Introduction à la cryptologie

Benoît Morgan

IRIT, INP-ENSEEIHT

2 février 2021

Avant propos

Ce cours à pour objectif de sensibiliser aux propriétés de sécurité couvertes par la cryptographie.

Les principales primitives cryptographiques sont présentées avant de montrer comment les utiliser correctement afin d'obtenir les services de sécurité désirés.

Ces slides sont principalement construites à l'aide du livre "A handbook of applied cryptography" [4].
Des chapitres de ce livres seront cités tout au long de ce cours.

Lecture conseillée pour aller plus loin [1], [2] (Dan Boneh).

Références

- Dan Boneh. *Online Cryptography Course*. Stanford university, 2020.
- Dan Boneh and Victor Shoup. *A Graduate Course in Applied Cryptography*. Stanford university, 2020.
- Lilyu. Surjection injection bijection-fr.svg. `https://commons.wikimedia.org/wiki/File:Surjection_Injection_Bijection-fr.svg`, February 2008.
- Alfred J Menezes, Jonathan Katz, Paul C Van Oorschot, and Scott A Vanstone. *Handbook of applied cryptography*. CRC press, 1996.

Plan du cours

Plan du cours

Terminologie

- Cryptologie = cryptographie + cryptanalyse
  - Cryptographie, du grec *kruptos* (caché) et *graphein* (écrire)
    définition : étude des techniques mathématiques liées aux aspects de la sécurité de l'information comme : la confidentialité ; l'intégrité ; l'authentification des entités et l'intégrité des données.
  - Cryptanalyse
    Découvrir le(s) secret(s), décrypter, se déguiser, etc.

Le chiffrement :
Chiffre, chiffrement, pas chiffrage ni cryptage (sauf au Québec), déchiffrement, clair, cryptogramme

Objectifs de sécurité de la cryptographie

- Confidentialité : protection contre les accès non autorisés à l'information
- Intégrité : protection contre les modification non autorisées de l'information
- Authentification : des données et des entités
  - Authentification des données (authenticité) : authenticité de l'origine, du contenu, validité (date). Authenticité $\Rightarrow$ intégrité
  - Authentification des entités (authentification) : identifications des entités participant à un communication
- Non-répudiation : empêche les entités de nier avoir exécuté des actions passées

D'autres services de sécurité de l'information peuvent être établis à l'aide des objectifs de sécurité précédents ([4] tableau 1.1).

Les fonctions

Fonction

Une fonction est définie par deux ensembles X et Y et une règle f qui associe à tous les éléments de X exactement un élément de Y.

- X est appelé *ensemble de départ* ou *domaine* et Y est appelé *ensemble d'arrivée*
- Si $x \in X$, l'image de x est l'élément de Y associé à x par f et est noté $y = f(x)$
- Si $y \in Y$, un antécédent (préimage) de y est $x \in X \mid f(x) = y$.
  $y \in Y$ n'a pas forcément d'antécédent par f
- Notation standard pour la fonction f de X vers Y
  $f : X \to Y$
- L'ensemble des $y \in Y$ qui ont au moins un antécédent par f est appelé *image* et est noté $\mathrm{Im}(f)$. $\mathrm{Im(f)} = \{x \in Y \mid x \in X \text{ et } f(x) = y\}$

Les fonctions

Fonction injective (*or one-to-one*)

Une fonction $f : X \to Y$ est dite injective si $\forall y \in Y$ il existe **au plus** $x \in X$ tel que $f(x) = y$.

$\forall x \in X \qquad \forall x' \in X \qquad f(x) = f(x') \Rightarrow x = x'$

Fonction surjective (*or onto*)

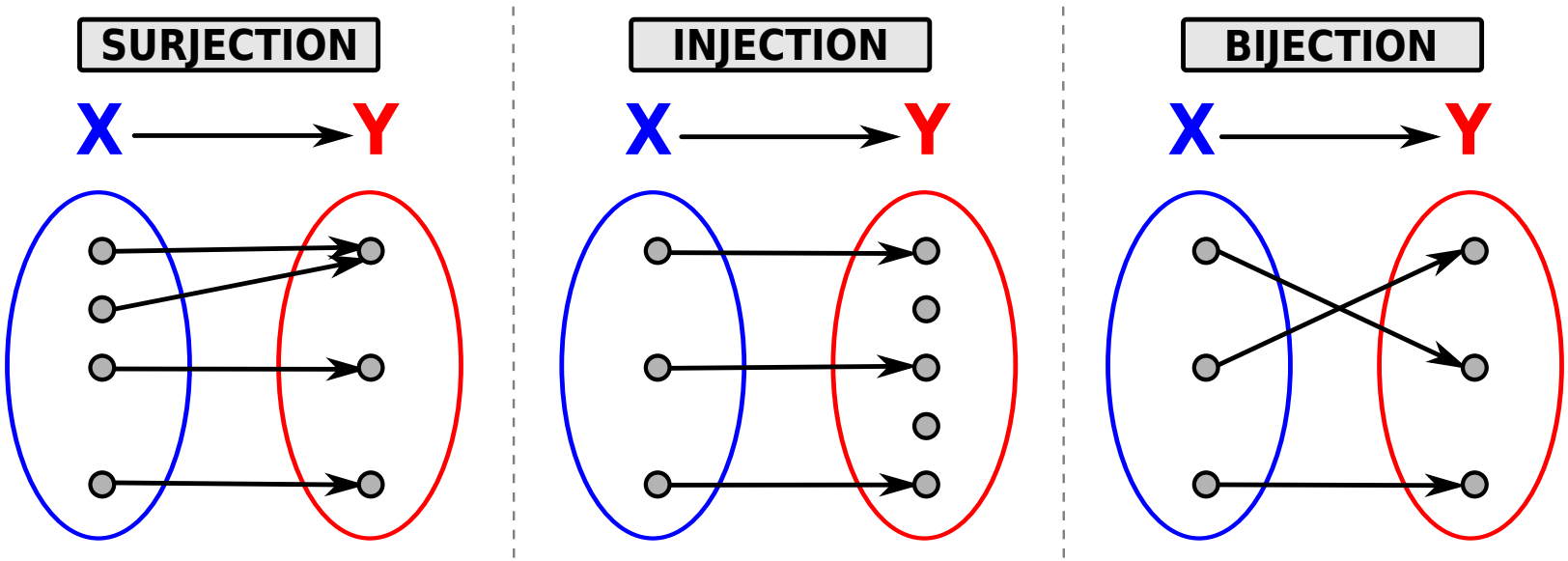
Une fonction $f : X \to Y$ est dite surjective si $\forall y \in Y$ il existe **au moins** $x \in X$ tel que $f(x) = y$.

$\forall x \in X \qquad \exists y \in Y \quad | \quad f(x) = y$

Fonction bijective

Une fonction bijective est une fonction à la fois injective et surjective

Les fonctions



[3]

Bijection et fonction inverse

Fonction inverse d'une bijection

Si f est une bijection de X vers Y, alors il est immédiat de définir la bijection g de Y ver X telle que :
$\forall y \in Y$ définir $g(y) = x$ où $x \in X$ et $f(x) = y$
g est notée f^{-1}

Primitive de chiffrement et f surjective

Soit f une fonction surjective et f^{-1} sa fonction inverse deux transformations secrètes utilisées pour chiffrer des messages.
Si f est surjective sans être une bijection, quelle conséquence pour le chiffrement ?

Bijection et fonction inverse

Fonction inverse d'une bijection

Si f est une bijection de X vers Y, alors il est immédiat de définir la bijection g de Y ver X telle que :
$\forall y \in Y$ définir $g(y) = x$ où $x \in X$ et $f(x) = y$
g est notée f^{-1}

Primitive de chiffrement et f surjective

Soit f une fonction surjective et f^{-1} sa fonction inverse deux transformations secrètes utilisées pour chiffrer des messages.
Si f est surjective sans être une bijection, quelle conséquence pour le chiffrement ?
Plusieurs clairs pour un chiffré... f^{-1} n'est pas une fonction

Bijection et primitive de chiffrement

Primitive de chiffrement : cohérence

Soit E une primitive de chiffrement et $D = E^{-1}$ sa transformation inverse de déchiffrement. E est cohérente si est seulement si E est une fonction bijective.

