre

Signature:

- Elle se décompose en deux parties:
  - Un procédé de signature: permet d'obtenir les données signées.
    - Ne doit pouvoir être appliqué que par le signataire.
    - Utilise en général un secret détenu par le signataire
  - Un procédé de vérification: à partir d'un texte signé et de l'identité du signataire, le procédé de vérification doit confirmer que le texte signé a bien été signé par le signataire désigné
    - doit pouvoir se faire sans secret détenu par le signataire.
- Signature et confidentialité sont deux services distincts qui sont parfois intégrés dans un même système.

Chiffrement symétriques

- clef de chiffrement = clef de déchiffrement
- Chiffrements par bloc:
  - DES (1977), blocs de 64 bits, clefs de 56 bits
  - IDEA (Lai Massey 1991), blocs de 64 bits, clefs de 128 bits
  - Rinjdael (Rinmen, Daemen, 1997): blocs de 128 ou 256 bits, clefs de 128, 192 ou 256 bits
  - AES, blocs de 128 bits, clefs de 128, 256 bits
- Chiffrement en continu d'un flux
  - RC4: chiffrement octets par octets
  - Pseudo-Vernam: XOR entre le flux et la sortie d'un générateur aléatoire

Principes de conception

- Problème :
  - On connaît de nombreux systèmes faibles (cesar, substitution, vigenere, ...)
  - comment construire des systèmes sûrs et plus efficaces que le masque jetable

Principes de conception

- Claude Shannon, 1949, « Communication Theory of Secrecy Systems »
  - Confusion: la relation entre le texte clair et le message chiffré doit être impossible à établir. Ca doit notamment être le cas pour les propriétés statistiques de l'un et de l'autre
  - Diffusion: un bit de clef ou de texte clair doit influencer de nombreux bits du texte chiffré.

Chiffrement symétrique : modes de chiffrement

- Chiffrement par bloc:
  - L'algo de chiffrement est capable de chiffrer des blocs de taille fixe
  - découper le message en bloc et chiffrer chaque bloc
  - Plusieurs modes opératoires peuvent être employées (voir plus loin: ECB, CBC, ...)
- Chiffrement en continu
  - On chiffre le texte en clair au fur et à mesure qu'il se présente
  - Utile dans le cadre d'application réseau
  - Une solution: utiliser la technique du masque jetable avec des masques précalculés ou calculés au vol à partir d'une clef de base

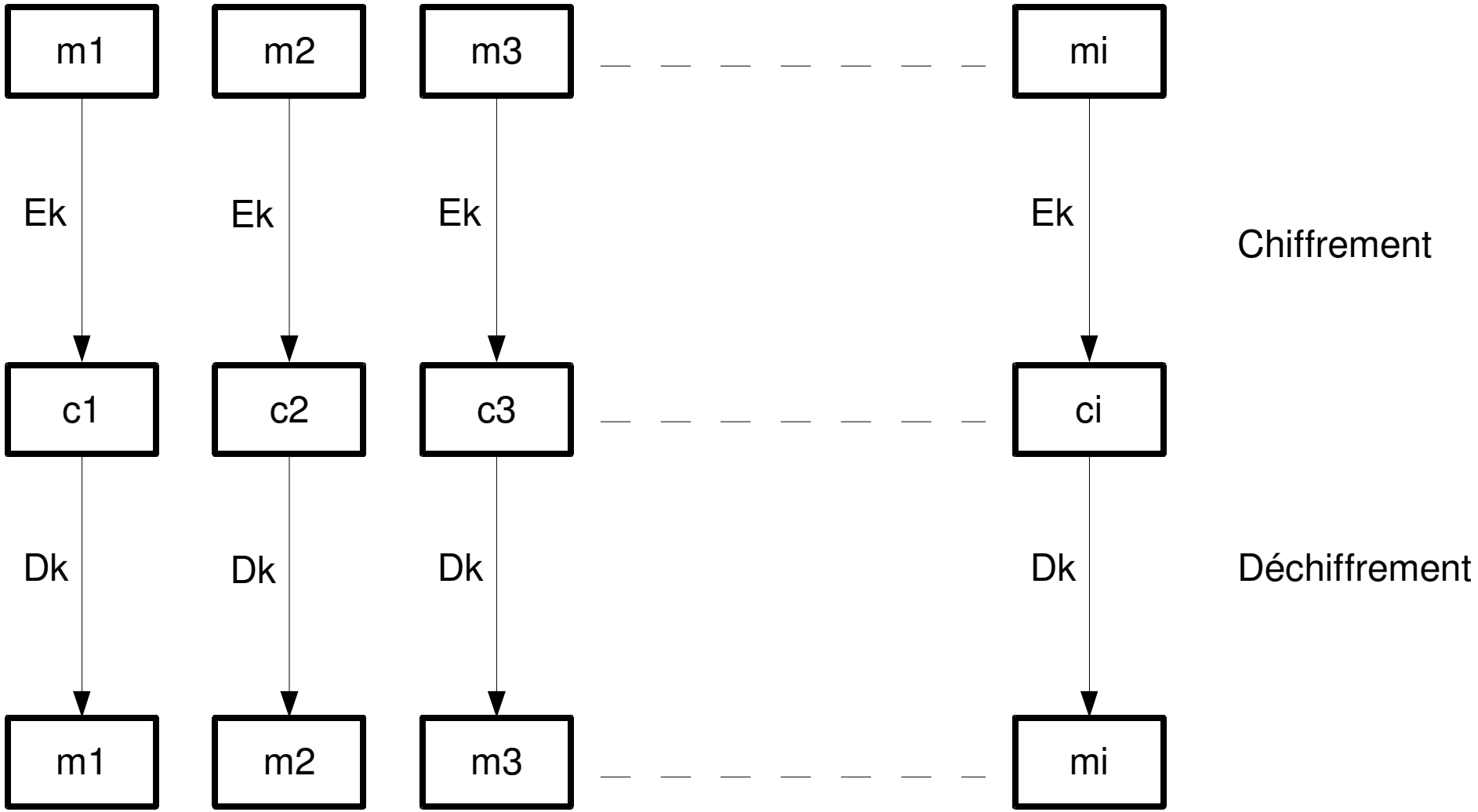
Chiffrement par bloc: ECB

- Electronic code Book (ECB) : on chiffre chaque bloc indépendamment des autres
  - Soit m le message en clair: $m=m_1 m2_{...} m_p$
  - c la version chiffrée de m: $c=c_1 c_{2...} c_p$
  - E une fonction de chiffrement capable de chiffrer des blocs, D la fonction de déchiffrement associée
  - K une clef de chiffrement pour E, déchiffrement pour D
- ECB:
  - Chiffrer: $c_1=E(m_1, k)$, $c_2=E(m_2, k)$, ..., $c_p=E(m_p, k)$
  - Déchiffrer: $m_1=D(c_1)$, $m_2=D(c_2)$, ..., $m_p=D(c_p, k)$

Chiffrement par bloc: ECB

- ECB:
  - Chiffrer: $c_1=E(m_1, k)$, $c_2=E(m_2, k)$, ..., $c_p=E(m_p, k)$
  - Déchiffrer: $m_1=D(c_1)$, $m_2=D(c_2)$, ..., $m_p=D(c_p, k)$
- On suppose que c2 est verollé: c'2. Quelle conséquence sur le déchiffrement :
  - m1=D(c1) : OK; D(c'2) ne donne **m2**
  - m3=D(c3) : OK, m4=D(c4): OK

Chiffrement par bloc: ECB



Chiffrement par bloc: ECB

- Electronic code Book (ECB) : on chiffre chaque bloc indépendamment des autres
  - Avantage:
    - Simple
    - On peut modifier un morceau des données sans modifier toutes la version chiffrée
    - On peut précalculer les versions chiffrées des blocs pour gagner en vitesse

Chiffrement par bloc: ECB

- Electronic code Book (ECB) : on chiffre chaque bloc indépendamment des autres
  - Défaut: mauvaise sécurité
    - Un bloc sera tout le temps chiffré de la même manière à l'intérieur d'un message ou dans des messages différents avec la même clef
    - L'espion peut se faire un dictionnaire s'il arrive à obtenir les versions chiffrées et en clair de certains message
  - Amélioration:
    - Ajouter un texte aléatoire avant chaque bloc à chiffrer
    - Le retirer au déchiffrement
    - Défaut: on augmenter la taille des données à chiffrer ce qui peut poser des problèmes de performances

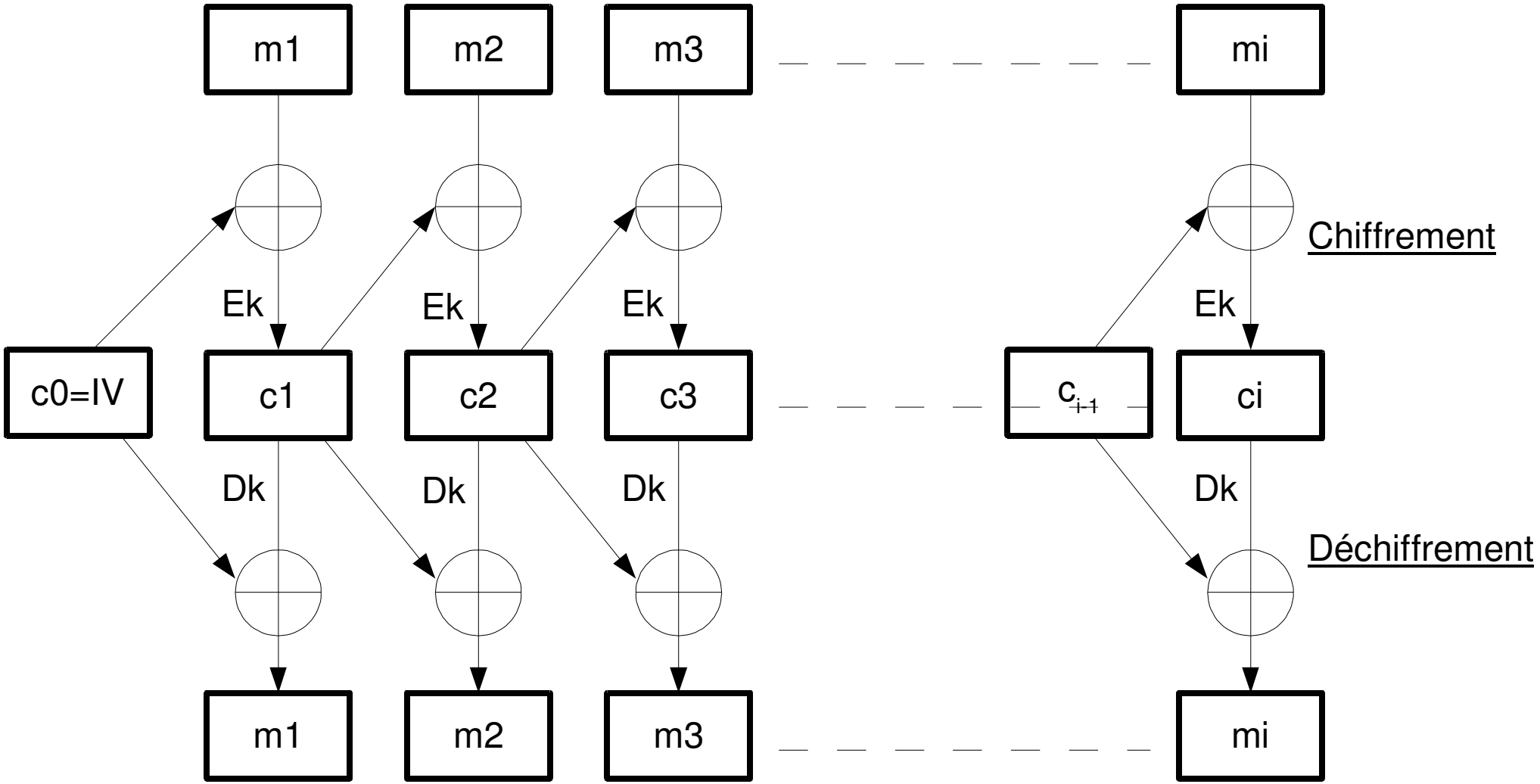
Chiffrement par bloc: CBC

- Cypher Bloc Chaining : les blocs sont chiffrés en fonction les uns des autres.
  - En mode CBC: le texte en clair est combiné par un XOR avec la version chiffrée du bloc précédent
  - 2 messages avec le même début et chiffrés avec la même ont leur version chiffrée qui commence pareil :
    - Solution : On ajoute un premier bloc aléatoire IV (vecteur d'initialisation)Cet IV est transmis en clair.
  - Avantage:
    - sûr
  - Défaut:
    - Une erreur rend illisible toute la suite des données
    - Peformance: Non parallélisable

Chiffrement par bloc: CBC

- CBC :
  - Chiffrer:
    - c_0=IV,
    - c_1=E($c_0 \oplus m_1$,k),
    - c_2=E($c_1 \oplus m_2$,k), ...,
    - c_p=E($c_{p-1} \oplus m_p$,k)
  - Déchiffrer:
    - c_0=IV,
    - m_1=$c_0 \oplus$D(c_1,k),
    - m_2=$c_1 \oplus$D(c_2,k), ...,
    - m_p=$c_{p-1} \oplus$D(m_p,k)

Chiffrement par bloc: CBC



Chiffrement par bloc: OFB

- Mode OFB (Output Feedback) ou à rétroaction de sortie
  - E: chiffrement de blocs de longueur n
  - $1 \leq r \leq n$
  - Un vecteur d'initialisation IV
  - On découpe les données à chiffrer en blocs de taille r
- Chiffrement:
  - I_1=IV
  - O_j=E(I_j,k)
  - t_j les r premiers bits de O_j
  - c_j= $m_j \oplus t_j$
  - I_{j+1}=O_j

Déchiffrement:

I_1=IV

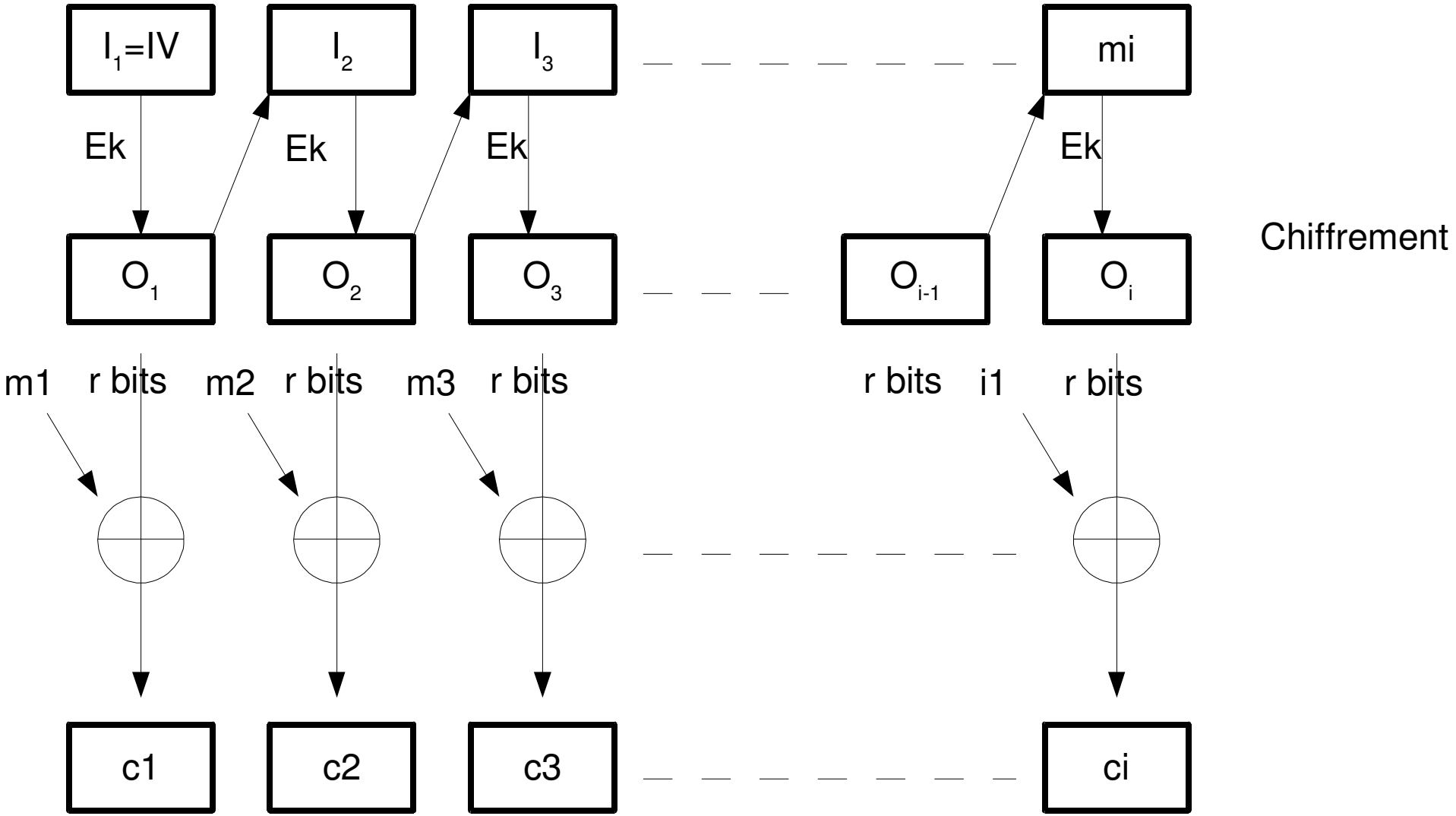
O_j=E(I_j,k)

t_j les r premiers bits de O_j

m_j= $c_j \oplus t_j$

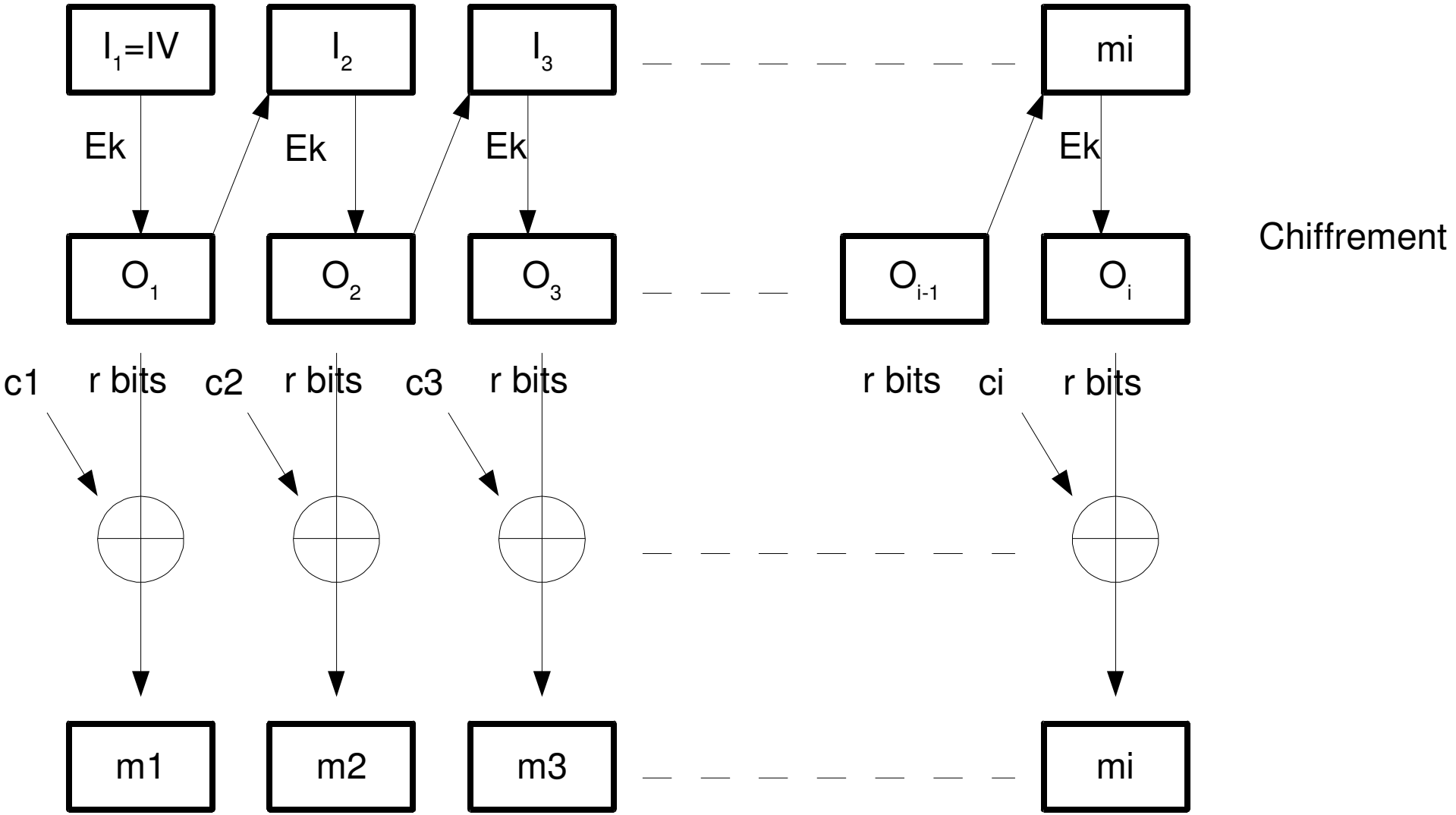
I_{j+1}=O_j

Chiffrement par bloc: OFB



Chiffrement

Déchiffrement par bloc: OFB



Même algorithme que pour le chiffrement

Chiffrement par bloc: OFB

- Avantages:
  - Simplicité
  - Dès qu'on a l'IV, on peut précalculer les O_j ce qui peut être pratique d'un point de vue performances dans certains contextes
  - Deux blocs identiques d'un même fichier seront chiffrés de façon différentes
  - En cas d'erreur, seul le bit erroné sera faux
- Défauts:
  - Le chiffrement d'un bloc en clair dépend seulement de sa position ce qui est plus faible qu'avec cbc où il dépendait du bloc chiffré précédent

Résistance aux erreurs de transmission

- Si un bloc chiffré est détérioré lors des transmissions (un méfait habituel du troll des réseaux), on montrera en TD que :
  - Méthode ECB: Un bloc déchiffré sera verollé
  - Méthode CBC: deux blocs déchiffrés seront verollés
  - Méthode OFB: Un bloc déchiffré sera verollé

Chiffrement par bloc: taille des blocs

- Paradoxe des anniversaires (Richard von Mises):
  - Combien de personnes doit-on rassembler pour avoir plus d'une chance sur deux que deux personnes de ce groupe aient leur anniversaire le même jour de l'année.
  - Réponse: 23, ce qui choque un peu l'intuition.
  - À partir d'un groupe de 57 personnes, la probabilité est supérieure à 99 %