Fait

Soit la primitive déchiffrement $E : M \to C$ associée à $D = E^{-1}$.
M ensemble des messages clairs. C ensemble des chiffrés.
Si E est cohérent : soit $m \in M$ on a $E(D(m)) = m$

Fonctions à sens unique

Fonction à sens unique : définition informelle [4] 1.12, 1.13

Une fonction f de X vers Y est appelée *fonction à sens unique* (*one-way function*) si $\forall x \in X,\ f(x)$ est "simple" à calculer et qu'au contraire pour la grande majorité des $y \in \mathrm{Im}(f)$, il est "impossible de calculer" un $x \in X \mid f(x) = y$.
Voir .

Exemple 1

Soit la fonction f telle que :
$X = \{1, 2, ..., 16\}$ $f : X \to X$ $f : x \mapsto x^3 \bmod 17$
Soit

x	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
f(x)	1	8	10	13	6	12	3	2	15	14	5	11	4	7	9	16

f est difficile à inverser pour presque tous les éléments à l'exception de 3
Note : Le travail pour calculer $f(x)$ croit moins vite que pour inverser f

Fonctions à sens unique

Exemple 2 (RSA)

- Soit $p = 48611$ et $q = 53991$ nombres premiers et $n = pq = 2624653723$
- Soit $X = \{1, 2, ..., n-1\}$ et la fonction $f : X \to X$, $f : x \mapsto x^3 \ mod \ n$
- Calculer $f(x) = y$ est relativement "simple" alors que $\forall y \in Im(Y)$ calculer $x \mid y = f(x)$ est "difficile"
- C'est à dire calculer la racine cubique de y modulo n :

Note : Néanmoins, si p et q sont connus, alors il existe un algorithme efficace [4] 8.2.2 (i). Cette dernière remarque est caractéristique d'un sous ensemble de fonctions à sens unique fondamental en cryptographie.

Fonctions à sens unique avec trappe

Fonction à sens unique avec trappe

Une fonction à sens unique avec trappe (*trapdoor one-way function*) est une fonction à sens unique $f : X \to Y$ avec la propriété supplémentaire suivante : étant donné une information supplémentaire, appelée trappe, il devient faisable de trouver $\forall y \in \mathrm{Im}(f)$ un $x \in X$ tel que $f(x) = y$

Exemple 3

L'exemple 2 illustre le concept de fonction à sens unique avec trappe.

$f : x \mapsto x^3 \bmod n$, $n = pq$ premiers

- **Trappe** : connaître $n = pq$ permet de calculer f^{-1} [4] 8.2.2 (i).
- p et q peuvent être retrouvés depuis n
- Si n est très grand et $|p| = |q| \Rightarrow$ problème difficile : *problème de factorisation de nombres entiers*

Existence des fonctions à sens unique

Essentielles pour la cryptographique à clé publique.

Existence des fonctions à sens unique

Essentielles pour la cryptographique à clé publique.

Remarque 1

Pas de preuve de l'existence de véritables fonctions à sens unique.
Pas de preuve utilisant des définitions rigoureuses de "simple" et "impossible" à calculer.

Existence des fonctions à sens unique

Essentielles pour la cryptographique à clé publique.

Remarque 1

Pas de preuve de l'existence de véritables fonctions à sens unique.
Pas de preuve utilisant des définitions rigoureuses de "simple" et "impossible" à calculer.

Remarque 2

Existence de fonctions à sens unique non établie $\Rightarrow$ même constat pour les fonctions à sens unique avec trappe.

Existence des fonctions à sens unique

Essentielles pour la cryptographique à clé publique.

Remarque 1

Pas de preuve de l'existence de véritables fonctions à sens unique.
Pas de preuve utilisant des définitions rigoureuses de "simple" et "impossible" à calculer.

Remarque 2

Existence de fonctions à sens unique non établie $\Rightarrow$ même constat pour les fonctions à sens unique avec trappe.

Remarque 3

Il existe tout de même de nombreux bons candidats pour les deux types de fonctions précédentes, que l'on utilise en pratique dans les schémas cryptographiques actuels.

Permutations et involutions

Permutation

Une permutation p est une bijection de S dans S : $p : S \to S$

Souvent utilisées dans des constructions cryptographiques

Exemple 4

Soit $S = \{1, 2, 3, 4, 5\}$ et la permutation $p : S \to S$:

$$p(1) = 3,\ p(2) = 5,\ p(3) = 4,\ p(4) = 2,\ p(5) = 1$$

ou :

$$p = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 5 & 4 & 2 & 1 \end{pmatrix} \quad p^{-1} = \begin{pmatrix} 3 & 5 & 4 & 2 & 1 \\ 1 & 2 & 3 & 4 & 5 \end{pmatrix}$$

Permutations et involutions

Involutions

Une involution f est une bijection de S dans S : $f : S \to S$ telle que $f = f^{-1}$ ou de manière équivalente $\forall x \in S \quad f(f(x)) = x$

Utilisées dans des constructions cryptographiques

Principe de Kerckhoffs

Interprétation

La sécurité d'un système de chiffrement ne doit reposer que sur le secret d'une clé. La clé doit être facilement modifiable.

Erreur passées

- Scytale
- César
- Vigenère
- Enigma
- ...

Domaine des primitives de chiffrement [4] 1.4

Alphabet de définition

- Ensemble fini $\mathcal{A}$
- Exemple l'alphabet binaire : $\mathcal{A} = \{0, 1\}$

Ensemble de départ : clairs

- Ensemble des messages $\mathcal{M}$, chaînes d'un alphabet de définition $\mathcal{A}$
- $m \in \mathcal{M}$ est appelé **un clair**

Ensemble d'arrivée : chiffrés

- Ensemble des chiffrés $\mathcal{C}$, chaînes d'un alphabet de définition $\mathcal{A}$
- $c \in \mathcal{C}$ est appelé **un chiffré**

Fonctions de chiffrement et déchiffrement [4] 1.4

- Un ensemble des clés est noté $\mathcal{K}$
- $e \in \mathcal{K}$ est appelé clé (de chiffrement, de déchiffrement)

Fonction de chiffrement (transformation)

- "chiffrement", "chiffrer un message", ~~"crypter"~~, ~~"encrypter"~~ (QC)
- $e \in \mathcal{K}$ détermine une seule **bijection** $E_e : \mathcal{M} \to \mathcal{C}$

Fonction de déchiffrement

- "déchiffrement", "déchiffrer"
- $d \in \mathcal{K}$ détermine une seule **bijection** $D_d : \mathcal{C} \to \mathcal{M}$

Schéma de chiffrement [4] 1.4

Un schéma de chiffrement se compose de :

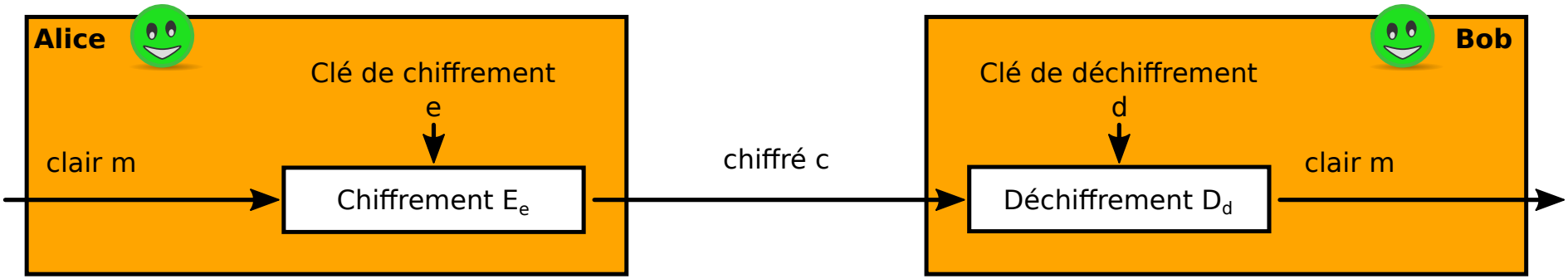
- Un ensemble de transformations de chiffrement $\{E_e \mid e \in \mathcal{K}\}$
- Un ensemble de transformations de déchiffrement $\{D_d \mid d \in \mathcal{K}\}$

$\Rightarrow$ avoir sélectionné l'ensemble des clairs $\mathcal{M}$ et l'ensemble des chiffrés $\mathcal{C}$