Chiffrement par bloc: taille des blocs

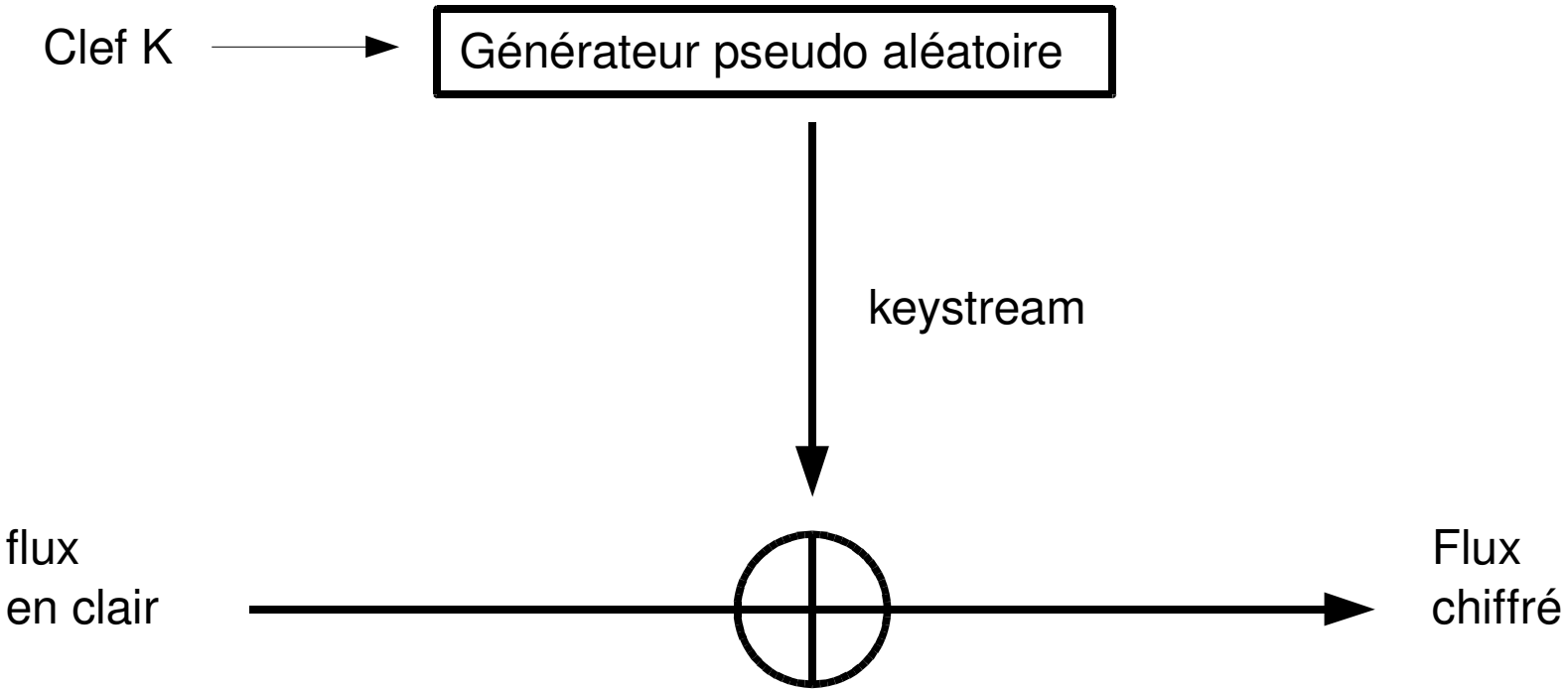
- Produire des blocs chiffrés identiques ouvre la porte à certaines attaques
- Avec des blocs de 64 bits, il faut 2^{32} (# 32Go) blocs distincts pour trouver une collision avec une chance sur deux
  - De nos jours, c'est trop peu
  - Si le mode opératoire est faible (mauvais aléa), la quantité nécessaire peut baisser
- AES utilise des blocs de 128 bits
  - Il faut alors 2^{64} blocs distincts, soit 256 exaoctets pour avoir une chance sur deux de trouver une collision

Chiffrement en continu ou par flot

- Les données sont traitées en flux
  - On commence à chiffrer
    - Sans avoir lu l'intégralité du message
    - Sans avoir sa longueur
- Exemples de chiffrement par flot
  - Masque jetable
  - RC4 (SSL, WiFi)
  - E0/1 (Bluetooth)
  - A5/1, A5/2, A5/3 (GSM)

Chiffrement en continu ou par flot

- On combine
  - Un peuso aléa (keystream ou « flux de clef »)
  - Avec le flux de données
- La combinaison est souvent un simple XOR



Chiffrement par flot

- Les chiffrements par flot sont
  - Très rapides
  - Adaptés aux application temps réel
- Problèmes
  - Propagation d'erreur en cas de problème de synchronisation
  - La sécurité repose sur la qualité du générateur pseudo aléatoire
  - Sécurité non prouvable

Masque jetable

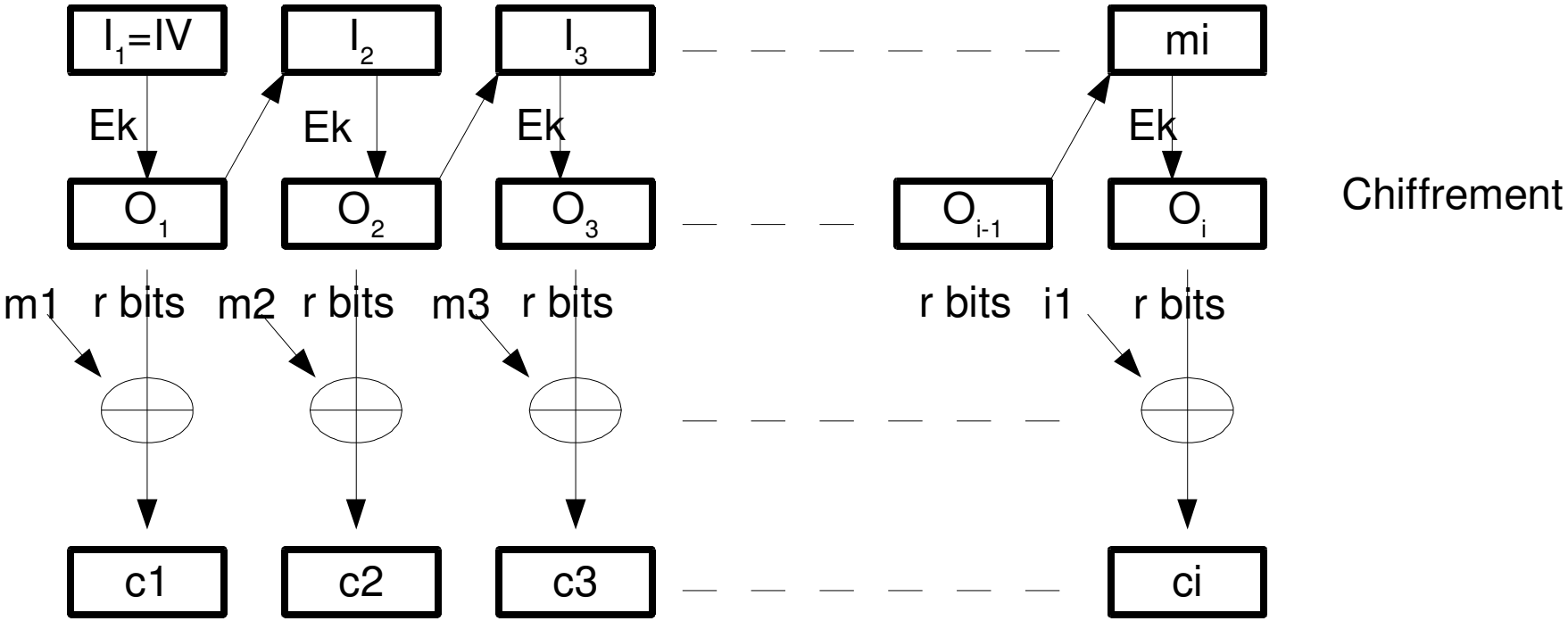
- Masque jetable:
  - La clef est une suite aléatoire parfaite
  - La clef est aussi longue que le message
  - La clef n'est jamais réutilisés
  - La sûreté est prouvée
- Chiffrement par flot
  - S'inspire du masque jetable
  - Mais
    - pseudo aléa généré à partir d'une valeur clef
    - Problème de réutilisation de la suite pseudo aléatoire si la clef ne change pas
      - => utilisation d'un valeur d'initialisation IV

Chiffrement par flot: pseudo aléa

- Il doit être impossible à disintiguer d'un véritable aléa
  - Equilibre: probabilité de ½ d'avoir 0 ou 1 en sortie
  - Indépendance: chaque bit de sortie est indépendant de ce qui précède et de ce qui suit
- Difficile à satisfaire
  - De nombreuses failles de grande ampleur ont eu comme source un mauvais aléa (ssl debian, netscape, ...)
- Exemple:
  - 414213; 732050; 236067; 645751
  - La suite paraît raisonnable mais c'est int(sqrt(2i+1))

Chiffrement par flot

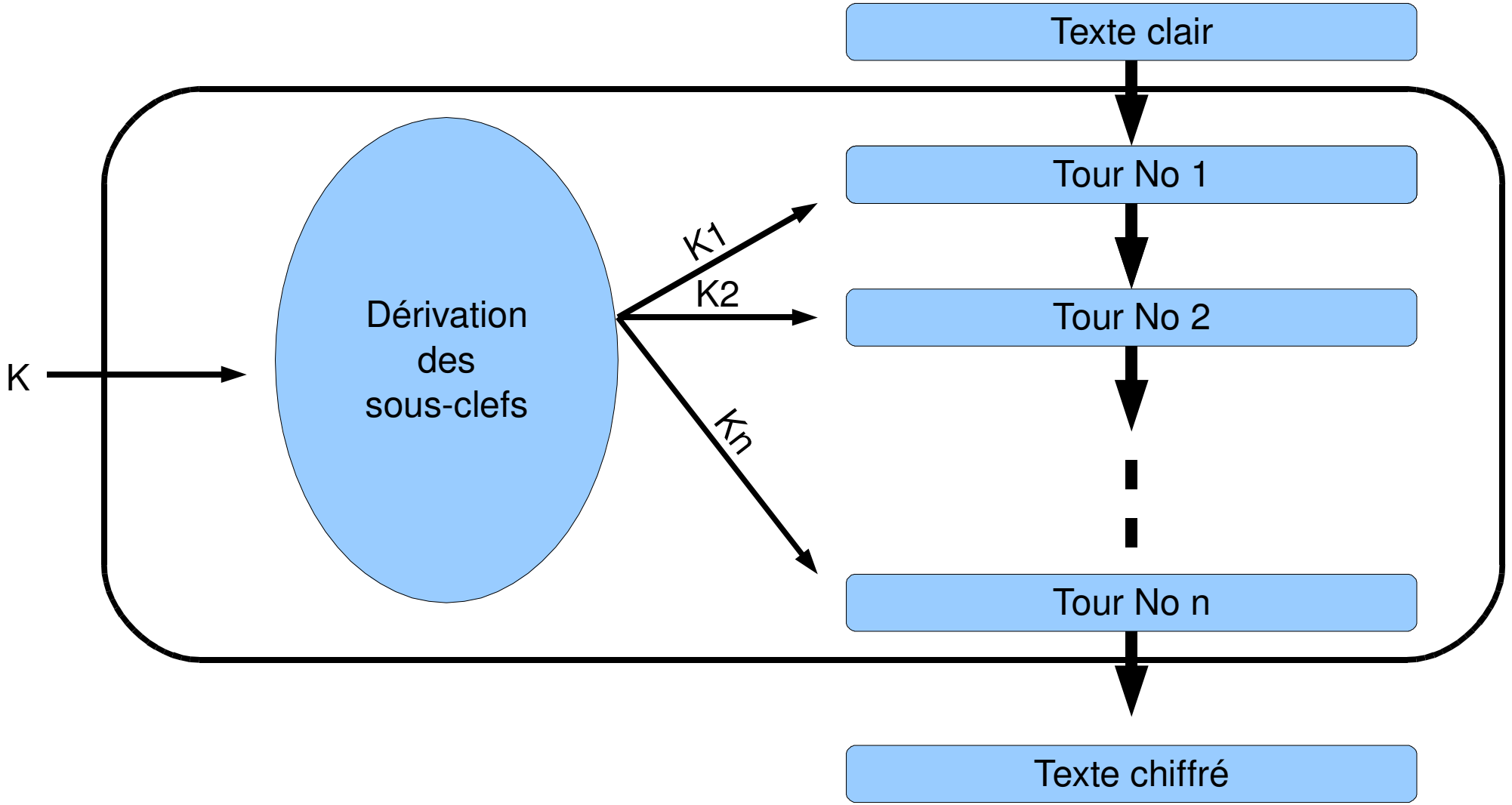
- Certains modes opératoires des chiffrements par blocs permettent leur utilisation comme chiffrement par flot. Exemple: OFB



Construction des chiffrements par bloc

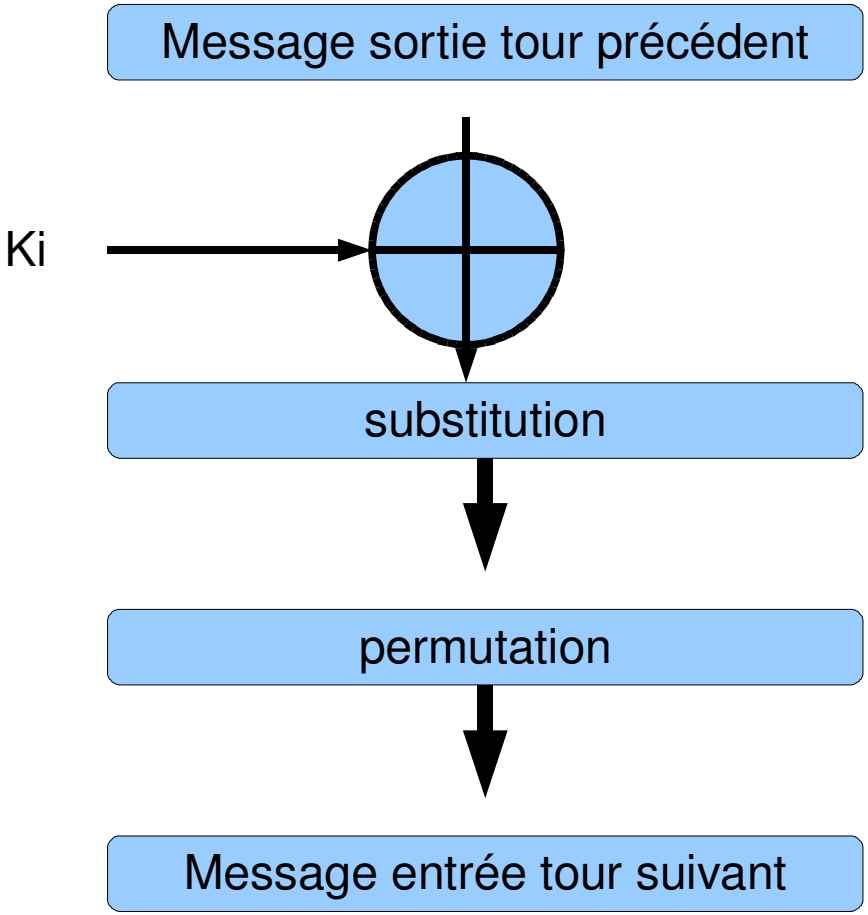
- Idée: itérer suffisamment de fois des opérations simples ***judicieusement choisies*** permet d'avoir une excellente sécurité
- Le chiffrement est une succession d'étapes similaires (appelées « tour ») utilisant chacun une sous-clef
  - Construction itérative et modulaire
  - Les sous-clefs sont dérivées de la clef secrète
  - Les fonctions utilisées par un tour doivent être optimisées et sont en général des opérations simples

Construction des chiffrements par bloc



Méthode par substitutions/permutations

- Décomposition d'un tour:
  - Utilisation de la clef Ki
  - Fonction de substitution
  - Fonction de permutation

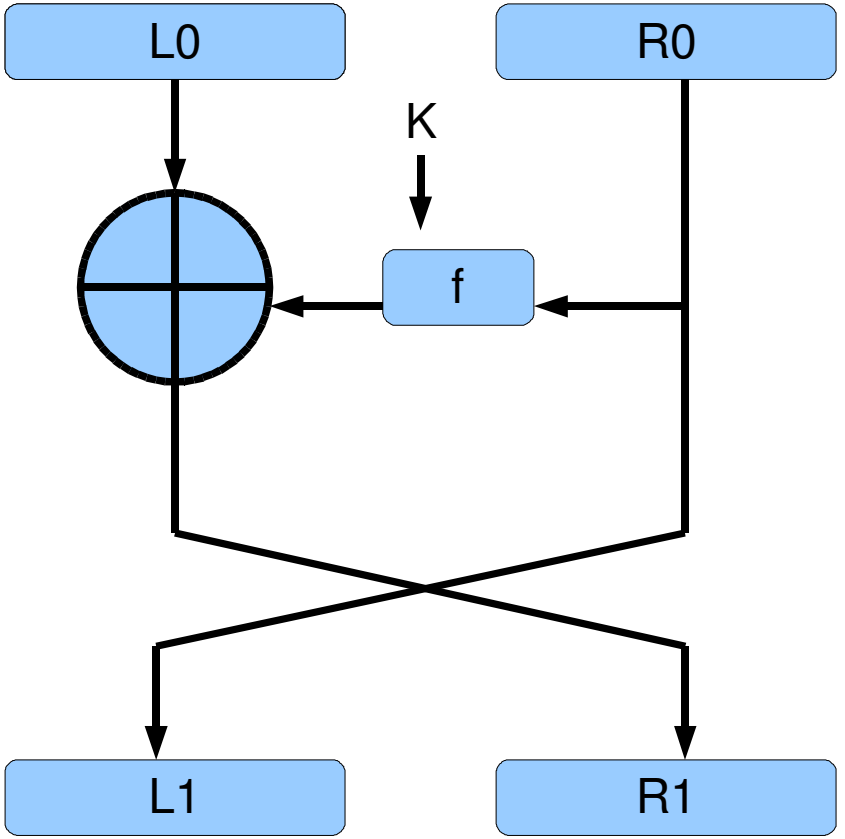


Substitution: un symbole du texte clair est remplacé par un autre symbole. Cf tables de substitution du DES (S-Boxes). Ajoute de la confusion.
permutation: on échange entre eux les symboles du texte en clair. Ajoute de la diffusion

Schéma de Feistel

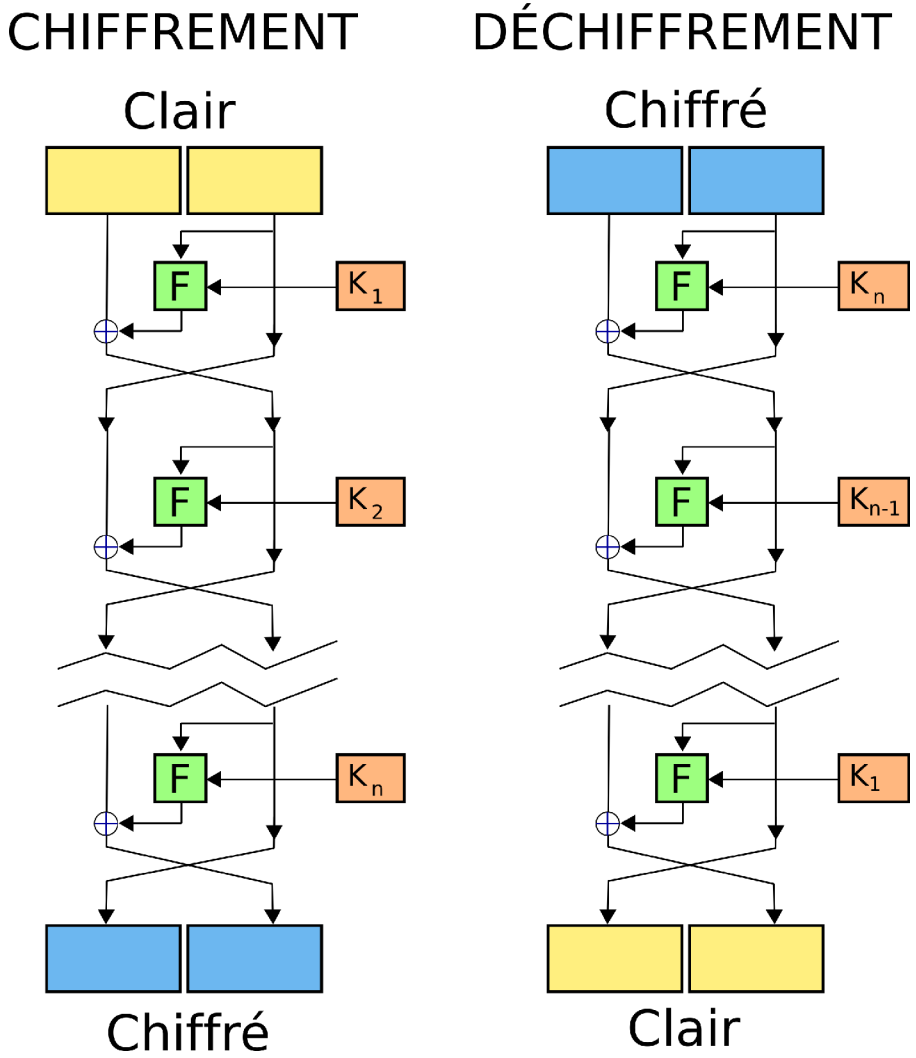
- Chiffrement:
  - L1=R0
  - R1=L0 ⊕ f(R0, K)
- Déchiffrement:
  - R0=L1
  - L0=R1⊕f(R0, K)

La fonction f est appelée fonction de confusion



Réseau de Feistel

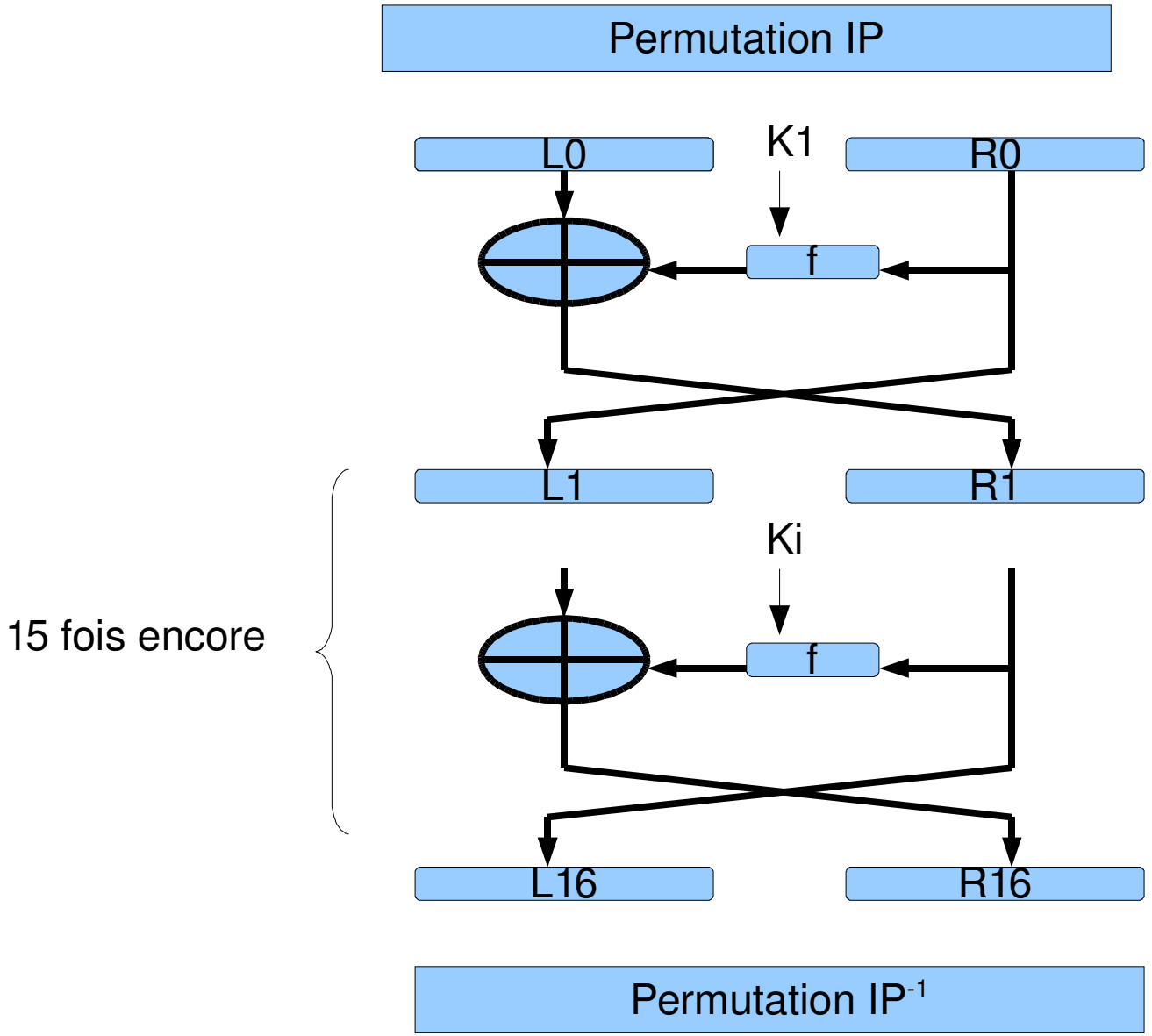
- Constitué de plusieurs tours (ou rondes) ou étages de Feistel
- Chiffrement et déchiffrement sont efficaces si chaque tour l'est
- La complexité du décryptage croit de façon exponentielle avec le nombre de tour



DES: Data Encryption Standard

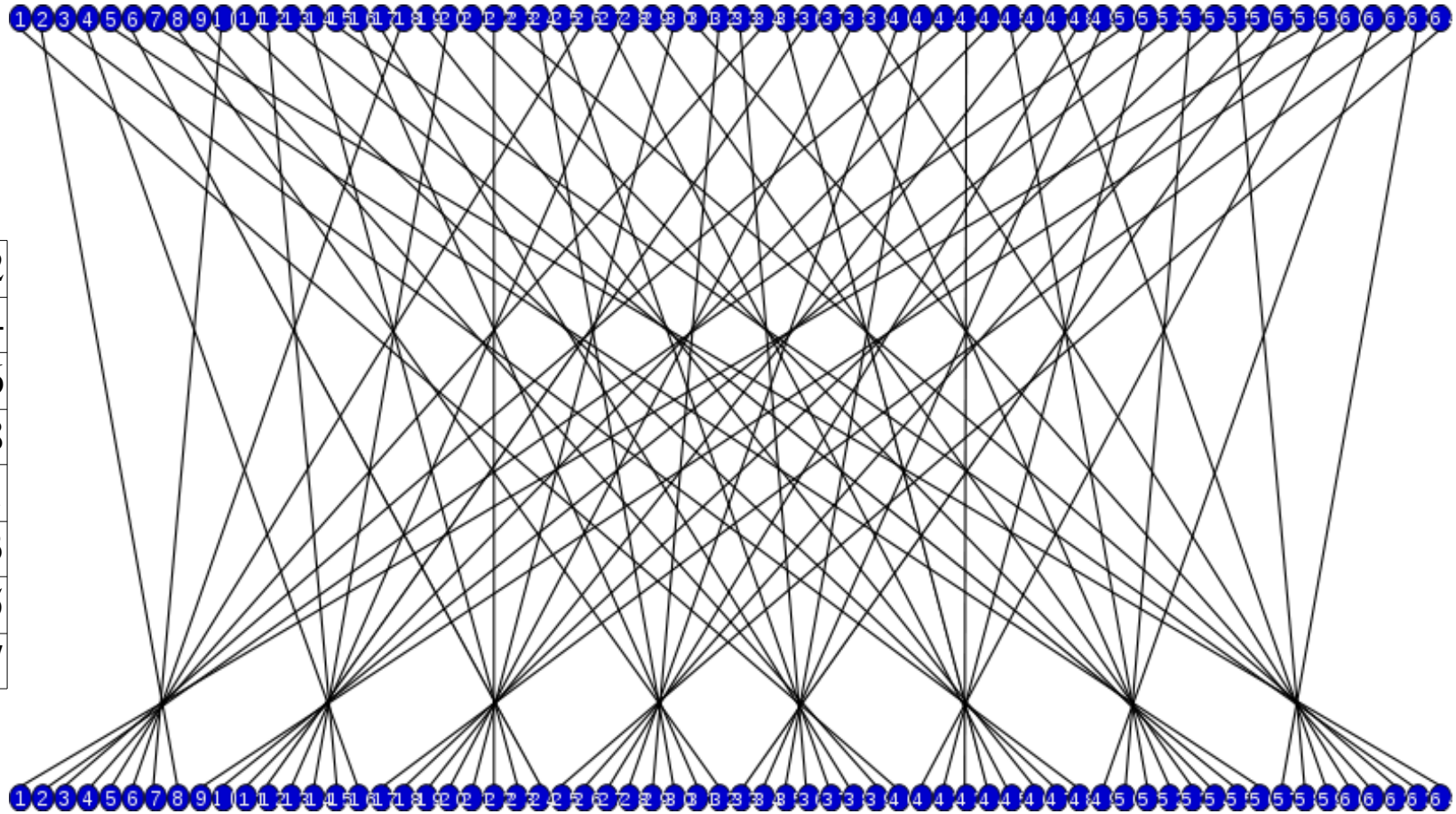
- Propriétés:
  - Algorithme à clef privée
  - Chiffrement par blocs de 64 bits
  - Feistel 16 tours avec des sous-clefs de 48 bits
  - Conçu dans les années 1970
  - Considéré comme obsolète de nos jours
- Conception:
  - À partir de l'algorithme Lucifer (IBM, 1970)
  - Corrigé par la NSA :
    - Clefs de 56 bits au lieu de 64
    - Boîtes S corrigées
    - Les but de la NSA ne seront découverts qu'en 1990
  - Standard FIPS 46-2 en 1977

DES: schéma


Permutation IP
L0
K1
R0
f
L1
R1
15 fois encore
Ki
f
L16
R16
Permutation IP-1

DES: permutation initiale IP

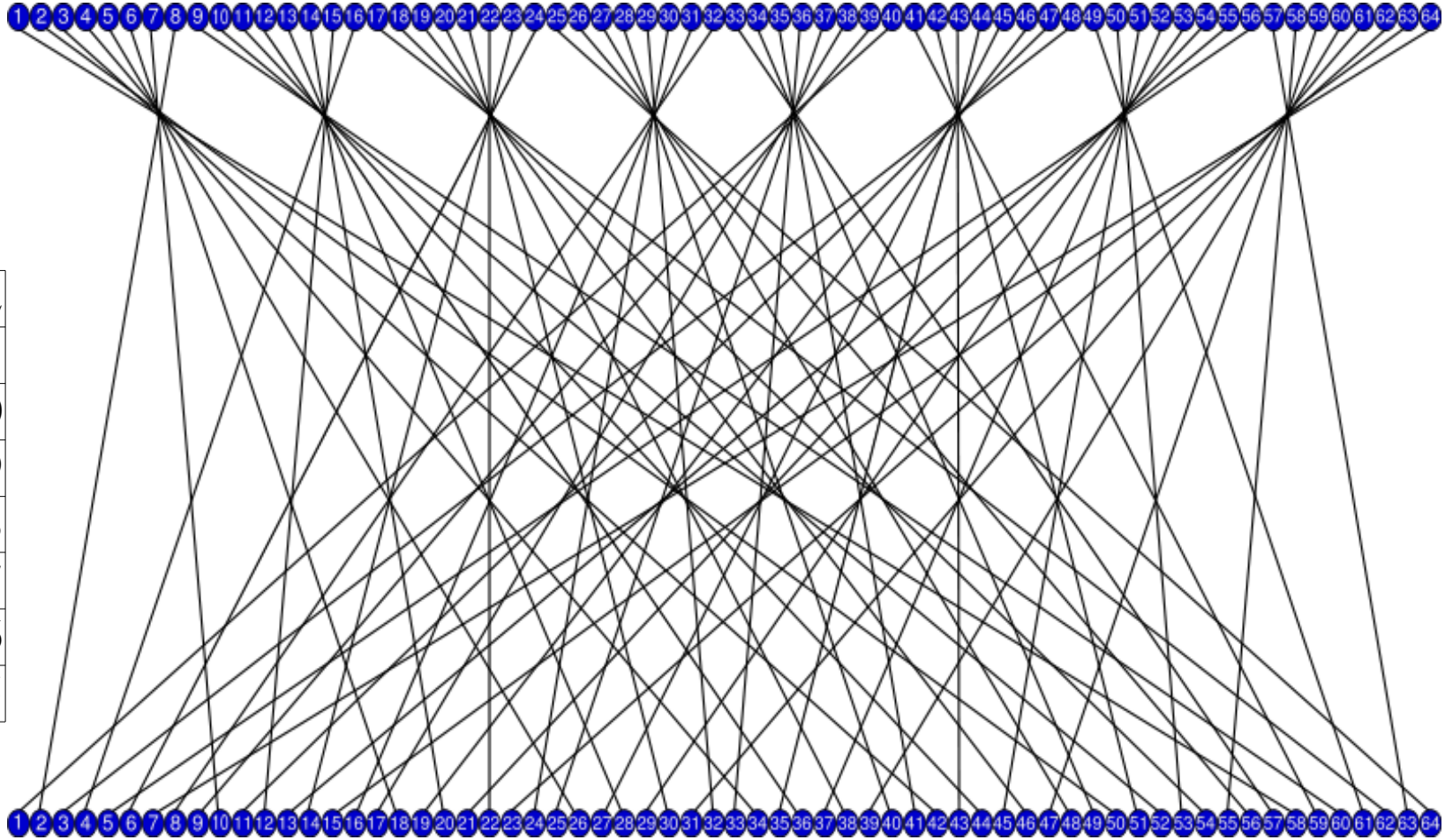
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7



Le 58e bit devient le premier, le 50e bit devient le deuxième, ..le 1er bit devient le 40e
si m=p1p2...p64 alors P(m)=p58p50...p7

DES: permutation finale

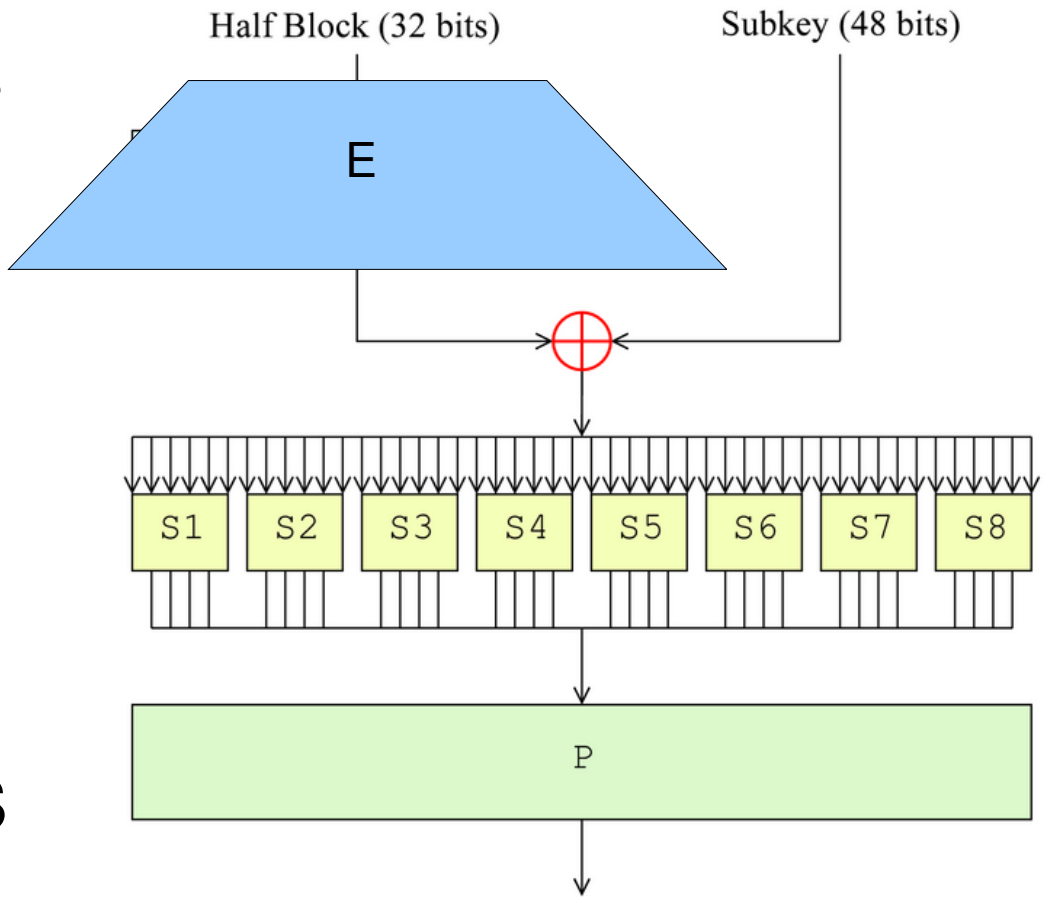
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25



Le 40e bit devient le premier, le 8e bit devient le deuxième, ...le deuxième bit devient le 50e, ... cette permutation est l'inverse de la première.

DES: fonction f

- La fonction f est une fonction de 32 bits vers 32 bits constituée:
  - d'une expansion de Ri de 32 bits vers 48 bits
  - d'un XOR avec 48 bits dérivés de la clef
  - de l'application de 8 sous-fonctionsde 6 bits vers 4 bits : les boîtes S
  - d'une permutation des 32 bits sortants

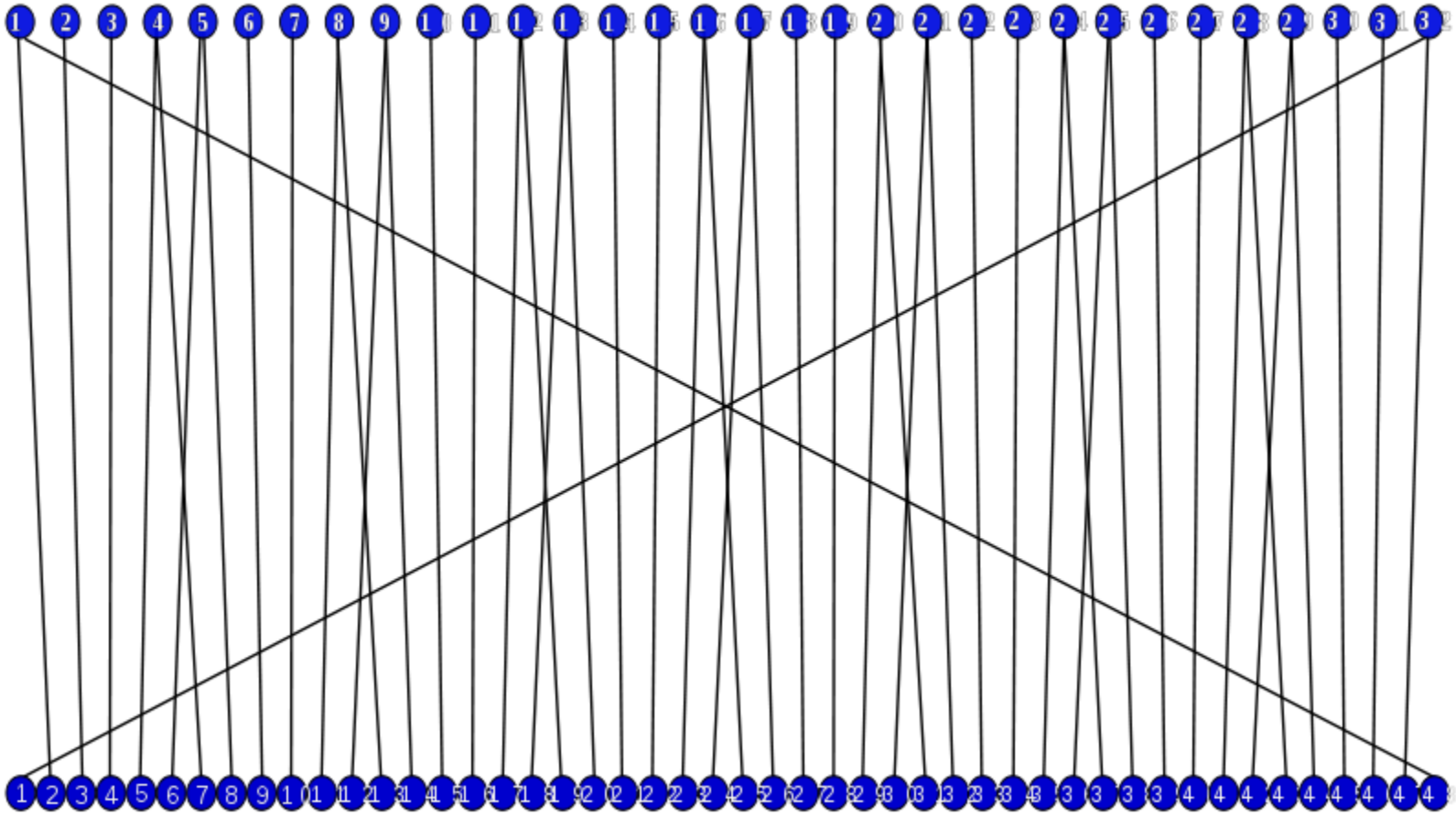


DES: fonction f

- La fonction f n'est pas inversible (elle n'a pas besoin de l'être)
- L'expansion duplique certains bits
- Les boîtes assurent la non linéarité (6 bits -> 4 bits pour chaque boîte)
- La permutation renforce la diffusion (tout comme le fait le croisement du schéma de Feistel)

DES: expansion E

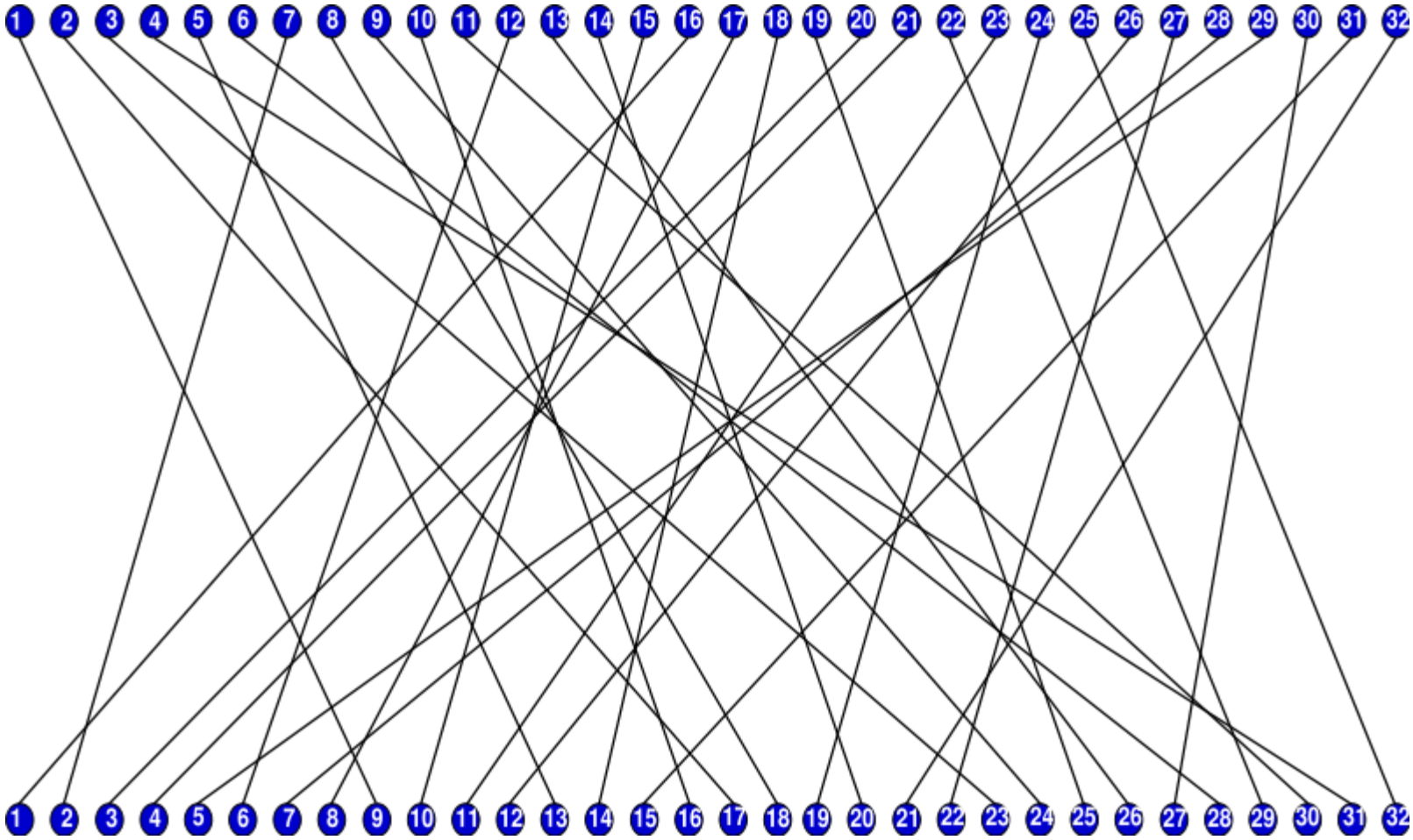
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1



Le 32e bit devient le premier, le 1er bit devient le 2e, ...

DES: permutation P

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25



Le 16e bit devient le 1er, le 7e bit devient le 2e, ...

DES: boîtes de substitutions

Utilisation des boîtes-S:

si B=b1b2b3b4b5b6

s(B) est le mot binaire situé à la ligne b1b6 et à la colonne b2b3b4b5.

La numérotation des lignes et des colonnes commence à 0.