Avec les propriétés suivantes :

- $\forall e \in \mathcal{K}\ \exists$ une unique clé $d \in \mathcal{K}$ telle que $D_d = E_e^{-1}$
- On appelle paire de clés (e, d). e peut être égal à d

Confidentialité et communication entre deux entités



- Entité : Alice ou Bob
- Émetteur : Alice
- Récepteur : Bob
- Attaquant : ni Alice ni Bob, noté Eve (*eavesdropper*)
  Objectif : casser les propriétés de sécurité de la communication

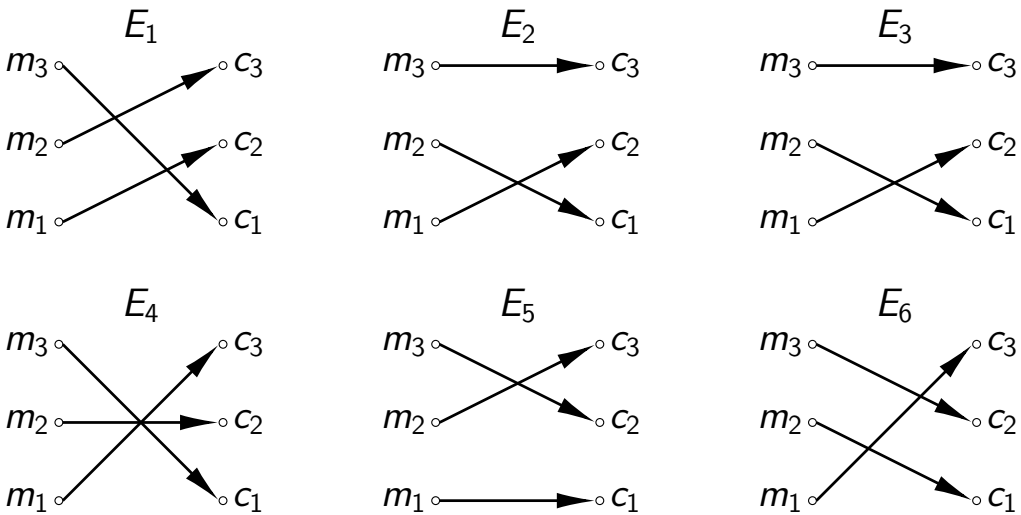
Confidentialité et communication entre deux entités

- Canal : moyen de communication entre Alice et Bob
- Canal sécurisé : l'attaquant à accès au canal mais ne peut lire ou modifier l'information
- Canal non sécurisé : l'attaquant à accès au canal et peut lire et modifier l'information

Schéma de chiffrement

Schéma trivial

- $\mathcal{M} = \{m_1, m_2, m_3\}$
- $\mathcal{C} = \{c_1, c_2, c_3\}$
- Il y a exactement 3! bjiections de $\mathcal{M}$ vers $\mathcal{C}$
- $|\mathcal{K}| = 3!$ $\quad \mathcal{K} = \{1, 2, 3, 4, 5, 6\}$



- Alice et Bob choisissent une clé e qui désigne E_e
- Alice chiffre $m \in \mathcal{M}$ avec $E_e \mid c = E_e(m)$
- Alice déchiffre $c \in \mathcal{C}$ avec $E_e^{-1} \mid m = E_e^{-1}(c)$

Sécurité du chiffrement

Principe de Kerckhoffs (2)

Les ensembles $\mathcal{M}$, $\mathcal{C}$, $\mathcal{K}$, $\{E_e \mid e \in \mathcal{K}\}$, $\{D_d \mid d \in \mathcal{K}\}$ sont publics, c'est à dire connus des participants **de l'attaquant**.
L'unique information secrete entre participants est la paire : (e, d)

Schéma de chiffrement cassé

Un algorithme de chiffrement est dit **cassé** si un attaquant qui ne possède pas la paire (e, d) peut systématiquement retrouver le clair depuis son chiffré correspondant dans **une limite de temps appropriée** inférieure à la force brute.

Cette limite de temps est **fonction de la durée nécessaire de confidentialité de l'information**.

Sécurité du chiffrement (2)

Force brute : recherche exhaustive des clés (e, d)

Parcourir $\mathcal{K}$ à la recherche de d permet par construction de casser un schéma en une durée donnée.

- $\Rightarrow$ $|K|$ doit être dimentionné pour rendre cette approche infaisable
- $\Rightarrow$ Objectif de conception : la recherche exhaustive doit être autant que possible la meilleure attaque d'un schéma de chiffrement

Sécurité de l'information

Quelques éléments sur la sécurité de l'information en général

Service / objectif de sécurité

Méthode pour assurer des propriétés de sécurité de l'information.
Par exemple : intégrité et confidentialité d'un message transmis sur un canal non sécurisé

Attaque

On parle de **casser** un objectif de sécurité

Adversaire

Attaquant passif

Un attaquant passif est capable de seulement **lire** l'information sur un canal non sécurisé

Attaquant actif

Un attaquant passif est capable de **lire et modifier** l'information sur un canal non sécurisé

Cryptanalyse

Étude des techniques mathématiques pour casser des objectifs de sécurité. Par les cryptanalistes (défenceurs ou attaquants !)

Plan du cours

Schéma de chiffrement symétrique

Définition

Un schéma de chiffrement aux transformations $\{E_e \mid e \in \mathcal{K}\}$ et $\{D_d \mid d \in \mathcal{K}\}$ est dit symétrique s'il est "facile" de déterminer pour toutes les paires (e, d) la clé de déchiffrement d depuis la clé de chiffrement e et inversement.

Fait

Dans la plupart des schémas de chiffrement symétriques on a $e = d$, d'où le terme "symétrique".

Exemple

Le chiffrement *One Time Pad*

Schéma de chiffrement symétrique

Exemple (évidemment non sécurisé !)

- $\mathcal{A} = \{A, B, ..., Z\}$. $\mathcal{M}$ et $\mathcal{C}$ sont toutes les chaines possibles de taille 5 sur $\mathcal{A}$.
  $\mathcal{M} = \mathcal{C} = \{(A, A, A, A, A), (A, A, A, A, B), ..., (Z, Z, Z, Z, Z)\}$
- $\mathcal{K}$ est l'ensemble des permutations sur $\mathcal{A}$
- $e \in K$
  Exemple : $e = \begin{pmatrix} A & B & C & D & E & F & G & H & I & ... \\ D & E & F & G & H & I & J & K & L & ... \end{pmatrix}$
- $d = e^{-1} \in \mathcal{K}$
  Exemple : $e = \begin{pmatrix} D & E & F & G & H & I & J & K & L & ... \\ A & B & C & D & E & F & G & H & I & ... \end{pmatrix}$
- $E_e : m_i \mapsto e(m_i) \mid i \in \{1, ..., 5\}$
- $D_d : c_i \mapsto e(c_i) \mid i \in \{1, ..., 5\}$

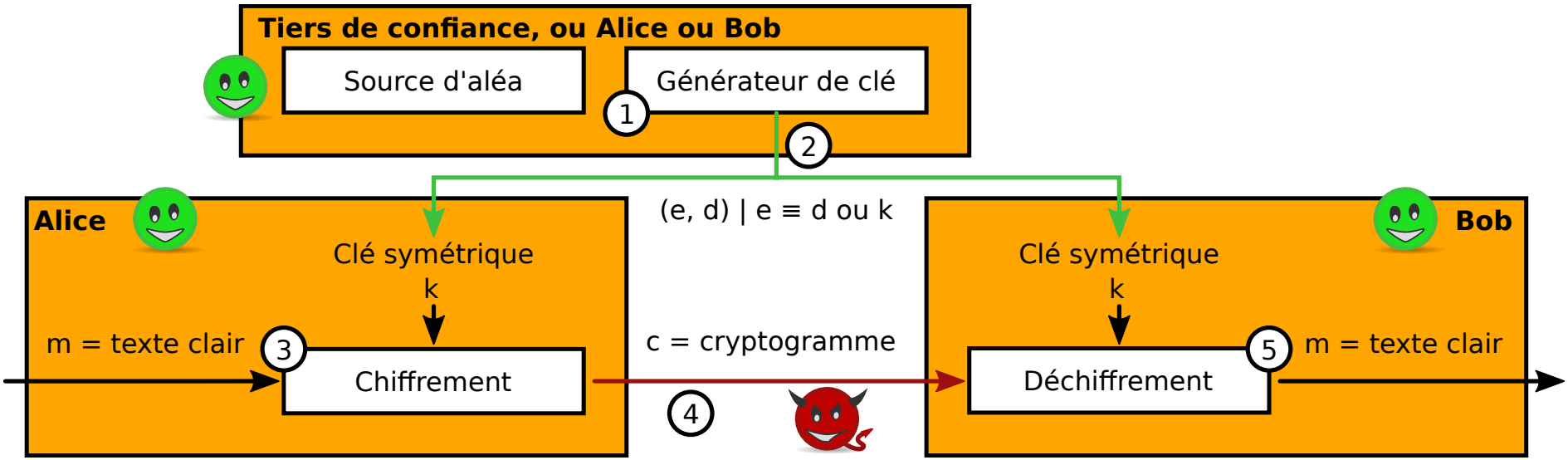
Schéma de chiffrement symétrique

Exemple (évidemment non sécurisé !) (suite)

- Chiffrement : Un clair est découpé en plusieurs clairs de taille 5, avant de leur appliquer la transformation E_e
- Déchiffrement : On applique aux chiffrés la transformation D_d avant de concaténer les clairs de taille 5 en un clair final.
- On bourre le dernier clair découpé si sa taille de n'est pas 5.
- $m =$ `CE CHIFFREMENT N EST PAS SECURISE`
- $m =$ `CECHI FFREM ENTNE STPAS SECUR ISE88`
- $E_e(m) = c =$ `FHFKL IIUHP HQWQH VWSDV VHFXU LVH88`

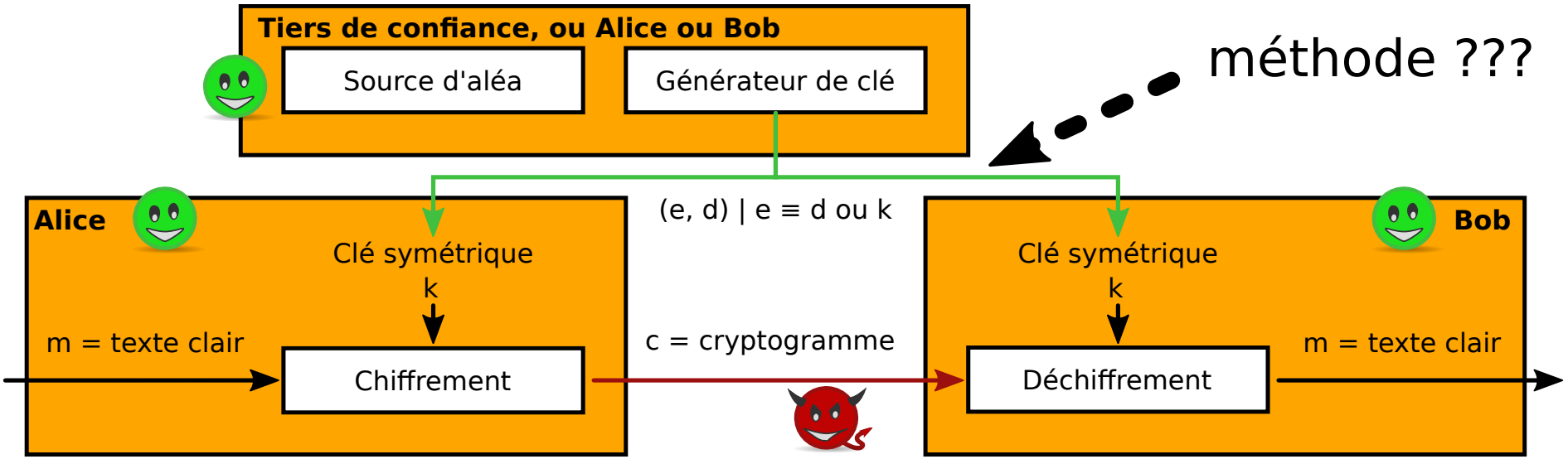
Ici, équivalent bloc du chiffre de César

Communication confidentielle et chiffrement symétrique



1. Génération de la clé k, ou (e, d) avec d déductible de e
2. Distribution de k à l'aide d'un canal sécurisé
3. Alice chiffre le clair m avec la transformation $E_k(m) = c$
4. Alice transmet à Bob le chiffré c sur un canal non sécurisé
5. Bob déchiffre c pour retrouver m

Distribution de clés symétriques



Distribution de la clé : *Quid* du canal sécurisé ?.
Appelé **problème de distribution de clé**

Solution 1 : remise en main propre
Solution 2 : schémas de distribution de clés

Définition

Définition

Un schéma de chiffrement par bloc est un schéma qui découpe les clairs en en chaînes de caractères de taille t sur l'alphabet $\mathcal{A}$, appelées blocs. Les blocq sont chiffrés les uns après les autres.

2 catégories principales

- Chiffrement par substitution
- Chiffrement par transposition

Chiffrement par substitution monoalphabétique (1) [4] 1.5.2

Définition

Soit $\mathcal{A}$ un alphabet de q symboles et $\mathcal{M}$ l'ensemble de toutes les chaînes de charactères de taille t sur $\mathcal{A}$. Soit $\mathcal{K}$ toutes les permutations sur $\mathcal{A}$. Pour tout $e \in \mathcal{K}$ définir une transformation de chiffrement E_e telle que :

$$E_e(m) = (e(m_1)e(m_2)e(m_3)\dots e(m_t)) = (c_1c_2c_3\dots c_t) = c$$

avec $m = (m_1m_2m_3\dots m_t) \in \mathcal{M}$

Pour déchiffrer $c = (c_1c_2c_3\dots c_t)$, calculer la permutation inverse $d = e^{-1}$:

$$E_d(c) = (d(c_1)d(c_2)d(c_3)\dots d(c_t)) = (m_1m_2m_3\dots m_t) = m$$

L'exemple précédent est un chiffrement par subsitution monoalphabétique

Chiffrement par substitution monoalphabétique (2) [4] 1.5.2

Nombre de clés

$|\mathcal{K}| = ?$

Chiffrement par substitution monoalphabétique (2) [4] 1.5.2

Nombre de clés

$|\mathcal{K}| = q!$

Independant de la taille du bloc t

Exemple : $\mathcal{A} = \{A, B, C, D, \ldots, Z\}$, soit

$|\mathcal{K}| = 26! = 403291461126605635584000000 \approx 10^{26}$ clés

Chiffrement par substitution monoalphabétique (2) [4] 1.5.2

Nombre de clés

$|\mathcal{K}| = q!$
Independant de la taille du bloc t
Exemple : $\mathcal{A} = \{A, B, C, D, \ldots, Z\}$, soit
$|\mathcal{K}| = 26! = 403291461126605635584000000 \approx 10^{26}$ clés

Attaques

Distribution de la fréquence d'apparition des lettres préservée par cette transformation.
L'observation d'une faible quantité de chiffrés ($\ll 10^{26}$), permet d'inverser la transformation de chiffrement.

Chiffrement par substitution polyalphabétique (1) [4] 1.5.2

Définition

Un chiffrement par substitution polyalphabétique est un chiffrement par bloc de taille de bloc t sur un alphabet $\mathcal{A}$ tel que :

- L'espace des clés comprend tous les tuples de t permutations $(p_1, p_2, p_3, \ldots, p_t)$ ou $p_i : \mathcal{A} \to \mathcal{A}$
- La transformation de chiffrement E_e sur le message $m = (m_1, m_2, m_3, \ldots, m_t)$ avec la clé $e = (p_1, p_2, p_3, \ldots, p_t)$ est définie par :

$$E_e(m) = (p_1(m_1)p_2(m_2)p_3(m_3)\ldots p_t(m_t))$$

- La clé de déchiffrement associées à e est $d = (p_1^{-1}, p_2^{-1}, p_3^{-1}, \ldots, p_t^{-1})$
- La transformation de déchiffrement $D_d = E_d$

Chiffrement par substitution polyalphabétique (2) [4] 1.5.2

Chiffre de Vigenère

- $\mathcal{A} = \{A, B, ..., Z\}$. $\mathcal{M}$ et $\mathcal{C}$ sont toutes les chaines possibles sur $\mathcal{A}$ telles que $t = 3$
- $e = (p_1, p_2, p_3)$ telles que p_1 décale de 3 à droite dans l'alphabet, p_2 de 7 et p_3 de 10.
- Chiffrement : Un clair est découpé en plusieurs clairs de taille $t = 3$, avant de leur appliquer la transformation E_e
- Déchiffrement : On applique aux chiffrés la transformation D_d avant de concaténer les clairs de taille t en un clair final.
- On bourre le dernier clair découpé si sa taille de n'est pas t.