Exemple: S1(<u>1</u>11001<u>1</u>):

- Ligne 11=ligne3
- Colonne 1100=colonne 12
- S1(111001)=10=1010

S1															
14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S2															
15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S3															
10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S4															
7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

DES: boîtes de substitutions

S5															
2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S6															
12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

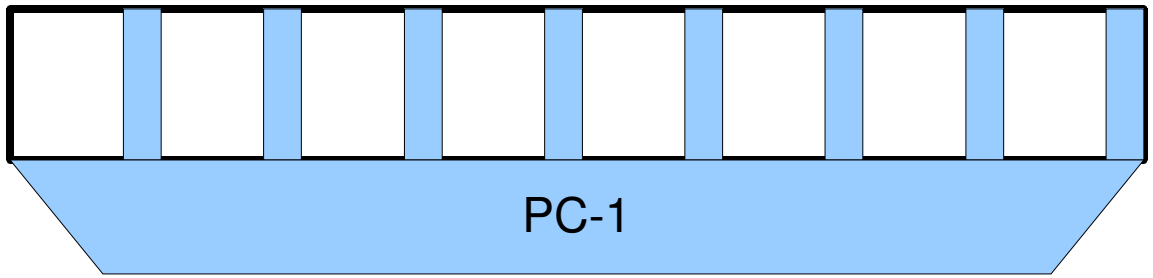
S7															
4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S8															
13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

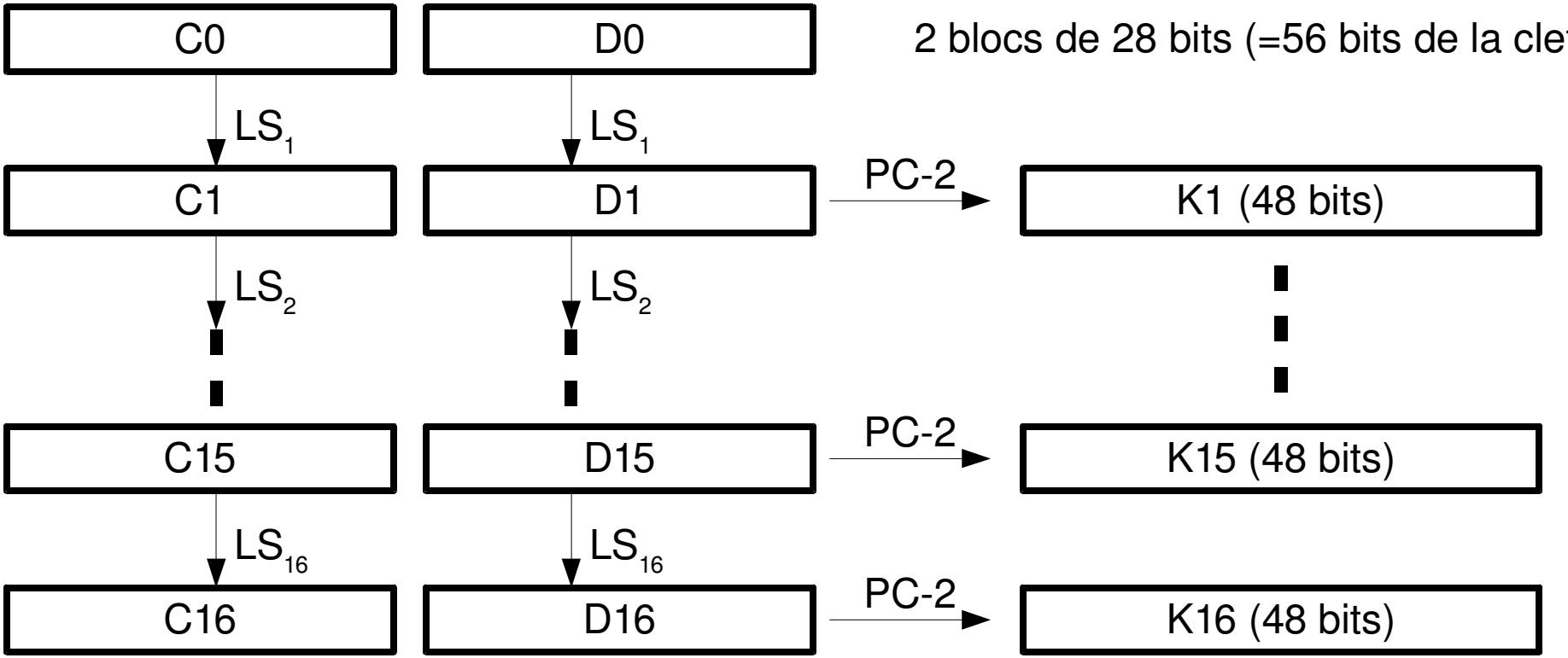
DES: génération des clefs des rondes

- Décalage: LSI : 28 bits -> 28 bits
  - $LS_I(C)$ décalage vers la gauche de V_i bits
  - V_i=1 si i =1,2,9 ou 16
  - V_i=2 sinon
- PC1: une fonction qui retourne 2 mots de 28 bits à partir d'un mot de 64 bits
  - Elle supprime les bits de parité (8, 16, ..., 64)
  - On obtient la clef de 56 bits en 2 mots de 28 bits
- PC2:
  - 28 bitsx28bits -> 48 bits
  - Son résultat fait 48 bits et sert de paramètre à la fonction f de DES

DES : génération des clefs



64 bits donc 8 bits de parité
répartis aux positions 8, 16, ... 64

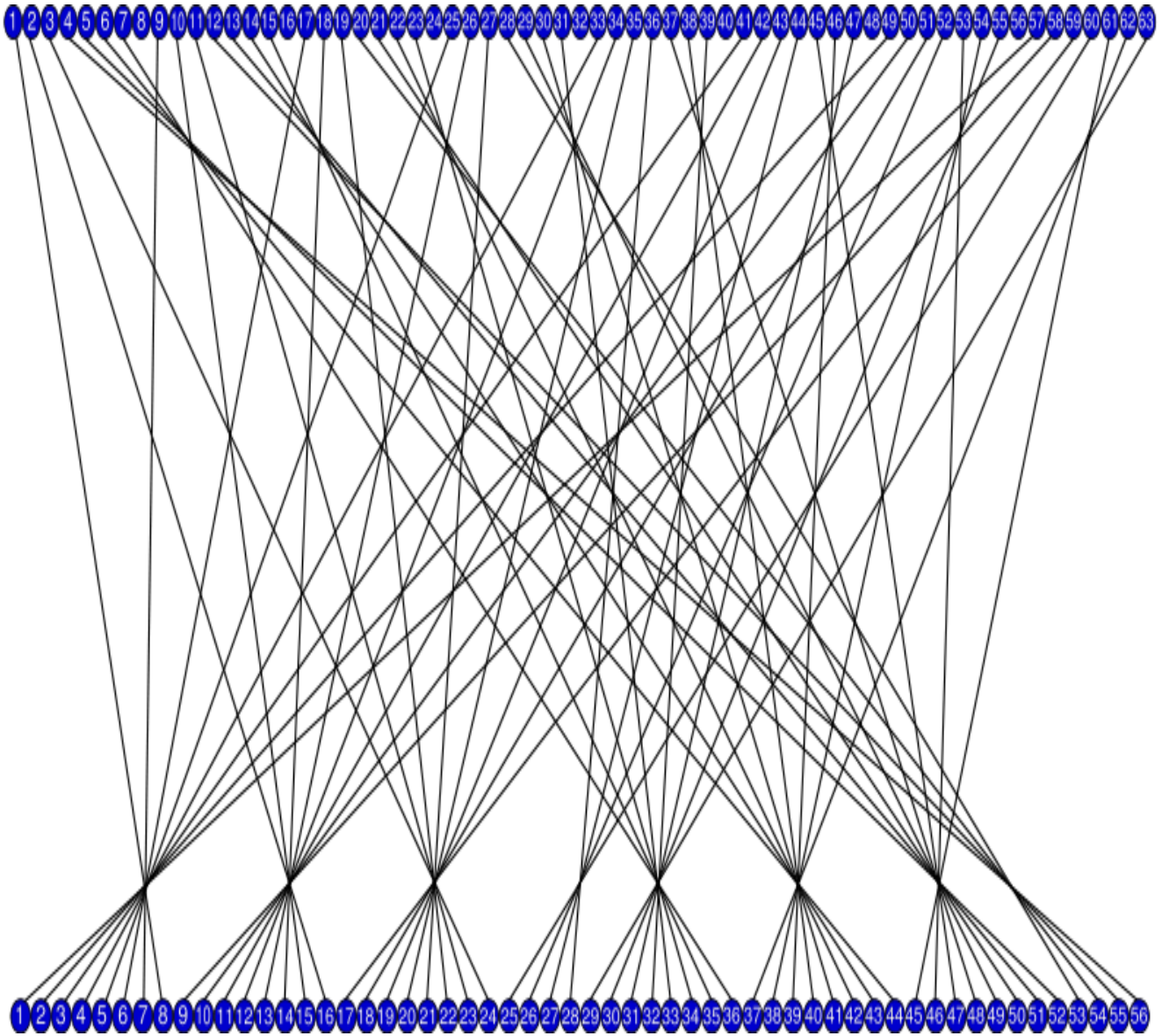


DES: génération des clefs des rondes PC1

Gauche						
57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
Droite						
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

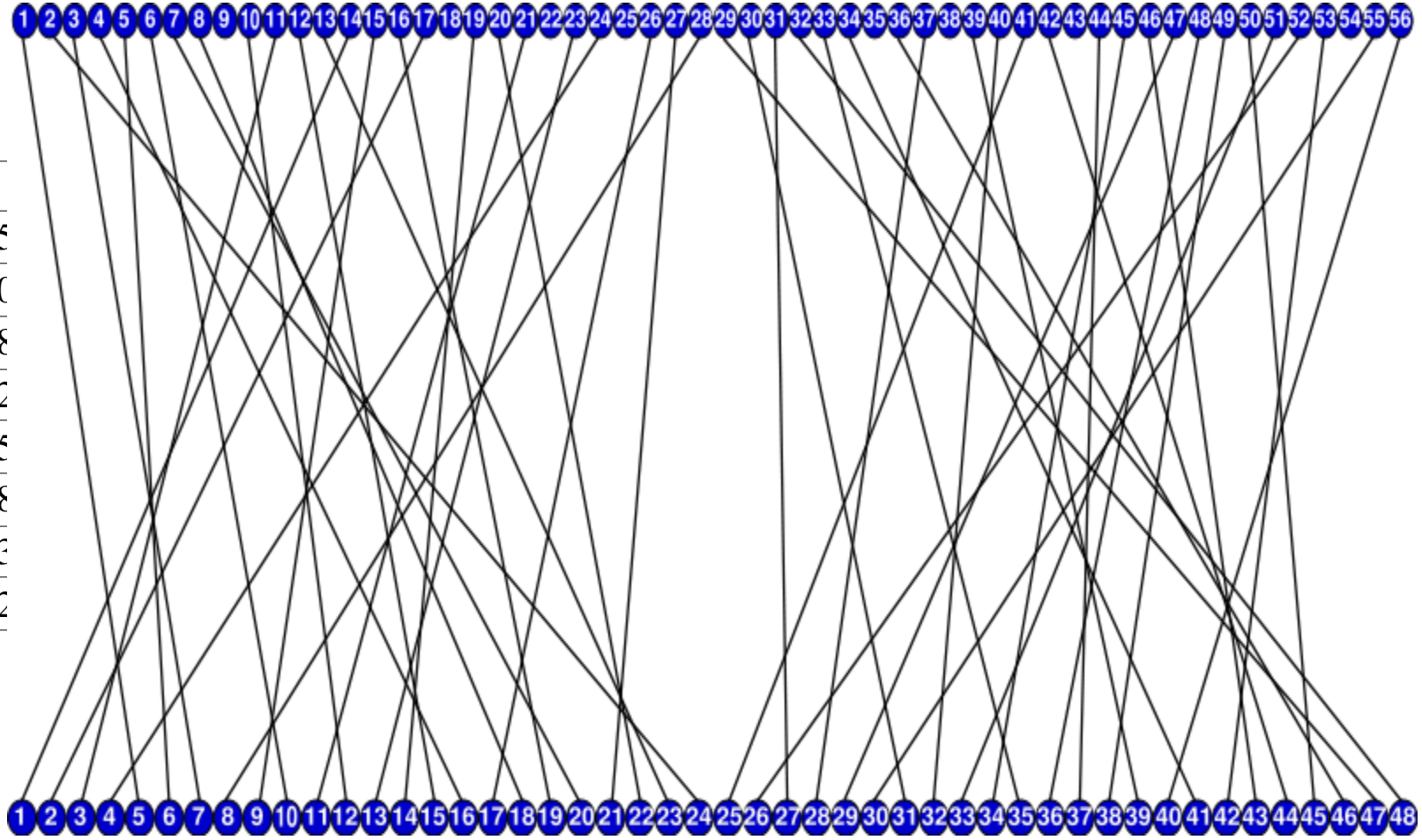
Si $k=k_1k_2...k_{64}$ alors :

- $C=k_{57}k_{49}...k_{36}$
- $D=k_{63}k_{55}...k_4$



DES: génération des clefs des rondes PC2

PC-2					
14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32



$PC2(b_1 b_2 ... b_{56}) = b_{14} b_{17} ... b_{32}$

DES: performances

- Des opérations simples:
  - La structure de Feistel est rapide
  - La dérivation des clefs est particulièrement simple et efficace
  - Les boîtes S se cablent facilement en matériel
- DES utilise des opérations simples facilement implémentables sur des puces spécialisées
  - XXXMb/s sur un XXX à XXX MHz
  - 1,5Gb/s sur une puce spécialisée bas de gamme
-

DES: sécurité

- Après 16 tours, résultat statistiquement plat
  - Aucune propriété du message source ne sera détectable
- Une légère modification du texte chiffré ou de la clef => de grosses modifications du texte chiffré
  - Bonné résistance aux attaques par analyse différentielles

DES: cryptanalyse

- Force brute : espace des clefs : 2^{56}
  - À la portée d'une PME (cf TD)
  - En 1998, l'EFF (Electronic Frontier Fondation) construit une machine
    - Craquant DES en 9 jours
    - Coût: $250 000
- Taille des blocs:
  - 64 est devenu court
  - On peut obtenir deux blocs chiffrés identiques ce qui laisse la porte ouverte à certaines attaques

DES: amélioration

- Idée d'amélioration :
  - Combiner plusieurs chiffrement DES de suite avec des clefs différentes
  - Le résultat n'est pas un DES
    - E(k1, E(k2, E(k3, M)) n'est pas égal à E(k4, M) (avec E: chiffrement DES et C: déchiffrement DES)
    - Sinon, pas d'augmentation de la complexité
  - Plusieurs tentatives :
    - Double DES : complexité #64 bits
    - Triple DES-EEE : 3 chiffrements avec des clefs différentes: E(k1, E(k2, E(k3, M))
    - Triple DES-EDE: E(k1, D(k2, E(k1, M)))
  - 3 fois plus lent que DES

DES: double DES

- Double DES:
  - E(k1, E(K2, M))
  - Atteint-on le niveau de sécurité d'un espace de clef de 56+56=112 bits. NON

Double DES: Attaque

1) On suppose que l'on connait un coupe clair M/chiffré C (un bloc d'un long message suffit)
2) On chiffre M avec toutes les clefs possibles : 2^{56}clefs possible
3) On trie les clefs : $56*2^{56}$ (n log n)
4) On déchiffre C avec toutes les clefs possible : 2^{56}clefs possible
   - On cherche la version déchiffrée avec la clef j dans l'ensemble de l'étape 2: au pire en log n donc au pire $56*2^{56}$
   - Si on trouve i une clef telle que E(i,M)=D(j,C) alors, on a trouvé le bon couple de clefs

– Complexité totale équivalente à celle d'un espace de clef de 64 bits

Double DES: Attaque

- L'attaque précédente ne permet pas d'avoir la certitude de trouver le bon couple de clef
  - On a 2^{64} messages chiffrés possibles
  - On a 2^{112} clefs possibles
  - Donc en moyenne, il y a $2^{112}/2^{64} = 2^{48}$couples de clefs permettant de transformer M en C
  - Donc si on trouve un couple qui marche, c'est l'un des couples qui marchent mais pas forcément le couple de clefs.
- Si on a deux textes clair M/ Chiffré C, le nombre moyen de clefs permettant de passer de M en C pour les deux textes est 2^{-16}. (16=2*64-112)
  - Donc si on trouve un couple qui marche, c'est probablement le bon couple de clef

Triple DES:

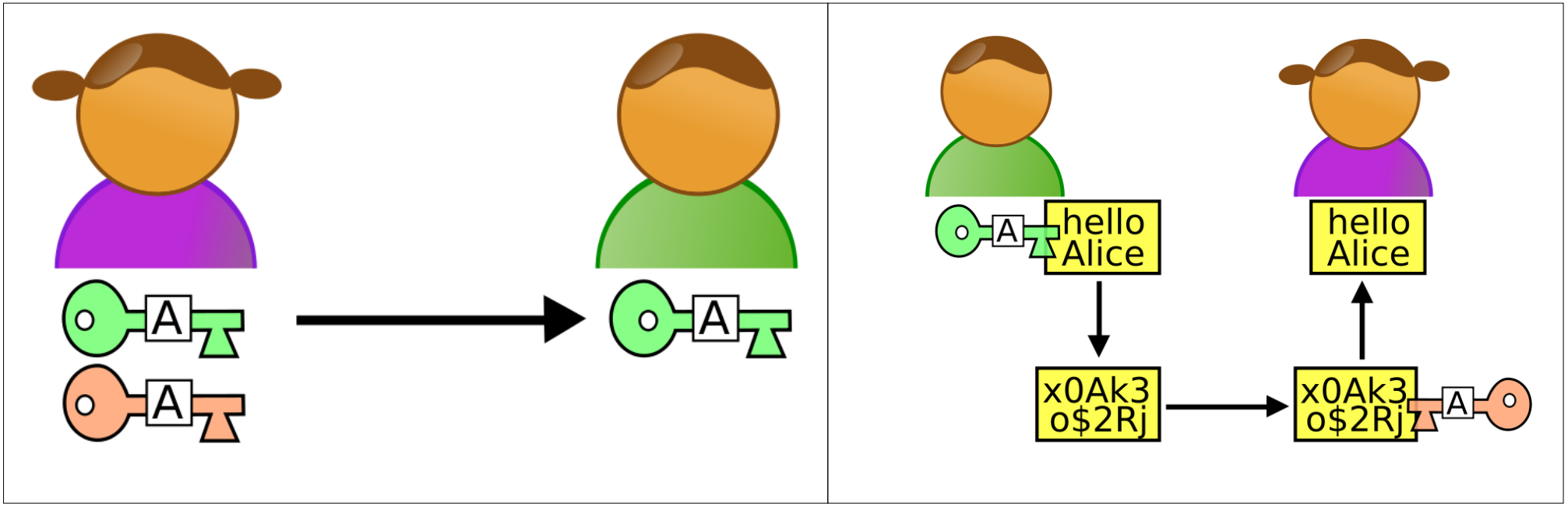
- De nombreuses variantes possible
  - Certaines ont des faiblesses
  - DES-EEE : 3 clefs de 56 bits => 168 bits mais l'attaque « meet in the middle » rend la sécurité équivalente à un espace de clef de 112 bits
- La variante conseillée (FIPS 46-3) est DES-EDE:
  - 2 clefs
  - E(k1, D(k2, E(k1, M)))
- Espace de clef de 112 bits
  - Mais en tenant compte des attaques possibles, équivalent à un espace de clef de 80 bits
- Triple Des a été très utilisé avant l'arrivé d'AES pour palier les faiblesses de DES

Chiffrement symétrique: nombre et diffusion des clefs

- Pour permettre à n personnes de communiquer,
  - il faut que chaque personne ait les clefs des (n-1) autres.
  - Au total: n(n-1) clefs
- Centrale de clefs:
  - Elle a la clef secrète de chaque utilisateur
  - Les utilisateurs lui envoient les messages
  - Elle les déchiffre et le rechiffre avec la clef privée du destinataire
  - Il faut n clefs
  - La centrale peut lire les messages de tout le monde et doit stocker les clefs de façon sûre.

Principe du chiffrement asymétrique

- La clef de chiffrement est publique
- La clef de déchiffrement est secrète et ne sert qu'au déchiffrement
- La clef publique
  - Ne permet pas de déchiffrer
  - Ne permet pas à un attaquant de trouver la clef privée



Chiffrement à clef publique

- Il est constitué de 3 algorithmes
  - L'algorithme de génération des clefs
    - (pk, sk)=KG(l) produit un couple clef privée, clef publique à partir d'un paramètre de sécurité l (souvent une saisie aléatoire)
  - L'algorithme de chiffrement
    - C=E(M, pk) utilise la clef publique pk
  - L'algorithme de déchiffrement
    - M=D(C, sk) utilise la clef privée sk
  - On a M=D(E(M, pk), sk)
  - On a **parfois** M=E(D(M, sk), pk). Les procédés qui vérifient cette seconde propriété peuvent servir de base à un procédé de signature

Le chiffrement RSA

- Soit deux grands nombres premiers impairs p et q
- n=pq
- Soit e un entier tel que
  - $1<e<\varphi(n)=(p-1)(q-1)$. $\varphi(n)$ est l'ordre du groupe multiplicatif (Z/nZ)*
  - PGCD(e, (p-1)(q-1))=1 (i.-e. e est premier avec $\varphi(n)$)
- Soit d tel que $d=e^{-1} \bmod \varphi(n)$ (d est l'inverse de e dans (Z/nZ)*) donc il existe u tel que $ed+u\varphi(n)=1$
- D'après le théorème d'Euler ($x^{\varphi(n)}=1 \bmod n$)
  - $(m^e)^d=m^{ed}=m^{1-\varphi(n)}=m \bmod n$

RSA:

- Informations publiques :
  - n=pq (p et q premiers)
  - e
- Informations privées:
  - $d=e^{-1} \bmod \varphi(n)$: clef secrète

- Génération de clefs: ((n,e), d)=KG(l)
- Chiffrement: $C=E(m)=m^e \bmod n$
- Déchiffrement: $m=D(C)=c^d=m^{ed}=m \pmod n$

Propriétés de RSA

- Propriété multiplicative
  - Le chiffré d'un produit est égal au produit des chiffrés
    - $(m_1 m_2)^e = m_1^e \, m_2^e = c_1 c_2 \bmod n$
  - Conséquence: attaque à chiffré choisi
    - Soit C un message chiffré (C=E(m), m est inconnu)
    - Supposons qu'on connaisse r et C' tels que
      - On sait déchiffrer C' en m'
      - $C' = r^e C$
    - Alors on sait déchiffrer C car m'=r.m

Propriétés de RSA

- Chiffrement et déchiffrement sont commutatifs
- $m=D(C(m))=m^{ed}=C(D(m))=m^{de}$
- Conséquence: RSA pourra être utilisé comme base d'un algorithme de signature:
  - On signe un document en chiffrant avec la clef privée
  - On le vérifie en déchiffrant avec la clef publique
- Pouvoir être déchiffré avec la clef publique est-il une preuve que ça a été chiffré avec une clef privée donnée ?