Chiffrement par substitution polyalphabétique (3) [4] 1.5.2

Chiffre de Vigenère (suite)

- $m =$ CE CHIFFREMENT N EST PAS SECURISE
- $m =$ CEC HIF FRE MEN TNE STP ASS ECU RIS E88
- $E_e(m) = c =$ FLM KPP IYO PLX WUO VAZ DZC HJE UPC H88

Remarque : la clé (p_1, p_2, p_3) peut être encodée par la chaîne de taille 3 $e =$ DHK telle que e_i représente la permutation de décalage de e_i lettres par rapport a la première lettre de l'alphabet (A).

Chiffre de Vigenère (suite)

- $m =$ CEC HIF FRE MEN TNE STP ASS ECU RIS E88
- $e =$ DHK DHK DHK DHK DHK DHK DHK DHK DHK DHK
- $E_e(m) = c =$ FLM KPP IYO PLX WUO VAZ DZC HJE UPC H88

Chiffrement par substitution polyalphabétique (4) [4] 1.5.2

Attaques

Distribution de la fréquence d'apparition des lettres préservée par chaque permutation p_i

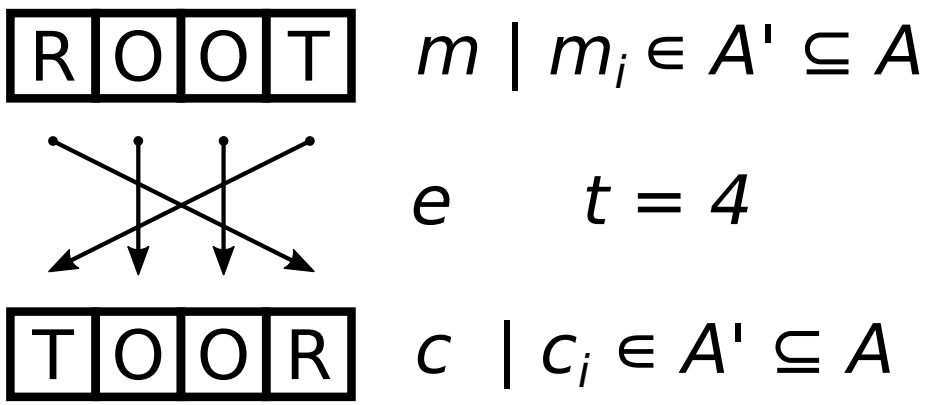
p_i se répète modulo t

Si on connait t, on peut refaire un analyse fréquencielle chaque groupe de chiffrés modulo t et retrouver la permutation p_i avec une faible quantité de chiffrés.

Chiffrement par transposition (1)

Définition

Permute les symboles dans un bloc de taille t
Produit des anagrammes d'un mot



Chiffrement par transposition (2)

Définition

Soit un schéma de chiffrement symétrique par blocs de taille t
Soit $\mathcal{K}$ l'ensemble de toutes les permutations sur l'ensemble $\{1, 2, 3, \ldots, t\}$
Définir la transformation de chiffrement E_e telle que :

$$E_e(m) = (m_{e(1)} m_{e(2)} m_{e(3)} \ldots m_{e(t)}) = c$$

avec $m = (m_1 m_2 m_3 \ldots m_t) \in \mathcal{M}$.
La clé de déchiffrement correspond à la permutation inverse $d = e^{-1}$
Pour déchiffrer $c = (c_1, c_2 c_3 \ldots c_t)$, définir D_d telle que :

$$D_d(c) = (c_{d(1)} c_{d(2)} c_{d(3)} \ldots c_{d(t)}) = m$$

Composition de chiffrements (1)

Afin de pouvoir définir les schémas de chiffrement par produit il est nécessaire de faire quelques rappels sur la composition de fonctions.

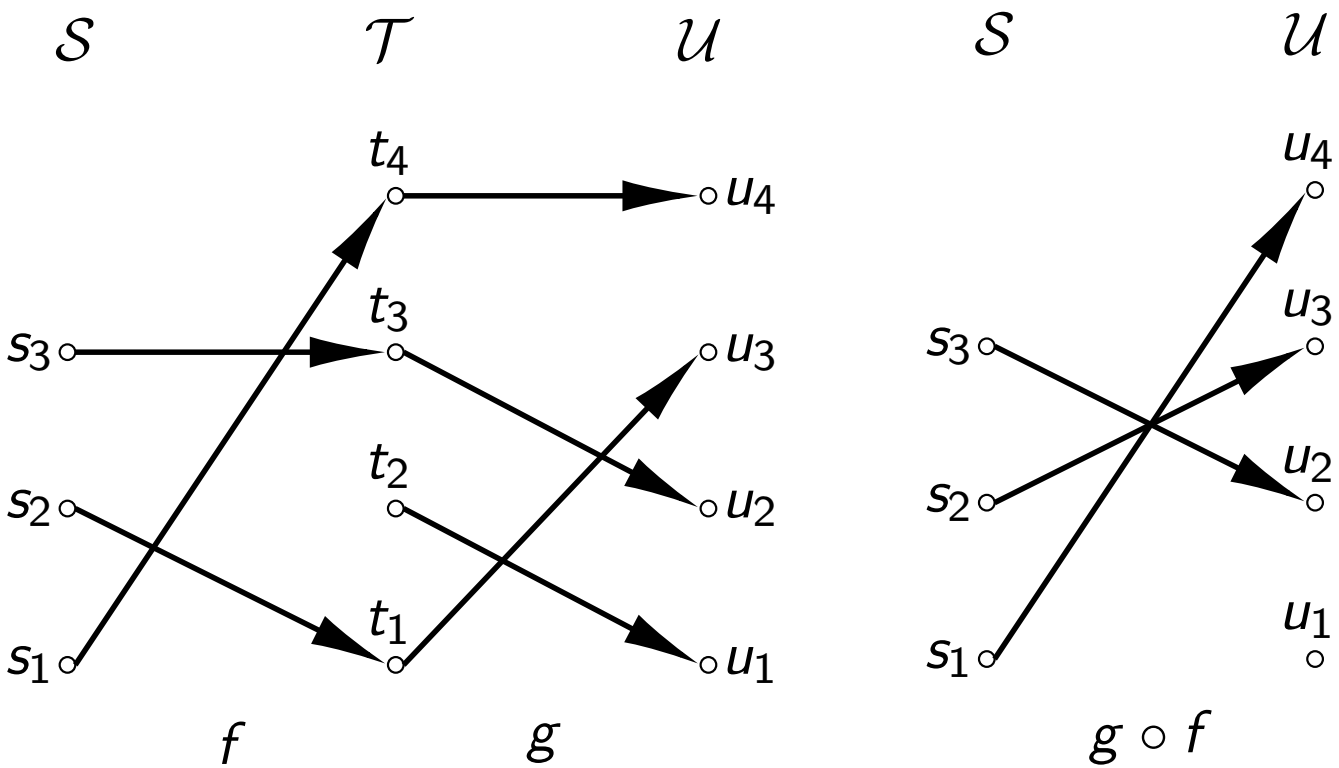
Définition : composition de fonctions

Soient $\mathcal{S}, \mathcal{T}, U$ des ensembles finis et soient $f : \mathcal{S} \to \mathcal{T}$ et $g : \mathcal{T} \to \mathcal{U}$ deux fonctions quelconques.
La composition de g avec f, notée $g \circ f$, est une fonction de $\mathcal{S}$ dans $\mathcal{U}$ définie telle que :

$$(g \circ f)(x) = g(f(x)) \; \forall x \in \mathcal{S}$$

La composition peut-être facilement étendue à plus de 2 fonctions à condition que les ensembles de départ de d'arrivée correspondent.

Composition de chiffrements (2)



Composition de chiffrements (3)

Chiffrement par produit : idée

- Les chiffrement par substitution et transposition seuls n'apportent que peu de sécurité
- **Intuition** : la composition bien choisie de transformations simples permet d'obtenir des transformations fortes !

Chiffrement par produit : définition informelle

Soient $E_{k_1}^{(1)}\ E_{k_2}^{(2)}\ E_{k_3}^{(3)} \dots\ E_{k_n}^{(n)}$ des permutations ou des substitutions :

$$E_e(x) = E_{e_n}^{(n)} \circ \dots E_{e_3}^{(3)} \circ E_{e_2}^{(2)} \circ E_{e_1}^{(1)} = c$$

$$E_e^{-1}(c) = E_{e_1}^{(1)^{-1}} \circ E_{e_2}^{(2)^{-1}} \circ E_{e_3}^{(3)^{-1}} \circ \dots E_{e_n}^{(n)^{-1}} = m$$

Composition de chiffrements (4)

Remarque : il est dit qu'une substitution ajoute de la *confusion* et qu'une permutation ajoute de la *diffusion* à la transformation composée.

- Diffusion : si un bit change dans l'antécédant $t/2$ change en sortie ;
- Confusion : complexifier la relation entre une clé un chiffré. Un bit de chiffré doit dépendre d'un maximum de bits de clé.

Chiffrement par produit : tour

Tour ou *round* : série de compositions de transformations simples.
On utilise n tours pour obtenir le "niveau" de confusion et de diffusion nécessaire pour le niveau de sécurité choisi.
Au moins une de ces transformations utilise une clé symétrique (Kerckhoffs)

Composition de chiffrements (5)

Dans l'exemple qui suit un tour est la composition d'une transposition et d'une substitution

Exemple : produit simple

Soient $\mathcal{M} = \mathcal{C} = \mathcal{K}$ toutes les chaines binaires possibles de taille $t = 6$. $|M| = 64$. Soit $m = (m_1 m_2 m_3 \dots m_6) \in \mathcal{M}$ définir :

$$E_k^{(1)}(m) = m \oplus k, \text{ where } k \in \mathcal{K}$$

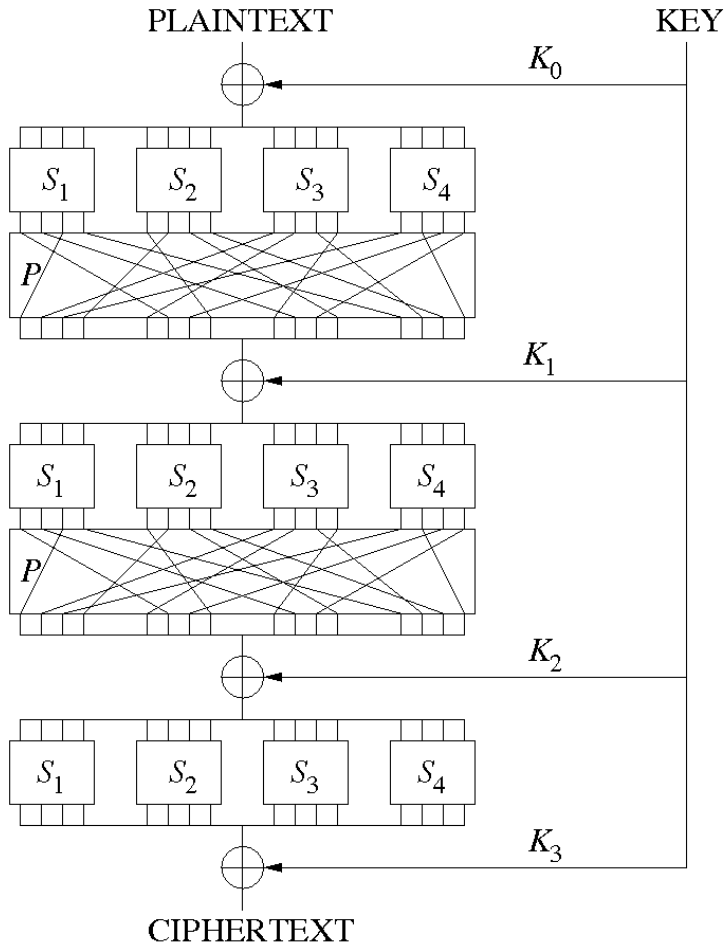
$$E^{(2)}(m) = (m_4 m_5 m_6 m_1 m_2 m_3)$$

Le tour est défini par :

$$E_k = E_k^{(1)} \circ E^{(2)}$$

Remarque : $E^{(2)}$ ne dépend pas de la clé. Ce n'est pas tout le temps le cas !

Réseau de substitution permutation (1)



https://upload.wikimedia.org/wikipedia/commons/c/cd/SubstitutionPermutationNetwork2.png

Réseau de substitution permutation (2)

Définition

Défini sur $\mathcal{A} = \{0, 1\}$, $\mathcal{M} = \mathcal{C} = \mathcal{A}^t$ et $k \in \mathcal{K} = \{0, 1\}^g$.
Chiffrement par produit à la structure générique suivante à n-tours.

Tour r

Défini sur $\mathcal{M} = \mathcal{C} = \mathcal{A}^t$ et $k_r = \mathrm{KDF}(k, r) \mid k_r \in \mathcal{A}^t, r \in [0; n-1]$:

1. Chiffrement par substitution polyalphabétique.
2. Substitution polyalphabétique.
3. Transposition.

Remarque : ici la permutation est une transposition.
Exemple : AES

Réseau de substitution permutation (3)

1. Chiffrement par substitution polyalphabétique

Défini sur $\mathcal{M} = \mathcal{C} = \mathcal{A}^t$ et $k_r \in \mathcal{A}^t$:
$E_{k_r}(m) = m \oplus k_r = c$

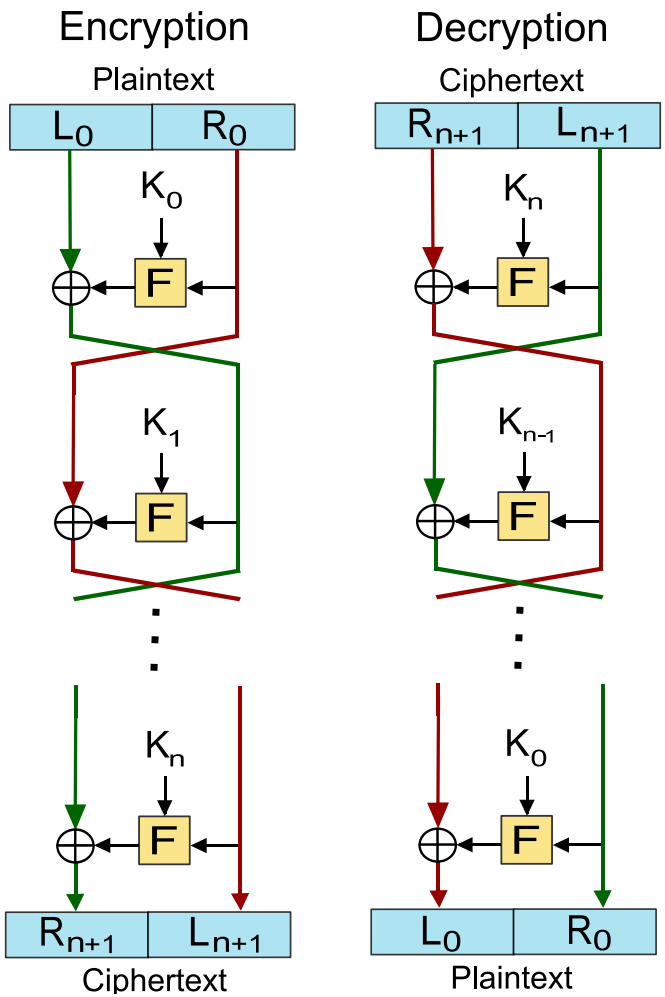
2. Substitution polyalphabétique : boite-S (S-box)

Défini sur $\mathcal{M} = \mathcal{C} = \mathcal{A}^b \mid t/b = s, t \equiv 0 \pmod{b}$
$e = (p_1, p_2, p_3, \ldots, p_s)$ ou $p_i : \mathcal{A}^b \to \mathcal{A}^b$. e est connue de l'adversaire :
$E(m) = c = (p_1(m_1)p_2(m_2)p_3(m_3)\ldots p_s(m_s))$