RSA: sélection des paramètres

- Choisir p et q de taille similaire pour éviter la factorisation par courbes elliptiques
- Choisir p et q au hasard pour éviter que p et q soit trop proches (sinon p et q de l'ordre de racine de n)
- Choisir p et q des nombres premiers forts
  - p-1 a un grand facteur premier (pour éviter l'algo p-1 de Pollard). grand=au moins 200bits
  - p+1 a un grand facteur premier (pour éviter l'algode Williams
  - r-1 a un grand facteur premier (attaques par cycles)
- Tiré de [Bresson]

Sécurité de RSA

- 2 méthodes d'attaque
  - Trouver d ou p et q : problème de factorisation
  - Trouver m à partir de m^e: extraction de racines e-ièmes
- En pratique, la factorisation est la seule méthode connue pour casser RSA
  - C'est un problème difficile
- Il n'a pas été prouvé que RSA était aussi solide que la factorisation
  - Apparition d'une attaque plus simple que la factorisation possible

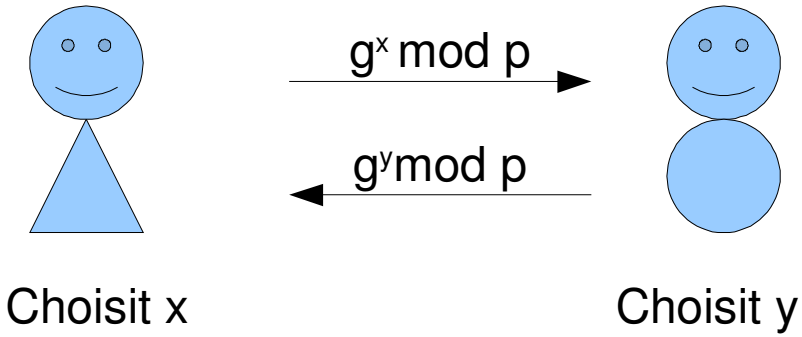
Diffie Hellman

- Années 1970: comment se passer de la transmission d'un secret partagé ?
- Réponses :
  - Chiffrement à clef publique: RSA 1977
  - Diffie Hellman 1976 : se mettre d'accord sur un secret partagé en n'échangeant que des informations publiques

Diffie Hellman: de l'importance la commutativité

- Exemple d'algorithme d'échange de clef :
  - Alice met un secret dans un coffre
  - Alice ferme le coffre avec un cadenas
  - Alice envoie le coffre à Bob
  - Bob ajoute au coffre un cadenas
  - Bob envoie le coffre à Alice
  - Alice retire son cadenas
  - Alice envoie le coffre à Bob
  - Bob retire son cadenas et lit le secret
- Le fait fondamental est que les cadenas peuvent être mis et retirés dans n'importe quel ordre
- Exemple incorrect: Bob met le coffre d'Alice dans un autre coffre fermé à clef par Bob

Diffie Hellman



- Valeurs publiques:
  - p un grand nombre premier
  - G un groupe multiplicatif de cardinal p-1
  - g un générateur de G
- Secret d'Alice: x, secret de Bob: y
- Secret commun: $K=g^{xy}=(g^x)^y=(g^y)^x$
- Attaque: trouver x connaissant g^x
  - Problème du logarithme discret
  - Pas d'attaque raisonnable si p est grand

Hachage

- Fonction h à sens unique
  - Entrée de taille quelconque
  - Sortie de taille fixe
  - h(m) est appelé l'empreinte (hash) du message m
- Propriétés:
  - Facile à calculer
  - Inversion difficile: « matériellement impossible » de retrouver m à partir de h(m)
  - Collission difficile: « matériellement impossible » de retrouver m' tel que h(m)=h(m')
  - Dispersion: changer un bit du message initial change de nombreux bits de la sortie

Hachage

- Utilisation:
  - Validation de l'intégrité des données
    - On obtient m (par un canal performant) et h (par un canal sur)
    - On vérifie que h=h(m) pour vérifier l'intégrité de m
  - Stockage des mots de passe:
    - On stocke h(mdp)
    - Pb: éviter que deux utilisateurs ayant le même mot de passe aient la même valeur stockée
      - Ajout d'une valeur (grain de sel ou « salt »)

Hachage: attaque des anniversaires

- On calcule et on stocke autant de valeur de h qu'on le peut
- Question: combien en faut-il pour avoir un probabilité ½ de trouver une collision (2 valeurs différentes ayant la même empreinte) ?
- Réponse : $(1+ \mathrm{sqrt}(1+8*\ln(2)*2^n))/2$
  - Soit un peu plus que $2^{n/2}$ (n est la taille de l'empreinte en nombre de bits)
  - De nos jours (12/2008), 128 ou 160 sont des valeurs minimales

Hachage: fonctions classiques

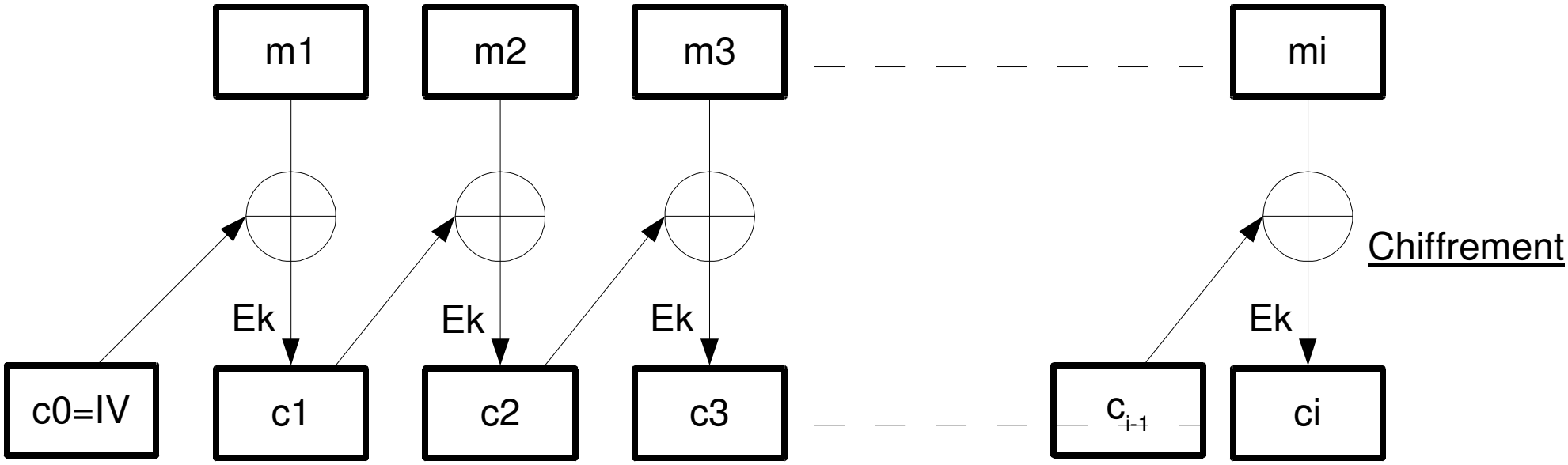
- SHA-1, SHA-224, SHA-256, SHA-512
- MD4, MD5

MAC: Message Authentification Code

- But: authentifier un message à l'aide d'un authentifiant de petite taille (de 64 à 256 bits)
- Crypto à clef secrète:
  - un MAC est une fonction de hachage qui prend une clef secrète en argument
  - L'expéditeur et le destinataire du message doivent avoir la clef secrète
- Exemple d'utilisation
  - Alice envoie m à Bob avec son MAC h=H(k, m)
  - Bob reçoit m' et h' (a priori m'=m et h'=h)
  - Bob calcule H(k,m') et le compare à h'. égalité => OK
  - Un attaquant ne pourrait pas créer m' et h' valides sans connaître k.
  - Remarque: la fonction de déchiffrement ne sert pas.

MAC: CBC-MAC

- CBC-MAC: on applique la méthode CBC et on garde seulement le dernier bloc chiffré



Mac versus signatures

- Signature
  - Mécanisme à clef publique
  - Non répudiation
  - Authentification du signataire
  - Intégrité des données
- MAC:
  - Mécanisme à clef secrète
  - Authentification de l'émetteur
  - Intégrité des données

PKI: Public Key Infrastructure (IGC: Infrastructure de Gestion de Clefs)

- Problème:
  - comment être sur qu'une clef publique est valide et est bien la clef publique d'une personne donnée ?
- Solutions:
  - PGP: confiance transitive : WeB Of Trust
  - Certification par un tiers de confiance : IGC

IGC: définition

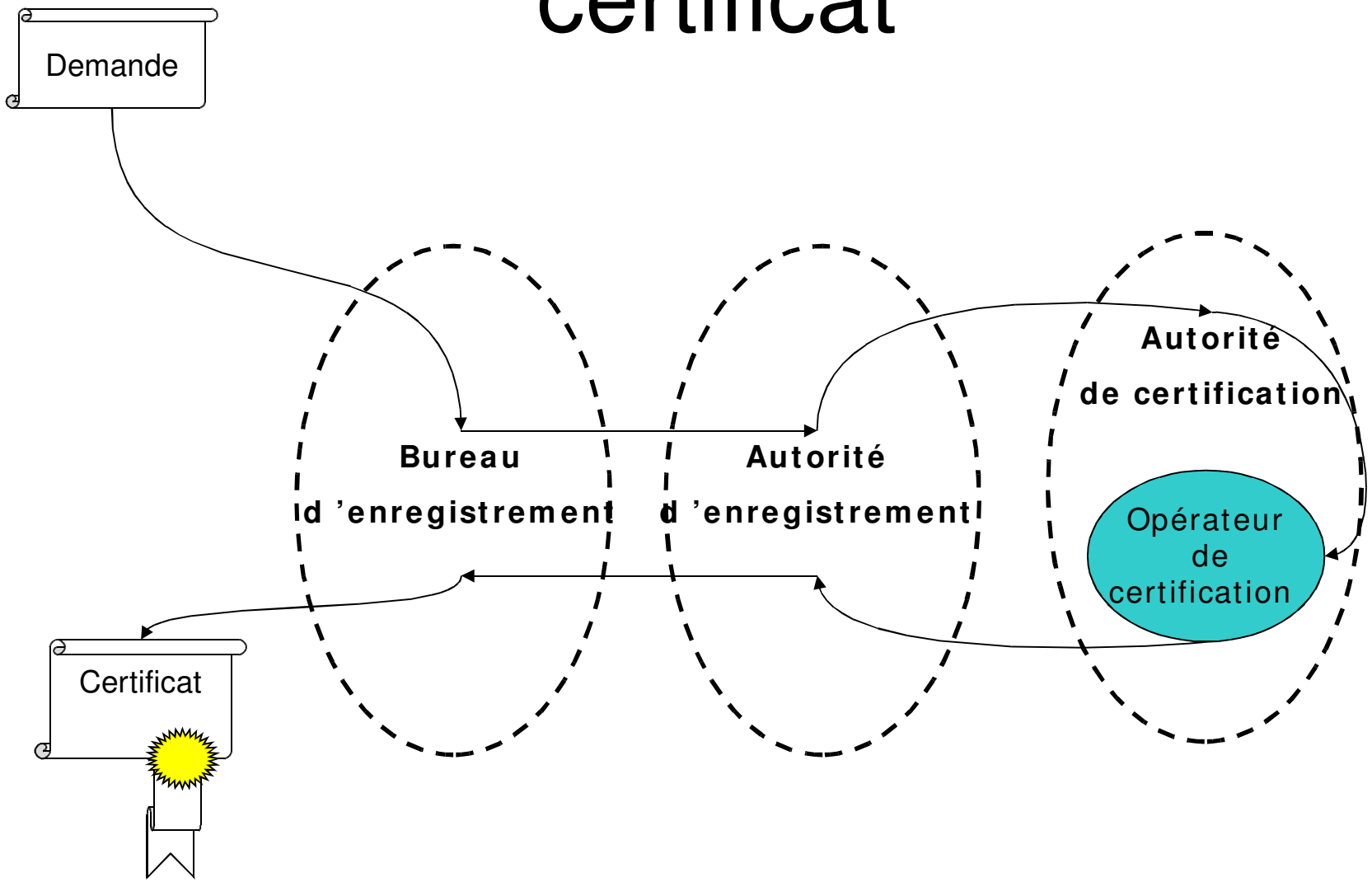
- IGC: ensemble de moyens matériels, de logiciels, de composants cryptographiques mis en oeuvre par des personnes, combinés par des politiques, des pratiques, des procédures requises qui permettent de :
  - créer
  - gérer
  - conserver
  - distribuer
  - révoquer

  des certificats basés sur la cryptographie asymétrique

éléments obligatoires d'une IGC

- 3 éléments obligatoires :
  - autorité de certification
  - autorité d'enregistrement
  - service de publication

PKI: processus de création d'un certificat



•autorité de certification (CA ou Certification Authority en anglais)

- autorité de confiance reconnue par une communauté d'utilisateurs
- délivre et gère des certificats de clefs publiques
- maintient une liste des certificats révoqués (LCR en français, CRL en anglais)
- les certificats sont conformes à la norme X.509
- génère les certificats à clef publique et garantit l'intégrité et la véracité des informations qu'ils contiennent en les signant avec sa clef privée

•Autorité d'enregistrement (RA: Registry Authority)

- intermédiaire entre l'utilisateur et l'autorité de certification.
- l'utilisateur s'adresse à elle
- en application de la politique de certification, elle vérifie les données de l'utilisateur :
  - identité
  - correspondance clef privée/publique
  - ...
- transmet les informations validées à l'AC

•Service de publication

- met à la disposition de la communauté les certificats générés par l'AC
- publie aussi la liste des certificats révoqués

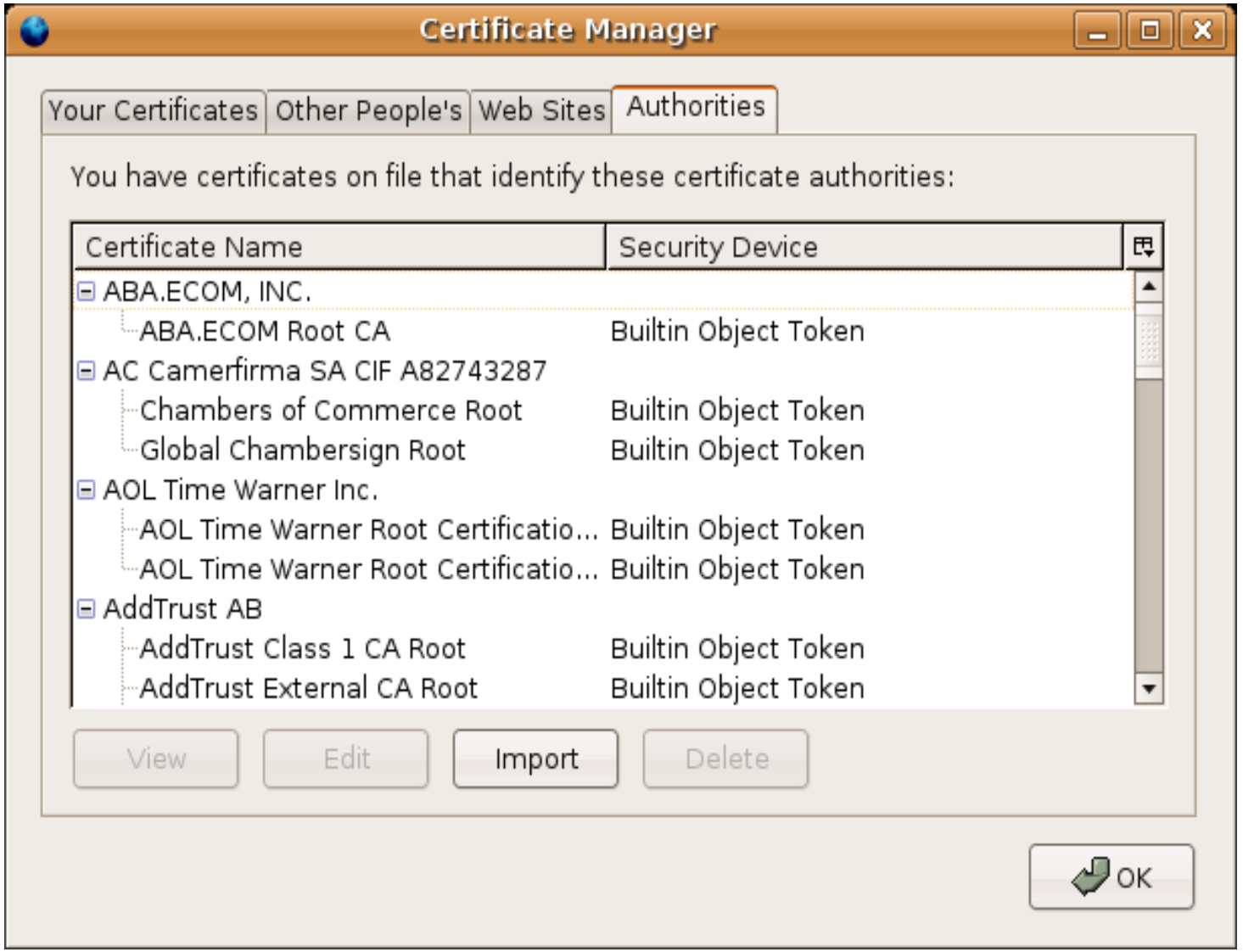
•Composants optionnels de l'IGC

- autorité d'horodatage (AH ou Timestamping Authority)
  - date des données qui lui sont transmises
  - Le Protocole D'horodatage (ou Time-Stamp Protocol) : rfc 3161
- service de séquestre:
  - stocke de façon sûre des clefs privées
  - pour permettre le déchiffrage des données en cas de perte
  - ne doit pas concerner les clefs de signature

•Certificat

- contient entre autre
  - l'identité de son propriétaire (personne, machine, ...)
  - sa clef publique signé par une AC
  - période de validité
  - type d'utilisation de la clef (champ optionnel)
  - ...

- Exemple: navigateur WeB

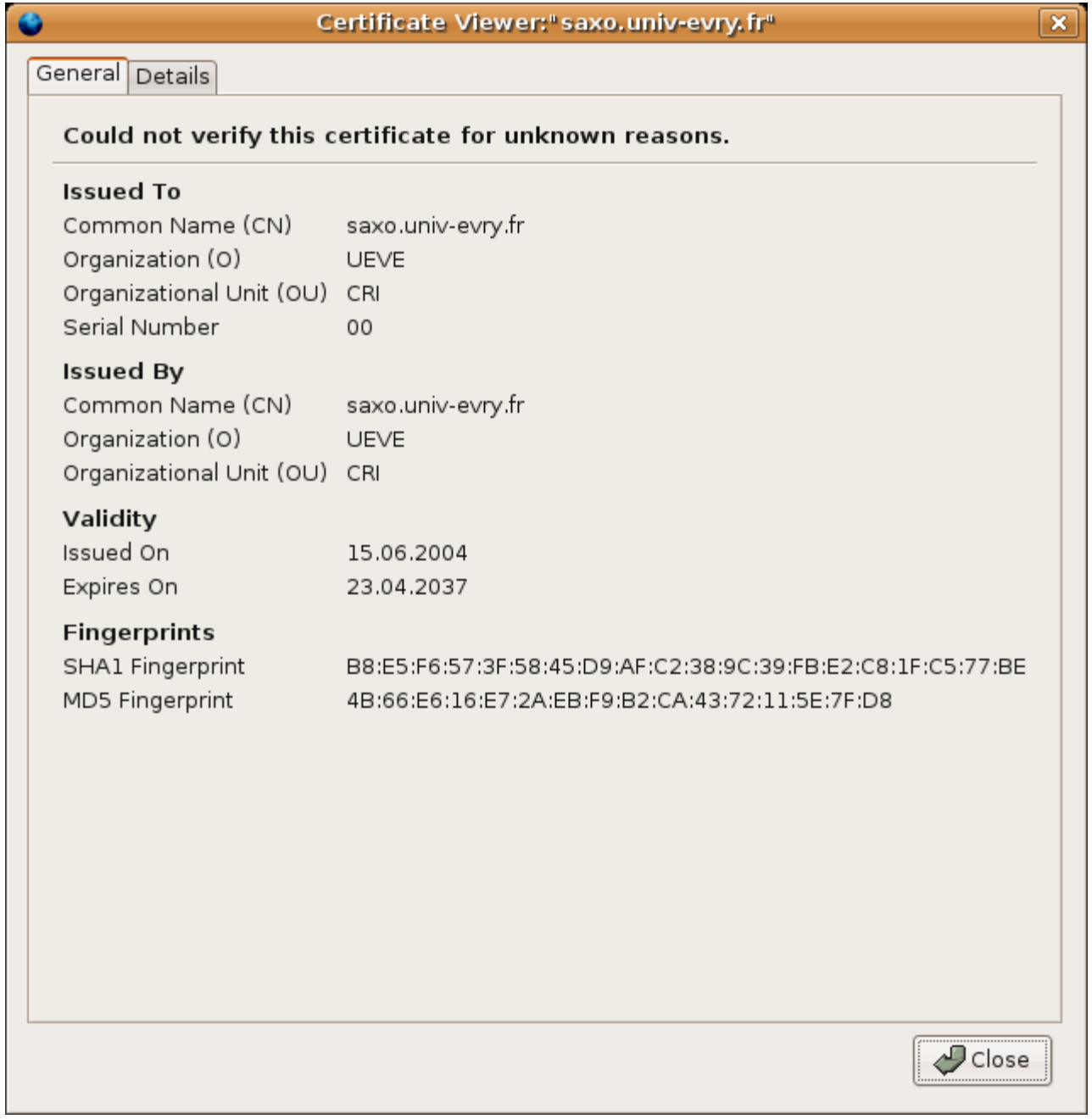


•Exemple: navigateur WeB:

- un client se connecte sur le site WeB de l'entreprise
- il obtient les références de l'AC et le certificat
- il vérifie le certificat
- il génère une clef de session qu'il transmet chiffrée au serveur de l'entreprise
- la session est maintenant chiffrée

•Exemple: Pb certification

•Exemple: certificat



•Authentification interactive

- Prouver interactivement son identité
- 2 entités:
  - Valérie: vérificateur
  - Pedro: le prouveur : il a un secret qui prouve son identité
  - Pedro doit convaincre Valérie de son identité
- Solutions usuelles
  - Mot de passe
  - Chiffrement symétrique
  - Chiffrement asymétrique, signature
  - Divulgation nulle (zero knowledge)

•Propriétés des protocoles utilisés

- Correction: Si Pedro connait le secret, il doit pouvoir convaincre Valérie qu'il est bien Pedro
- La preuve d'identité est significative: une personne ne connaissant pas le secret (l'espion) doit être détecté
- Sans divulgation d'informations
  - Réutilisables : Valérie ne doit pas pouvoir utiliser les informations pour s'authentifier à la place de Pedro
  - Un espion ne doit pas pouvoir rejouer le protocole
  - Si l'espion a remplacé Valérie, il ne doit pouvoir rien faire des informations obtenues