3. Transposition

Défini sur $\mathcal{M} = \mathcal{C} = \mathcal{A}^t$ et e une permutation sur $\{1, 2, 3, \ldots t\}$.
e est connue de l'adversaire :
$E_e(m) = (m_{e(1)}m_{e(2)}m_{e(3)}\ldots m_{e(t)}) = c$

Réseau de Feistel (1)



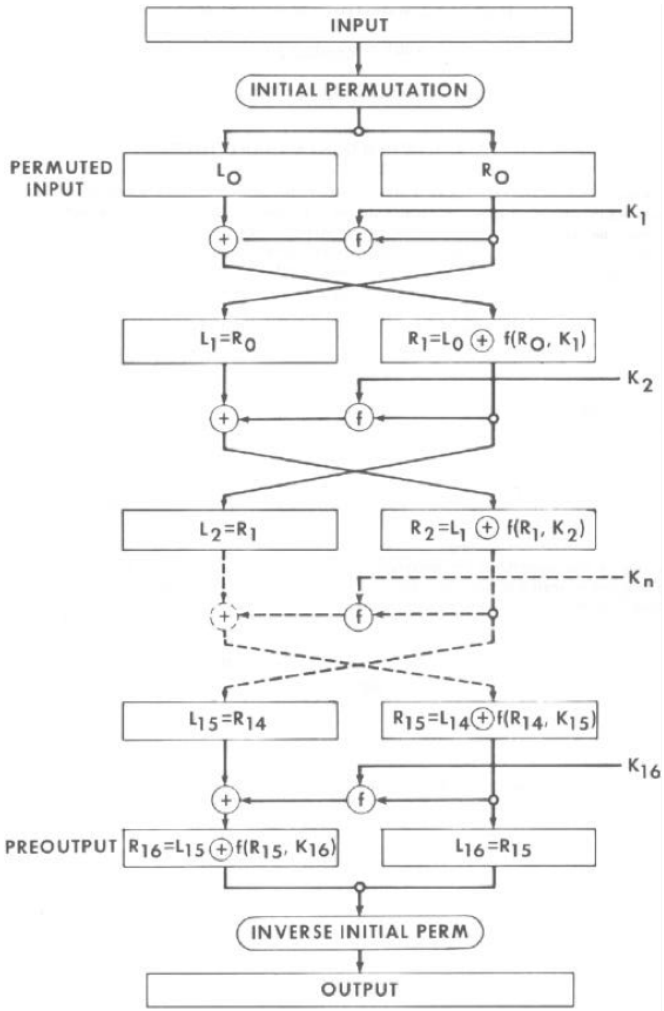
https://upload.wikimedia.org/wikipedia/commons/a/ab/Feistel_cipher_diagram.svg

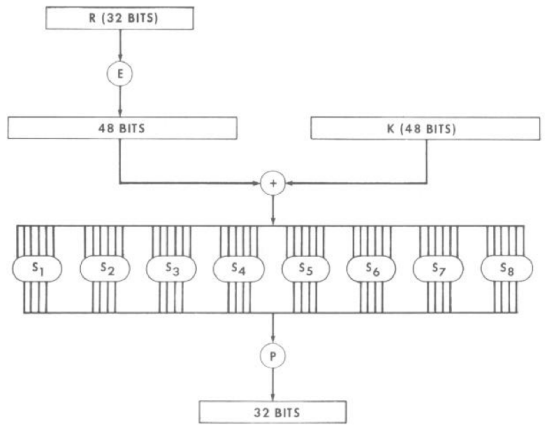
DES, Data Encryption Standard (1)

`csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf`

1. Diversification de la clé $\rightarrow$ 16 sous clés $K_{1..16}$ de 48 bits
   Chaque K_i est composé de 48 bits de K pris dans un certain précis
2. Fractionnement du texte en blocs $B_{1..n}$ de 64 bits
3. Pour chaque bloc B_j
   - 3.1 Permutation initiale du bloc B_j
   - 3.2 Découpage du bloc B_j en parties gauche G_0 et droite D_0
   - 3.3 Pour chaque sous clé, K_i
     - 3.3.1 $G_i = D_{i-1}$
     - 3.3.2 $D_i = G_{i-1} \oplus f(D_{i-1}, K_i)$
   - 3.4 Reconstitution du bloc B'_j à partir de G_{16} et D_{16}
   - 3.5 Permutation initiale inverse du bloc B'_j

DES, Data Encryption Standard (2)





Fonction $f(R, K)$

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

Permutation initiale

Chiffrement par flot (1)

Description

- Un chiffrement par flot peut être défini comme un chiffre par bloc de taille $t = 1$
- Autrement dit chaque élément du clair $\in \mathcal{A}$ peut être chiffré indépendamment
- Autrement dit la transformation appliquée peut changer pour chaque élément du clair

Avantages

- réseau : pas de propagation des erreurs
- ressources : traitement symbole par symbole

Chiffrement par flot (2)

Définition : flot de clés ou *keystream*

Une séquence d'éléments $e_1 e_2 e_3 \ldots l_i \in \mathcal{K}$ est appelée un flot de clés.

Définition : chiffrement par flot

Soit $\mathcal{A}$ un alphabet de q symboles. et E_e un chiffre par substitution à la taille de bloc $t = 1$ et $e \in \mathcal{K}$.
Soit $m_1 m_2 m_3 \ldots$ une chaine de symboles clairs et $e_1 e_2 e_3 \ldots$ un flot $\in \mathcal{K}$.
Un chiffrement par flot prend la chaîne de clairs et la transforme une chaîne de chiffrés telle que :

$$c_i = E_{e_i}(m_i)$$

Si d_i est la transformation inverse de e_i, alors pour déchiffrer la chaîne de chiffrés :

$$D_{d_i}(c_i) = m_i$$

Le *One Time Pad* (1)

Originalement appelé chiffre de Vernam appelé aussi le *One Time Pad*

Définition : transformation de chiffrement

Chiffrement par flot sur $\mathcal{A} = \{0, 1\}$.
$m_1 m_2 m_3 \dots m_t$ est chiffré avec $k_1 k_2 k_3 \dots k_t$ vers $c_1 c_2 c_3 \dots c_t$ tel que :

$$c_i = m_i \oplus k_i,\ 1 \leq i \leq t$$

Un flot de clé est **tiré au hasard** $(1/|\mathcal{K}|)$ et n'est jamais réutilisé

Exactement deux substitutions :

- $E_0 = e_0(m_i) \quad e_0 = m_i \oplus 0 = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}$
- $E_1 = e_1(m_i) \quad e_1 = m_i \oplus 1 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

Le *One Time Pad* (2)

Remarque : e_0 et e_1 sont des involutions.

Le *One Time Pad* (2)

Remarque : e_0 et e_1 sont des involutions.
$\Rightarrow$ exactement deux substitutions inverses équivalentes

Le *One Time Pad* (2)

Remarque : e_0 et e_1 sont des involutions.
$\Rightarrow$ exactement deux substitutions inverses équivalentes

Définition : transformation de déchiffrement

$c_1 c_2 c_3 \dots c_t$ est déchiffré avec $k_1 k_2 k_3 \dots k_t$ vers $m_1 m_2 m_3 \dots m_t$ tel que :

$$m_i = c_i \oplus k_i,\ 1 \leq i \leq t$$

Le *One Time Pad* (3)

Propriété de sécurité du *xor*

Soient $c = c_1 c_2 c_3 \dots c_n$, $m = m_1 m_2 m_3 \dots$ et $k = k_1 k_2 k_3 \dots k_n$
$P[c_i = 1]$? $c_i \in c$ à la position i

```
mi : 0 1 0 1
ki : 0 0 1 1
-----------
ci : 0 1 1 0
```

Le *One Time Pad* (3)

Propriété de sécurité du *xor*

Soient $c = c_1 c_2 c_3 \dots c_n$, $m = m_1 m_2 m_3 \dots$ et $k = k_1 k_2 k_3 \dots k_n$
$P[c_i = 1]$? $c_i \in c$ à la position i

```
mi : 0 1 0 1
ki : 0 0 1 1
-----------
ci : 0 1 1 0
```

$$P[C_i = 1] = P[K_i = 0 \cap m_i = 1] \quad + P[K_i = 1 \cap m_i = 0]$$

Le *One Time Pad* (3)