•Mot de passe

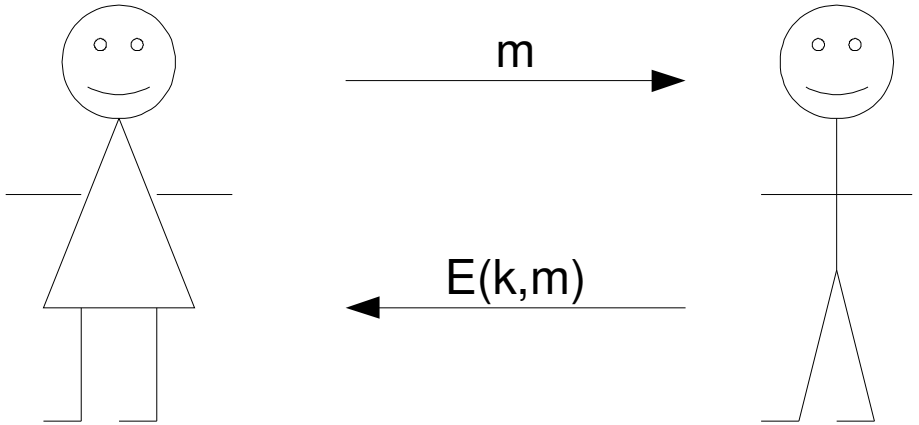
- Exemple de l'authentification sur le WeB:
  - Le même mot de passe sur chaque site ?
    - Attention à l'espionnage sur le réseau
    - Si le webmaster est malhonnete, il peut tenter de réutiliser les informations d'authentification ailleurs
  - Exemple de la caverne
  - Principe du zero knowledge
  - Exemple de protocole

•Mot de passe

- Authentification sur un site WeB: avec un login/mot de passe
- En général :
  - Aucune garantie que le concepteur du site WeB ne conserve pas le mot de passe
  - Impose un mot de passe différent sur chaque site (lourd)
- Idéalement:
  - Pouvoir utiliser un mot de passe commun à tous les sites WeB
  - Impossibilité pour les responsables des sites ou un espion de récupérer le mot de passe

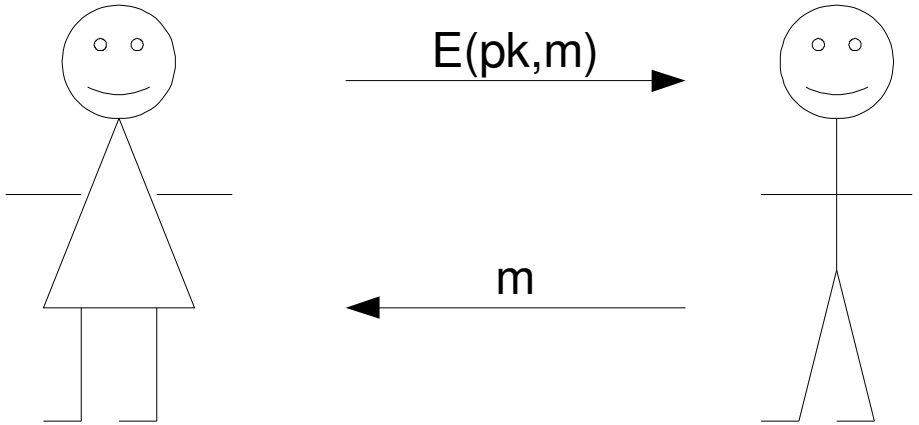
•Chiffrement symétrique

- Méthode par défi/réponse
  - Si Valérie est malhonnête, elle peut faire chiffrer des messages m arbitraires
  - L'espion apprend des couples (clair, chiffrés)



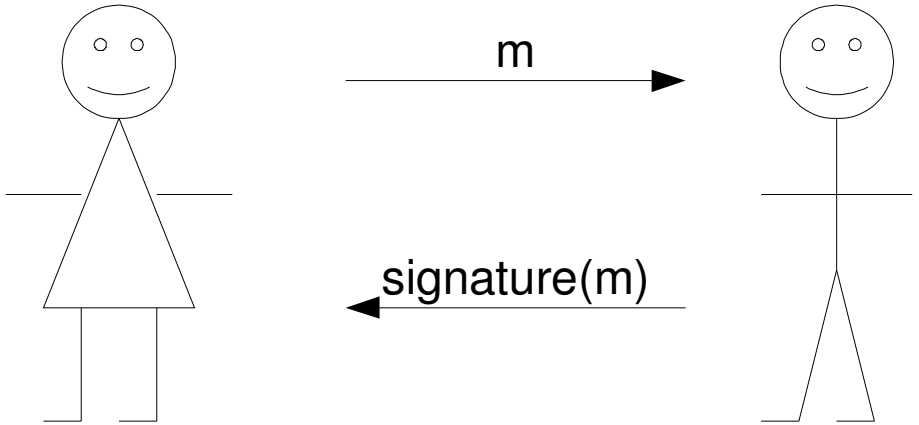
•Chiffrement à clef publique

- Méthode par défi/réponse
  - Pk : clef publique
  - L'espion n'obtient aucune information non publique



•signature à clef publique

- Méthode par défi/réponse
  - Si Valérie est malhonnête, elle peut faire signer des messages arbitraires m



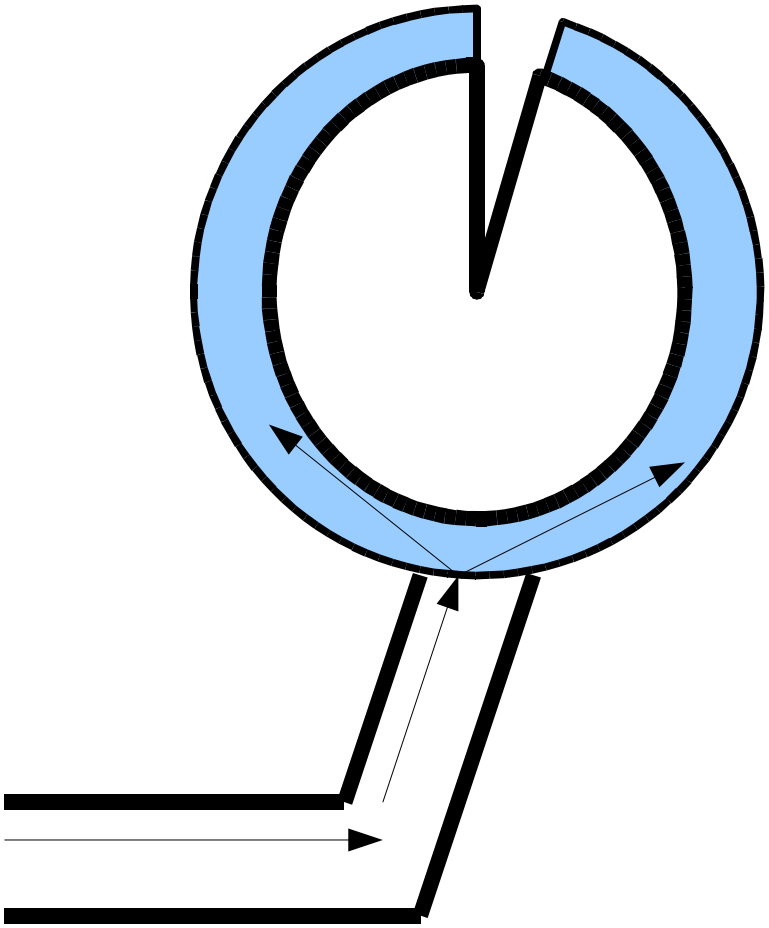
•SSO: authentification unique

- Authentification unique (SSO: Single Sign On) à la openID ou Kerberos

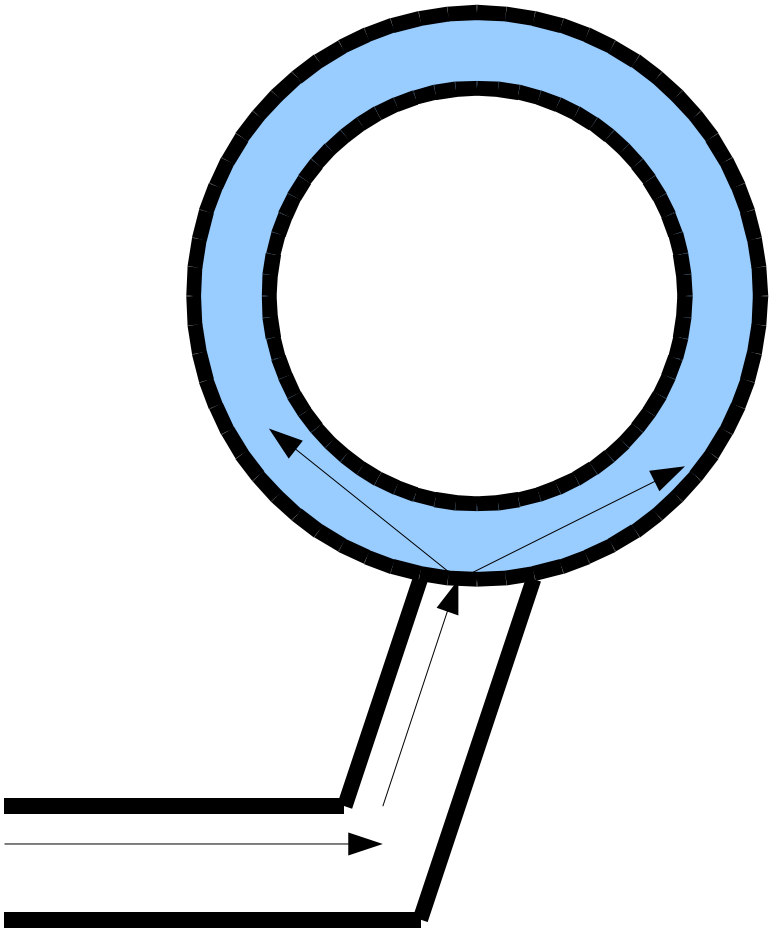
•Zero knowledge: buts

- Prouver **interactivement** que l'on connaît un secret
- Sans fournir d'information sur ce secret
  - Les échanges entre Pedro (Prouveur) et Valérie (Vérificateur) ne contiennent aucune information sur le secret

- Zero knowledge: AliBaba



- Zero knowledge: AliBaba



Zero knowledge: AliBaba

- De l'importance de l'interactivité
- De l'impossibilité de transmettre la preuve reçue
  - Exemple avec une caméra

Protocole de Fiat-Shamir

- Il s'appuie sur la difficulté de trouver des racines carrées modulo n si n est grand
  - Soit n =p*q avec p et q deux grands nombres premiers
  - Soit x premier avec n
  - **Connaître x^2 mod n ne permet pas de retrouver x dans un temps raisonnable**

Protocole de Fiat-Shamir

- Données secrètes:
  - s premier avec n
- Données publiques:
  - n
  - $v=s^2 \bmod n$
- But: prouver que l'on connaît s sans le révéler

Protocole de Fiat-Shamir: une étape

1) Pedro tire un nombre r_i mod n, non nul, au hasard
2) Pedro transmet $y_i = r_i^2$ mod n à Valérie (Vérifieuse)
3) Valérie tire une valeur b au hasard dans {0, 1}
4) Valérie transmet b à Pedro (le prouveur)
5) Si b vaut 0,
   - Pedro transmet r_i mod n à Valérie
   - Valérie élève la valeur reçue au carré et accepte la réponse si le résultat du calcl vaut y_i mod n
6) Si b vaut 1,
   - Pedro transmet $s*r_i$ mod n à Valérie
   - Valérie élève la valeur reçue au carré et accepte la réponse si le résultat du calcl vaut $v*y_i$ mod n

Protocole de Fiat-Shamir

- Le protocole consiste à effectuer cette étape un nombre de fois suffisant
- À noter qu'à chaque étape, on choisit un r différent
- Une personne qui réalise honnêtement le protocole et qui ne connaît pas le secret ne peut répondre que dans le cas où b=0
  - Elle a donc une chance sur deux de répondre correctement à chaque étape
- Une personne malhonnête a aussi une chance sur deux de répondre correctement (cf ci-après)

Protocole de Fiat-Shamir: authentification

- Ce protocole réalise bien de l'authentification
  - Si Pedro connaît le secret, il peut répondre aux deux questions
  - Une personne qui ne connaît pas le secret à deux solutions à l'étape 2
    - Fournir $y=r^2 \bmod n$
      - Si b=0: il peut fournir r à Valérie qui retrouve y en l'élevant au carré donc réussite
      - Si b=1: il ne peut fournir s*r mod n car il ne connaît pas s donc échec
    - Fournir $y=r^2/v \bmod n$ (ce faisant, il ne respecte pas le protocole mais Valérie ne peut le savoir)
      - Si b=1: il founit r mod n et Valérie en l'élevant au carré retrouve bien y * v mod n donc réussite
      - Si b=0, $r^2 \bmod n$ ne vaut pas y donc échec

Protocole de Fiat-Shamir: authentification

- À chaque étape: une personne ne connaissant pas le secret à une chance sur deux de fournir une réponse correcte
  - Si on réalise t étapes, $1/2^t$ probabilité de réussir
  - On peut donc rendre aussi faible que l'on veut la chance de réussite d'une personne ne connaissant pas le secret en augmentant le nombre d'étapes
- L'élément important du protocole est que y est fourni avant que l'on connaisse la valeur de b.
  - Le cas b=0 empeche l'espion de fournir $v*r^2$
  - On peut le mettre en parallèle avec le choix du boyau de caverne avant le tirage au sort.
  - **L'interactivité est donc vitale**

Protocole de Fiat-Shamir: divulgation nulle

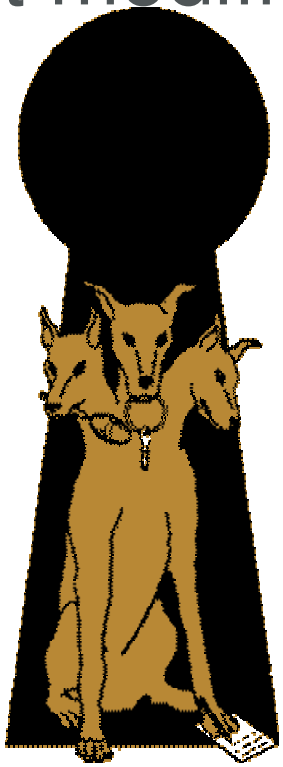
- Si b=0, s n'intervient pas
- Si b=1, s est masque dans s*r par le caractère aléatoire de r
- On ne divulgue donc aucune information sur s à part la valeur de s^2.

Étude de cas

- OpenPGP
- Kerberos
- SSL
- Gestion d'une PKI avec openssl

Kerberos et la Sécurité

Emmanuel Bouillon
SSTIC 04
Mise à jour et modification: P. Petit



1.1 Qu'est-ce Kerberos

Kerberos & Herakles poterie grecque VIe av JC

- Mythologie
- Projet Athena
- Aujourd'hui, protocole d'authentification réseau
  - Versions 1 à 3 : versions de développement
  - Version 4 :
    - Kerberos: An authentication service for computer networks.
    - The evolution of the Kerberos authentication system in distributed open systems.
  - Version actuelle : 5, RFC : 4120

1.2 Intérêt du protocole

- Problème de l'authentification réseau
  - **Unifiée (Sigle Sign On)**
  - **Hétérogène (y compris Unix / Windows)**
- Autres « solutions »
  - **Distributions /etc/passwd, /etc/shadow**
    - ☛**Difficultés de mise en œuvre, d'extensibilité**
    - ☛**Pas de SSO**
    - ☛**Pas d'intégration de Windows**
  - **Annuaires : NIS, LDAP**
    - ☛**Non conçus pour l'authentification**
    - ☛**Le rebond de service en service sans ré-authentification n'est pas sécurisé**
  - **Solutions propriétaires**
    - ☛**UIS (Unix Integration Services) : plus supporté**
    - ☛**SFU (Services For Unix)**

1.2 Intérêt du protocole

- Avantages de Kerberos
  - **Résout les problèmes classiques de l'authentification des clients et des services au sein d'un réseau**
    - **Authentification unifiée du client ET du Service**
    - **Mot de passe en clair**
    - **Rebond de services en services**
  - **Basé sur un standard**
    - **Ouvert :**
      - ✔ **RFC 4120**
      - ✔ **Plusieurs implémentations « Opensource »**
    - **Adopté par les systèmes propriétaires (Windows, Solaris, Irix, …)**
  - **Principal mécanisme de sécurité sous-jacent de la GSSAPI**
    - **RPCSEC_GSS**
      - ✔ **NFSv4**

2.1 Caractéristiques du protocole

- Basé sur
  - Needham et Schroeder "Using Encryption for Authentication in Large Networks of Computers" (1978)
  - Denning et Sacco "Time stamps in Key distribution protocols" (1981)
- Kerberos permet l'authentification des utilisateurs et des services sur un réseau
  - Part de la supposition que le réseau peut être non sûr
    - « open, unprotected network »
    - Les données sur le réseau peuvent est lues ou modifiées
    - Les adresses peuvent être faussées, …
  - Utilise une tierce partie de confiance
    - Toutes les entités du réseau (utilisateurs et services, appelées principaux) font confiance à cette tierce partie (le serveur Kerberos, KDC)
  - Utilise des mécanismes de chiffrement basés sur des algorithmes à clefs symétriques (ou secrète)
    - Tous les principaux partagent cette clef secrète avec le serveur Kerberos

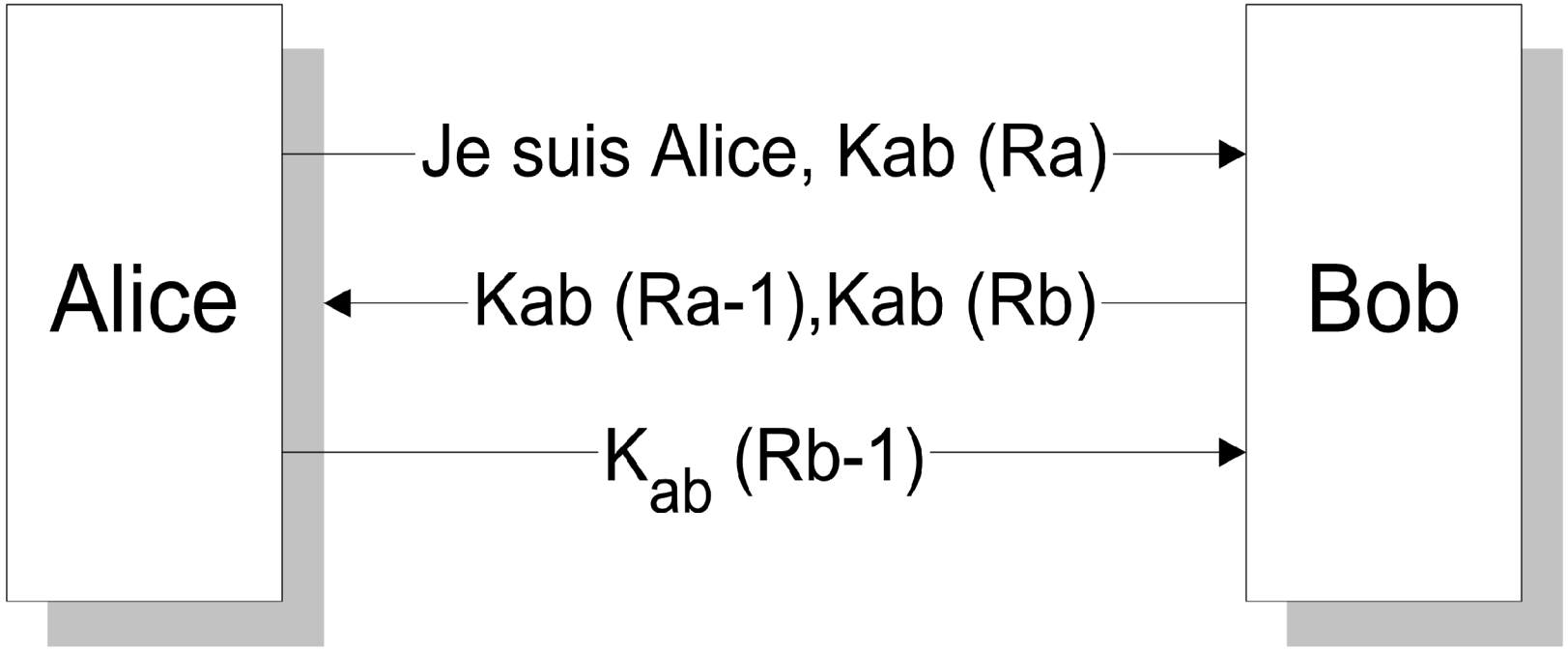
2.2 De l'authentification par secret partagé à Kerberos v5

- 2.2.1 : Authentification à l'aide d'algorithme de chiffrement à clefs secrètes : Alice et Bob
  - Alice initie la communication : le client ou l'utilisateur
  - Bob répond : service ou serveur applicatif
  - Alice veut accéder au service Bob

Authentification de Alice et Bob via la clé partagée K_{ab}

2.2.1 Authentification par secret partagé

- Authentification mutuelle, autre méthode (suite)



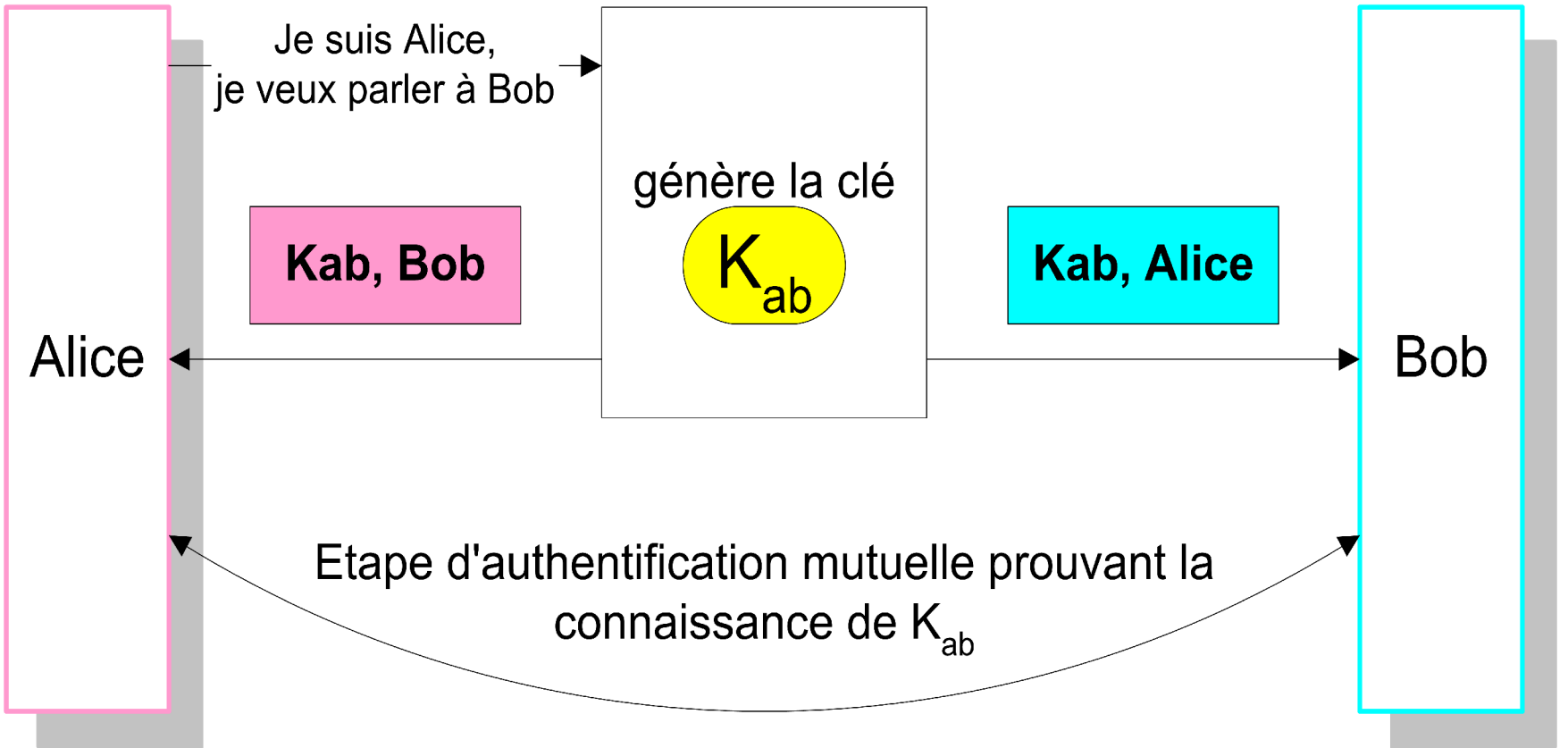
Authentification de Alice et Bob via la clé partagée K_{ab}

2.2.2 Utilisation d'une tierce partie de confiance

- Problème de ce schéma
  - Peu extensible
  - La généralisation à ***m*** utilisateurs et à ***n*** services, implique une distribution préalable de ***m x n*** clés partagées.
- Une amélioration possible :
  - Utiliser une tierce partie, avec laquelle tous les utilisateurs et les services partagent leur clé.
  - Présente aussi d'autres avantages :
    - ☛gestion centralisée de compte
    - ☛Plus facile de sécuriser une base de clés partagées que plusieurs

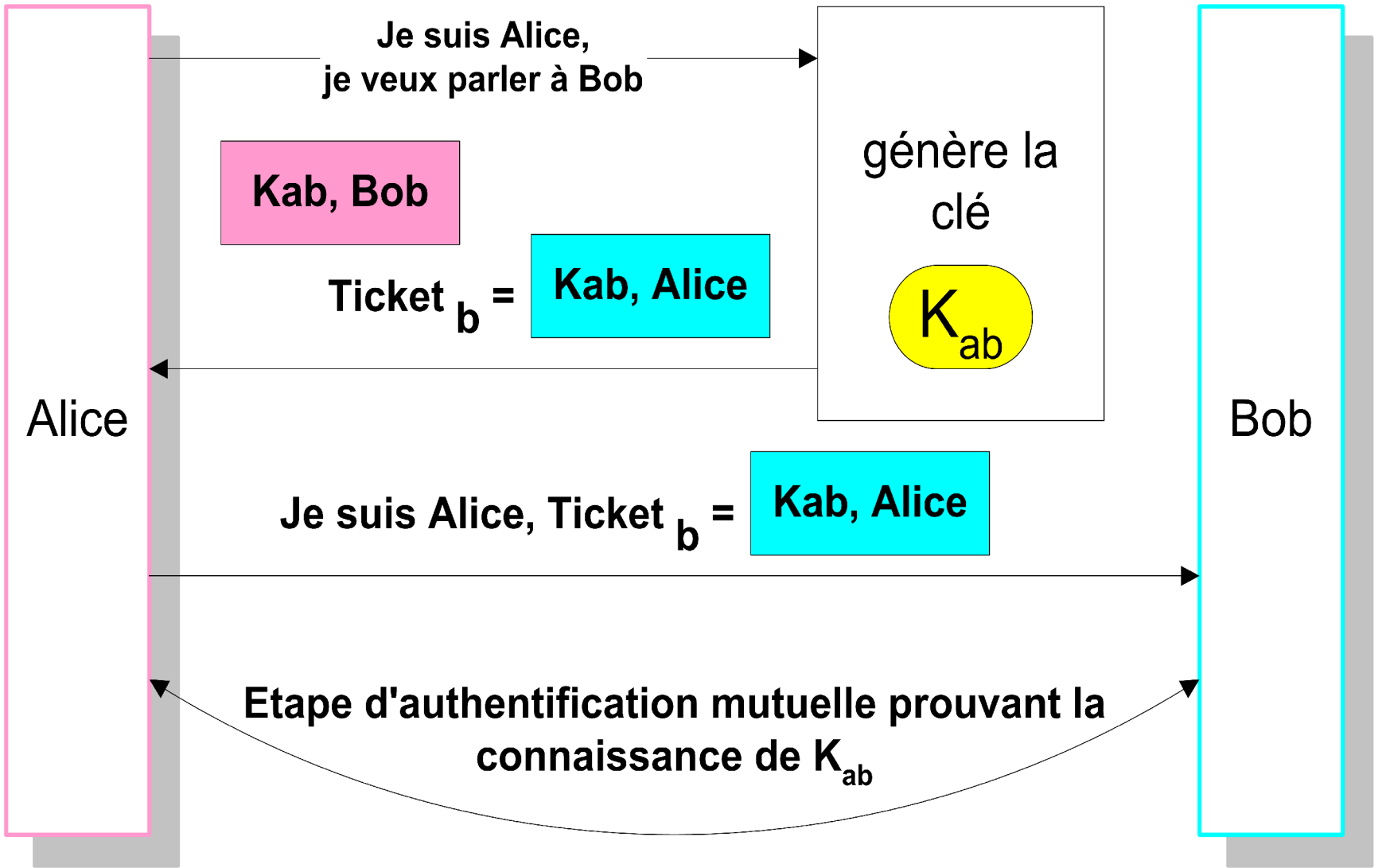
2.2.2 Utilisation d'une tierce partie de confiance

- Authentification mutuelle
- Notations
  - Ka : clé secrète de Alice, partagée par Alice et le KDC, rose
  - Kab : clé de session entre Alice et Bob, jaune
  - K{texte} : texte chiffré avec la clé K



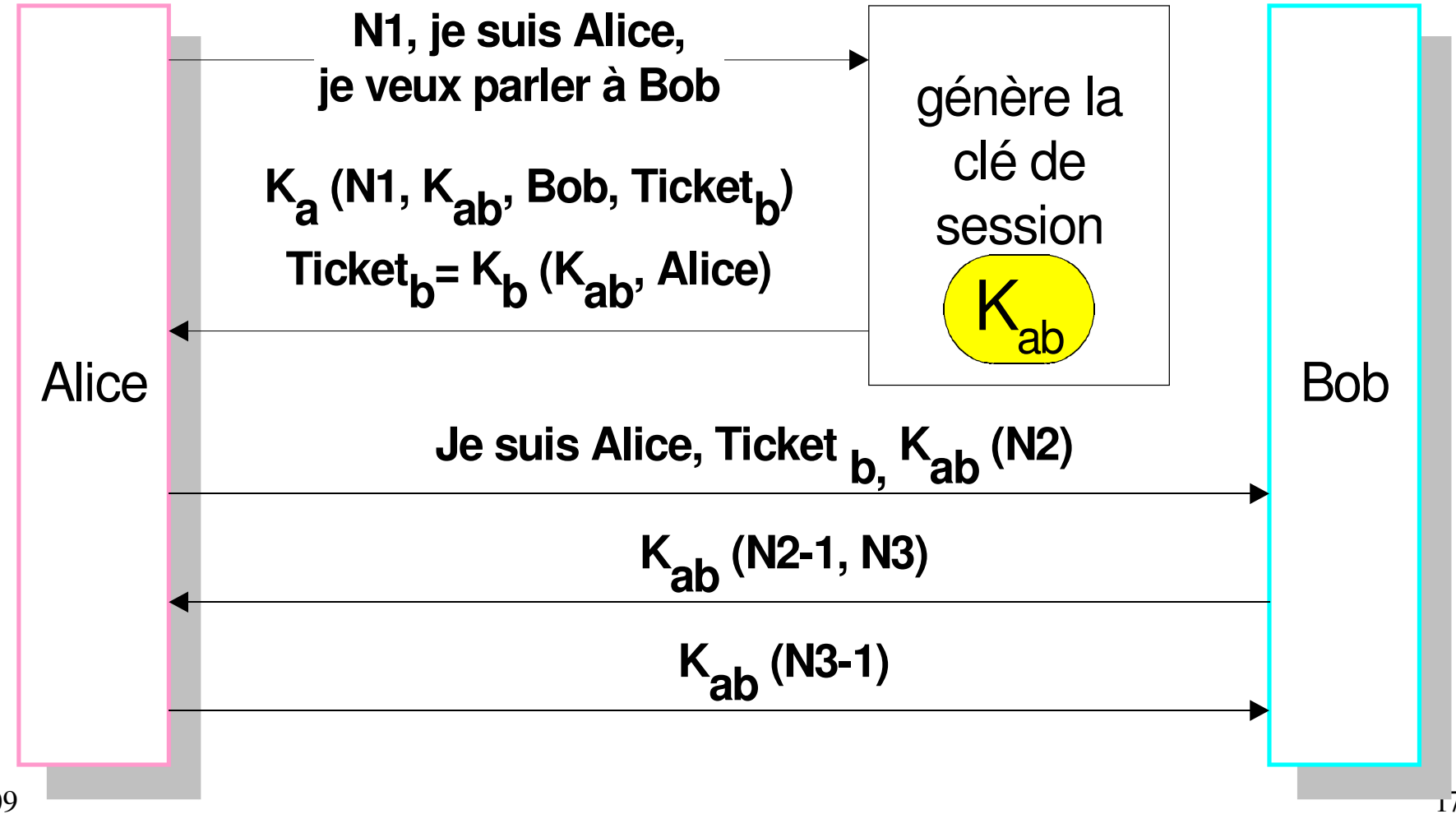
2.2.2 Utilisation d'une tierce partie de confiance

- Authentification mutuelle (suite)
  - Le ticket est envoyé à Alice



2.2.2 Utilisation d'une tierce partie de confiance

- Protocole Needham-Schroeder

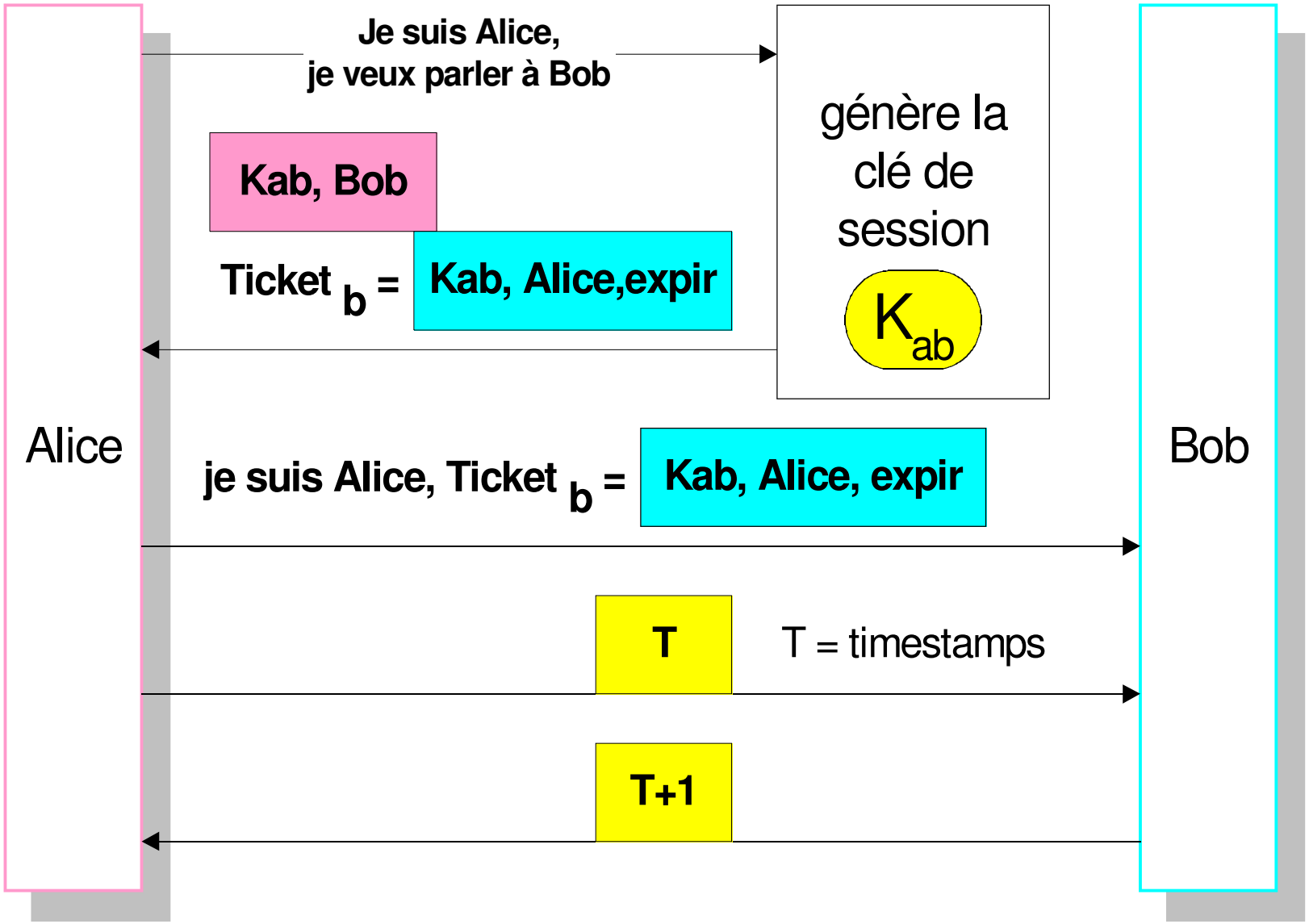


2.2.3 Kerberos

- Kerberos améliore le schéma de Needham et Schroeder
  - L'utilisation de l'horodatage (timestamps)
  - En séparant le rôle de la tierce partie de confiance en deux services :
    - ☛Le service d'authentification (AS pour Authentication Service)
    - ☛Le générateur de ticket de service (TGS pour Ticket Granting Service)
- Introduction des timestamps
  - Permettent d'introduire des dates d'expiration ce qui limite le rejeu
  - Ils réduisent le nombre total de messages dans le protocole
  - Cela implique la synchronisation horaire de chaque entité participant à la communication (KDC, client, service)

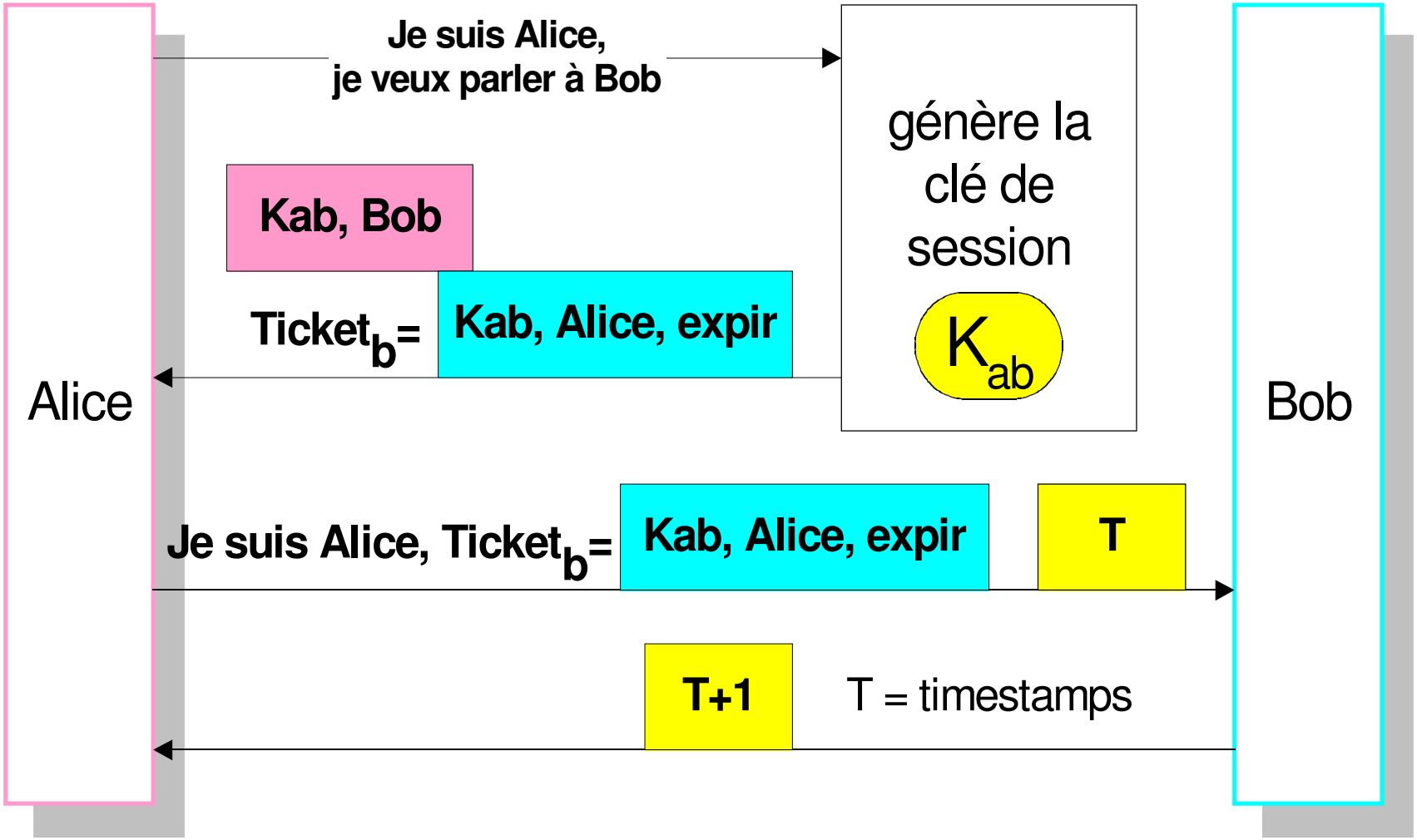
2.2.3 Kerberos

- Kerberos (presque)



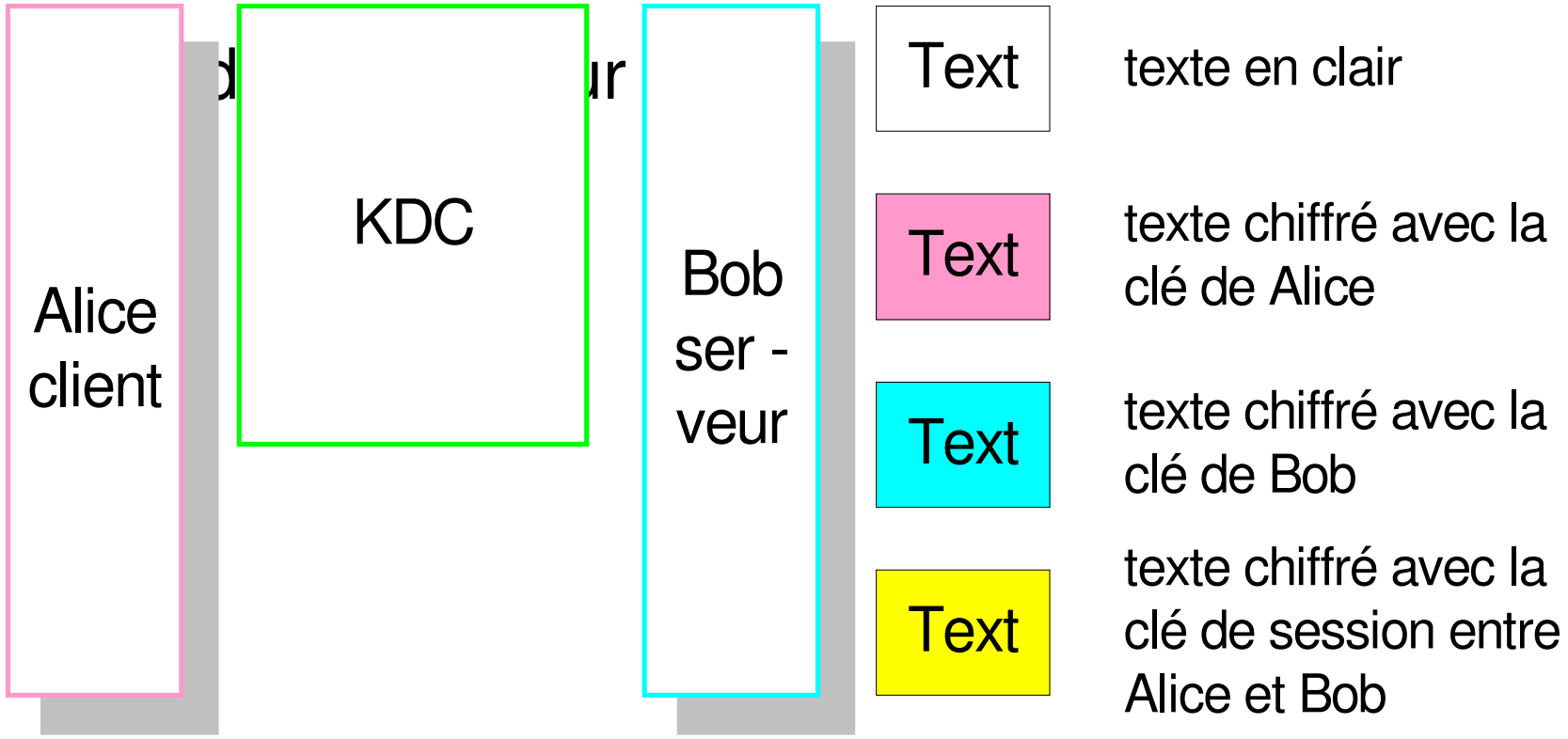
2.2.3 Kerberos

- Kerberos (presque)



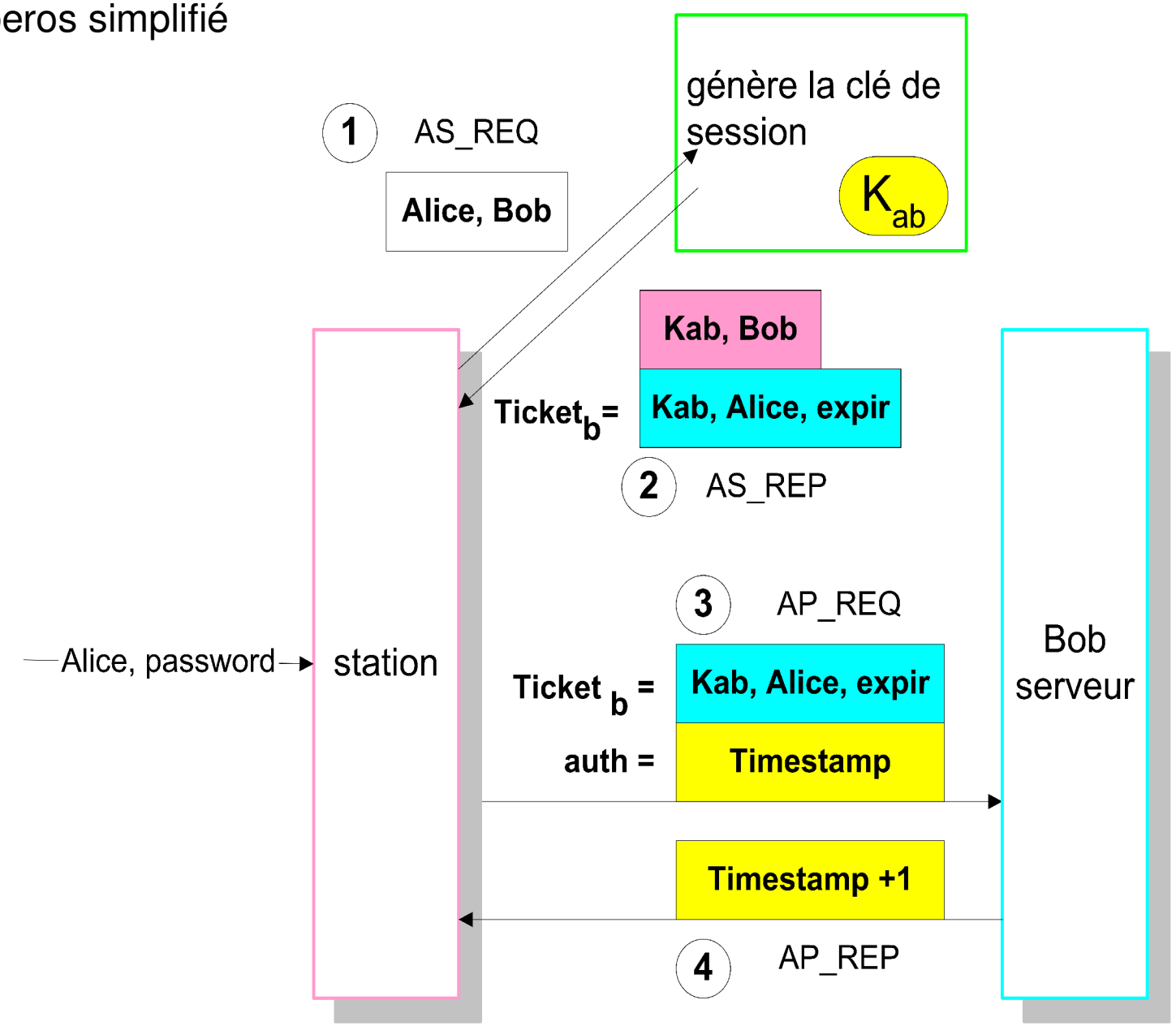
2.2.3 Kerberos

- Kerberos simplifié (sans TGS, Ticket Granting Service)



2.2.3 Kerberos

- Kerberos simplifié



2.2.3 Kerberos

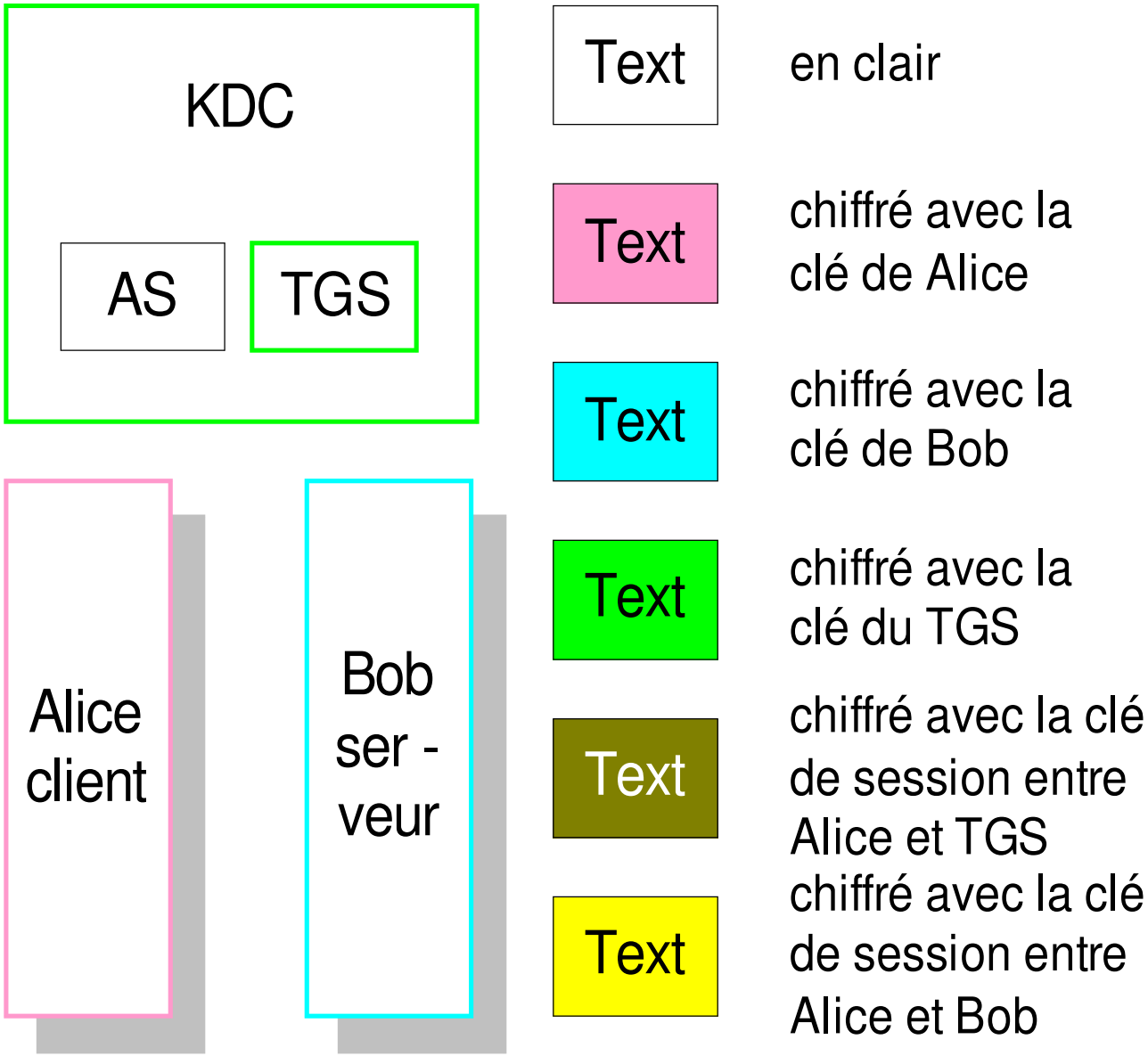
- Le service rendu par le KDC est séparé en deux:
  - Le service d'authentification (AS pour Authentication Service)
  - Le générateur de ticket de service (TGS pour Ticket Granting Service)
- Deux types
  - TGT (Ticket Granting Ticket): ticket d'authentification
  - TS : ticket de service

2.2.3 Kerberos

- Intérêt
  - Permet (avec l'option forwardable) le SSO (Single Sign On)
  - Limite l'utilisation du mot de passe:
    - ☛Moins de données chiffrées avec la clé secrète de l'utilisateur traverse le réseau
    - ☛On limite l'accès aux données susceptible d'être soumises à des attaque offline par dictionnaire

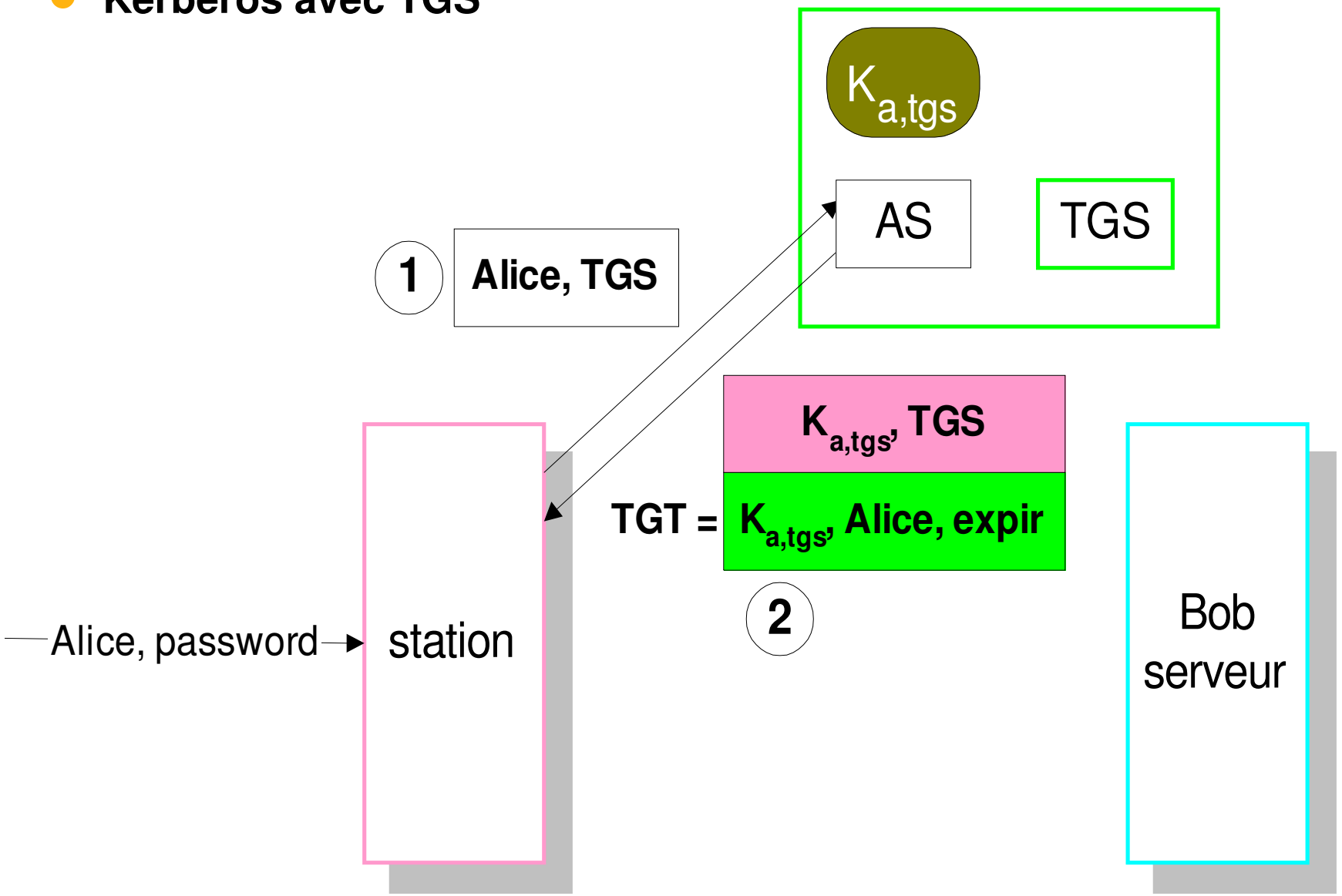
2.2.3 Kerberos

- Code de couleur



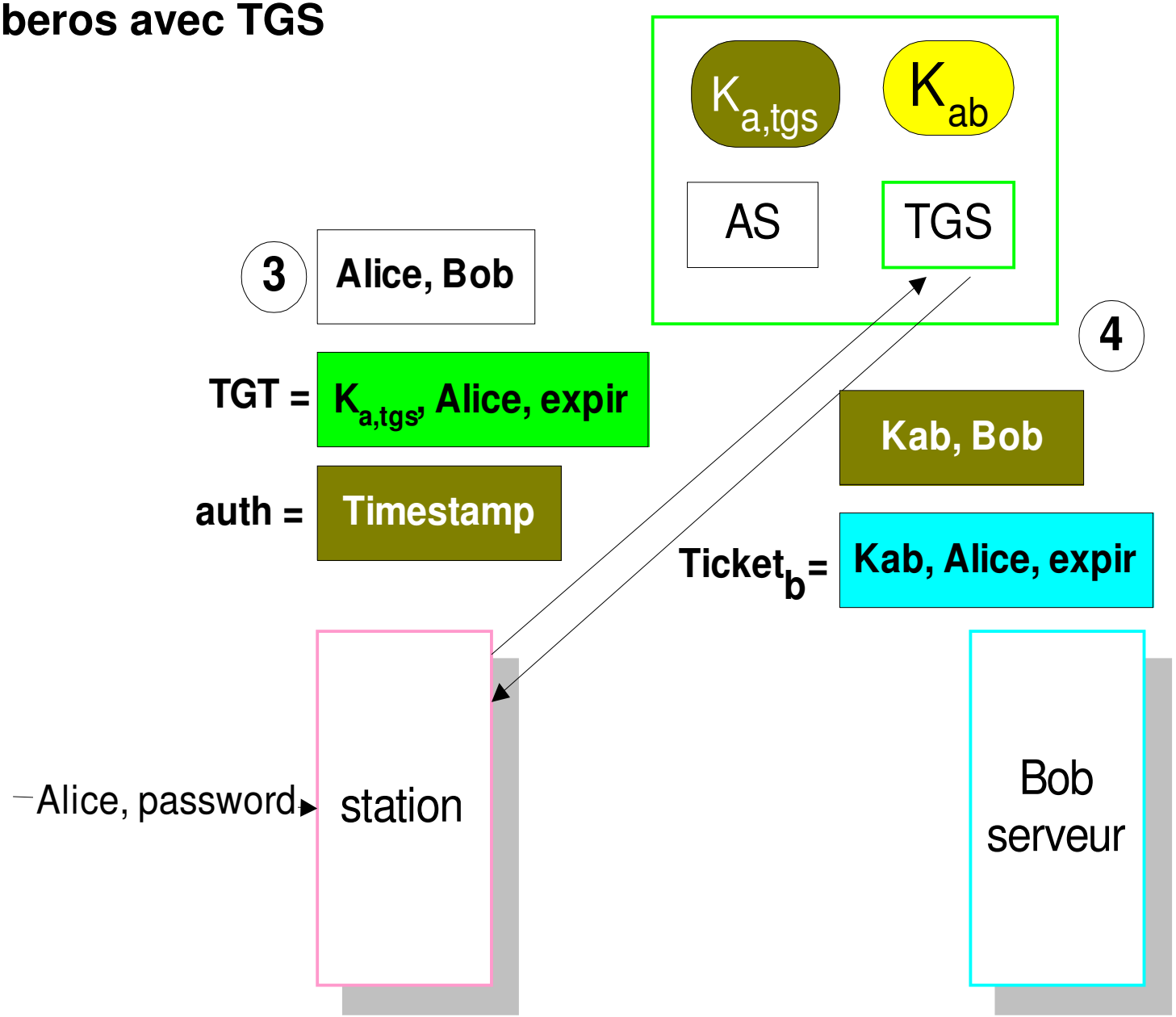
2.2.3 Kerberos

- **Kerberos avec TGS**



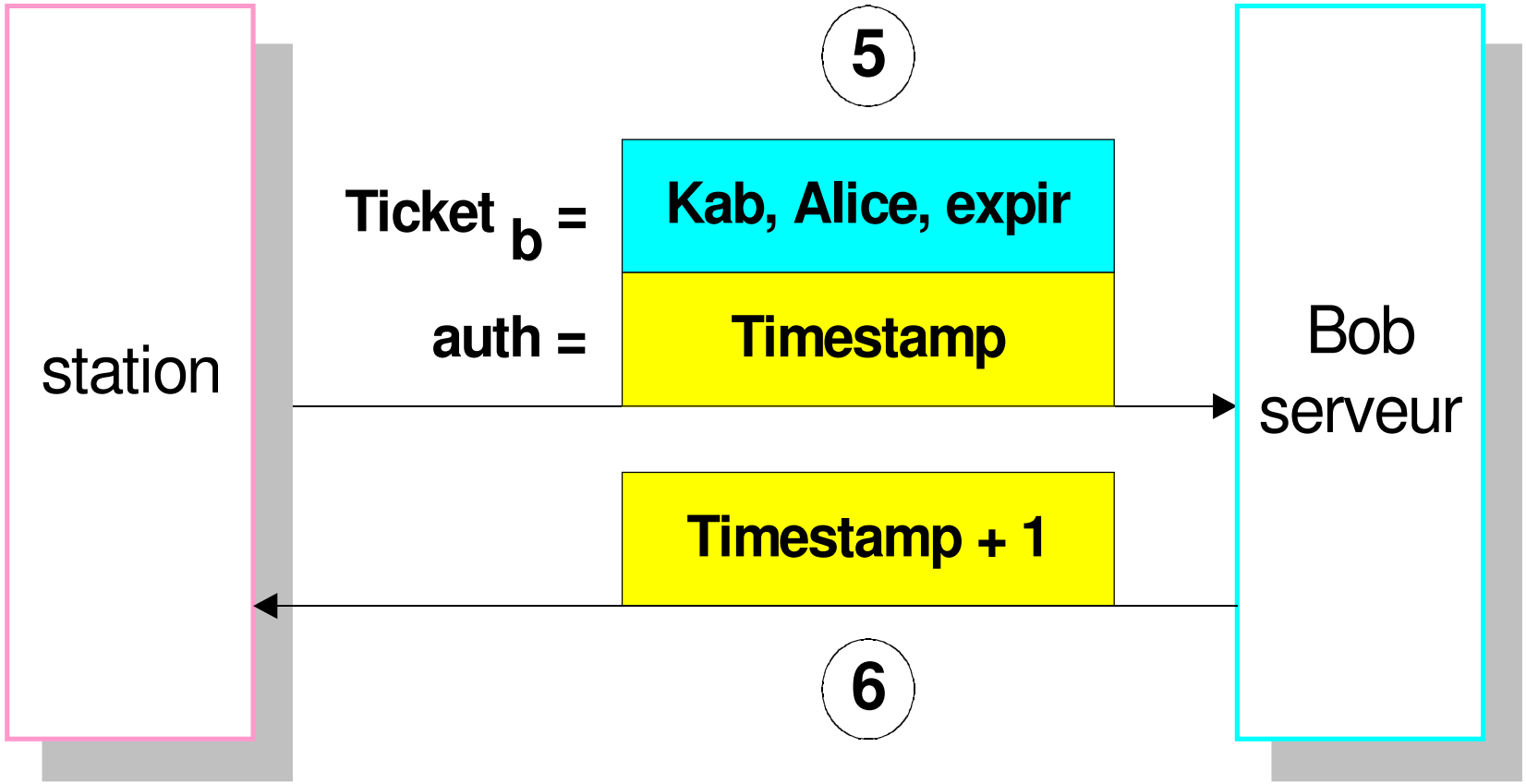
2.2.3 Kerberos

- **Kerberos avec TGS**

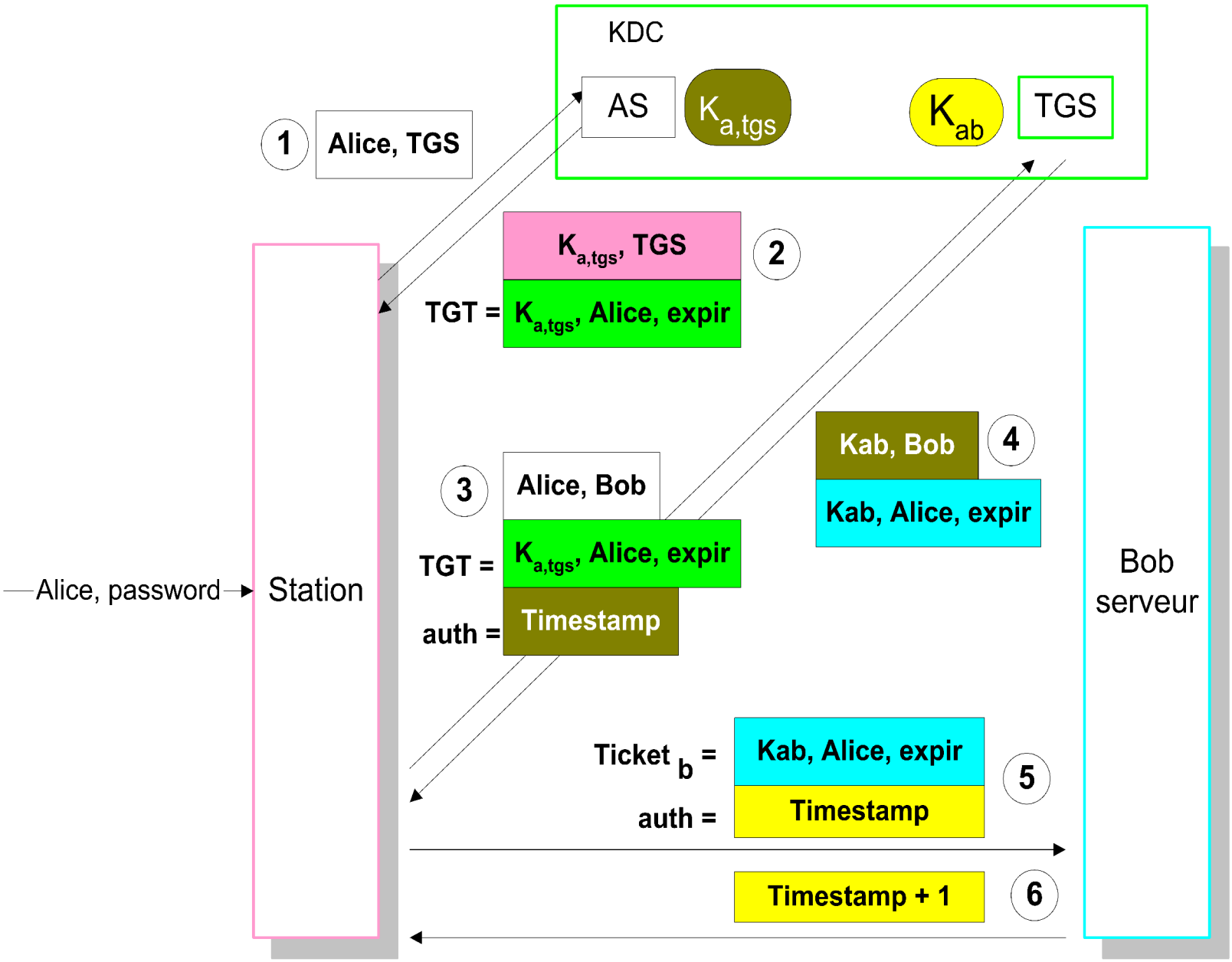


2.2.3 Kerberos

- **Kerberos avec TGS**



2.2.3 Kerberos



TLS/SSL : protocole

- Introduit par netscape
- But: « sécurisation » des transactions :
  - Authentification du serveur
  - Confidentialité des données échangées
  - Intégrité des données échangées
  - Authentification optionnelle du client
- Protocole de niveau application
  - Au dessus de la couche 4
  - Utilisé par la couche 7 (application)
  - Rappel: dans le modèle tcp/ip, on passe directement de la couche transport à la couche application.

TLS/SSL: historique

- Historique des versions publiques
  - 1994: SSL V2 (DES 56 bits, MD5)
  - 1996: SSL V3
  - 1999: TLS V1.0 (annoncé comme SSL 3.1 mais ~ SSL V3.0)
  - Avril 2006: TLS 1.1 (modifications minimes vs 1.0)
  - Aout 2008: TLS 1.2 (RFC 5246: une grosse évolution par rapport à 1.1)