Propriété de sécurité du *xor*

Soient $c = c_1 c_2 c_3 \dots c_n$, $m = m_1 m_2 m_3 \dots$ et $k = k_1 k_2 k_3 \dots k_n$
$P[c_i = 1]$? $c_i \in c$ à la position i

```
mi : 0 1 0 1
ki : 0 0 1 1
-----------
ci : 0 1 1 0
```

$$\begin{aligned} P[C_i = 1] &= P[K_i = 0 \cap m_i = 1] &&+P[K_i = 1 \cap m_i = 0] \\ &= P[K_i = 0] \times P[m_i = 1] &&+P[K_i = 1] \times P[m_i = 0] \end{aligned}$$

Le *One Time Pad* (3)

Propriété de sécurité du *xor*

Soient $c = c_1 c_2 c_3 \dots c_n$, $m = m_1 m_2 m_3 \dots$ et $k = k_1 k_2 k_3 \dots k_n$
$P[c_i = 1]$? $c_i \in c$ à la position i

```
mi : 0 1 0 1
ki : 0 0 1 1
-----------
ci : 0 1 1 0
```

$$\begin{aligned} P[C_i = 1] &= P[K_i = 0 \cap m_i = 1] && + P[K_i = 1 \cap m_i = 0] \\ &= P[K_i = 0] \times P[m_i = 1] && + P[K_i = 1] \times P[m_i = 0] \\ &= 1/2 \times p && + 1/2 \times (1 - p) \end{aligned}$$

Le *One Time Pad* (3)

Propriété de sécurité du *xor*

Soient $c = c_1 c_2 c_3 \dots c_n$, $m = m_1 m_2 m_3 \dots$ et $k = k_1 k_2 k_3 \dots k_n$
$P[c_i = 1]$? $c_i \in c$ à la position i

```
mi : 0 1 0 1
ki : 0 0 1 1
-----------
ci : 0 1 1 0
```

$$\begin{aligned} P[C_i = 1] &= P[K_i = 0 \cap m_i = 1] && + P[K_i = 1 \cap m_i = 0] \\ &= P[K_i = 0] \times P[m_i = 1] && + P[K_i = 1] \times P[m_i = 0] \\ &= 1/2 \times p && + 1/2 \times (1 - p) \\ &= 1/2 = P[C_i = 0] \end{aligned}$$

Le *One Time Pad* (3)

Propriété de sécurité du *xor*

Soient $c = c_1 c_2 c_3 \dots c_n$, $m = m_1 m_2 m_3 \dots$ et $k = k_1 k_2 k_3 \dots k_n$
$P[c_i = 1]$? $c_i \in c$ à la position i

```
mi : 0 1 0 1
ki : 0 0 1 1
-----------
ci : 0 1 1 0
```

$$\begin{aligned} P[C_i = 1] &= P[K_i = 0 \cap m_i = 1] & +P[K_i = 1 \cap m_i = 0] \\ &= P[K_i = 0] \times P[m_i = 1] & +P[K_i = 1] \times P[m_i = 0] \\ &= 1/2 \times p & +1/2 \times (1-p) \\ &= 1/2 = P[C_i = 0] \end{aligned}$$

Propriété : si K_i est uniforme et indépendant de M_i alors C_i est uniforme

Le *One Time Pad* (4)

Propriété de sécurité du *One Time Pad*

$\mathcal{M} = \mathcal{K} = \mathcal{C} = \{0,1\}^2$

```
m : 0 1 | 0 0 | 1 1 | 1 0
k : 0 0 | 0 1 | 1 0 | 1 1
-------------------------
c : 0 1 | 0 1 | 0 1 | 0 1
```

Le *One Time Pad* (4)

Propriété de sécurité du *One Time Pad*

$\mathcal{M} = \mathcal{K} = \mathcal{C} = \{0,1\}^2$

```
m : 0 1 | 0 0 | 1 1 | 1 0
k : 0 0 | 0 1 | 1 0 | 1 1
-------------------------
c : 0 1 | 0 1 | 0 1 | 0 1
```

Pour un chiffré donné, tout clair peut être un antécédent

Le *One Time Pad* (5)

Démonstration tirée du cours de Vincent Migliore (INSA Toulouse / LAAS)

Rappels probas !

- $P[A|B] = P[A \cap B]/P[B]$
- A indépendant de $B \Leftrightarrow P[A \cap B] = P[A] \times P[B]$
  $\Leftrightarrow P[A|B] = P[A]$
- Probabilités totales de B : $\mathrm{P[B]} = \bigcup P[A_i \cap B] = \sum P[A_i \cap B]$
  A_i est une partition des évènements possibles.

Le *One Time Pad* (6)

Propriété de sécurité du *One Time Pad*

Le *One Time Pad* (6)

Propriété de sécurité du *One Time Pad*

$$P[C = c] = \bigcup_{i \in \mathcal{M}} P[M = i \cap C = c]$$ tous les $i \in \mathcal{M}$ chiffrés en c

Le *One Time Pad* (6)

Propriété de sécurité du *One Time Pad*

$$P[C = c] = \bigcup_{i \in \mathcal{M}} P[M = i \cap C = c] \qquad \text{tous les } i \in \mathcal{M} \text{ chiffrés en } c$$

$$= \sum_{i \in \mathcal{M}} P[M = i \cap C = c]$$

Le *One Time Pad* (6)

Propriété de sécurité du *One Time Pad*

$$P[C = c] = \bigcup_{i \in \mathcal{M}} P[M = i \cap C = c] \qquad \text{tous les } i \in \mathcal{M} \text{ chiffrés en } c$$

$$= \sum_{i \in \mathcal{M}} P[M = i \cap C = c]$$

$$= \sum_{i \in \mathcal{M}} P[M = i \cap K = c \oplus i] \qquad \text{une seule clé donne } c \text{ pour } i$$

Le *One Time Pad* (6)

Propriété de sécurité du *One Time Pad*

$$P[C = c] = \bigcup_{i \in \mathcal{M}} P[M = i \cap C = c] \qquad \text{tous les } i \in \mathcal{M} \text{ chiffrés en } c$$

$$= \sum_{i \in \mathcal{M}} P[M = i \cap C = c]$$

$$= \sum_{i \in \mathcal{M}} P[M = i \cap K = c \oplus i] \qquad \text{une seule clé donne } c \text{ pour } i$$

$$= \sum_{i \in \mathcal{M}} P[M = i] \times 1/|\mathcal{K}| \qquad K \text{ est indépendant de } M$$

Le *One Time Pad* (6)

Propriété de sécurité du *One Time Pad*

$$P[C = c] = \bigcup_{i \in \mathcal{M}} P[M = i \cap C = c] \qquad \text{tous les } i \in \mathcal{M} \text{ chiffrés en } c$$

$$= \sum_{i \in \mathcal{M}} P[M = i \cap C = c]$$

$$= \sum_{i \in \mathcal{M}} P[M = i \cap K = c \oplus i] \qquad \text{une seule clé donne } c \text{ pour } i$$

$$= \sum_{i \in \mathcal{M}} P[M = i] \times 1/|\mathcal{K}| \qquad K \text{ est indépendant de } M$$

$$= 1/|\mathcal{K}| \times \sum_{i \in \mathcal{M}} P[M = i] = 1/|\mathcal{K}|$$

La sortie du *One Time Pad* a une distribution uniforme !

Sécurité parfaite

Selon Claude Shannon

Définition

$P[M = m|C = c] = P[M = m]$

- L'adversaire n'apprend rien du chiffré
- La cryptanalyse du chiffré est impossible

Sécurité parfaite du *One Time Pad* (1)

Démonstration

$P[M = m|C = c] = P[M = m \cap C = c]/P[C = c]$
$P[C = c] = 1/|\mathcal{K}|$

Sécurité parfaite du *One Time Pad* (1)

Démonstration

$P[M = m|C = c] = P[M = m \cap C = c]/P[C = c]$
$P[C = c] = 1/|\mathcal{K}|$

$P[M = m \cap C = c] = P[M = m \cap K = c \oplus m]$

Sécurité parfaite du *One Time Pad* (1)

Démonstration

$P[M = m|C = c] = P[M = m \cap C = c]/P[C = c]$
$P[C = c] = 1/|\mathcal{K}|$

$$P[M = m \cap C = c] = P[M = m \cap K = c \oplus m]$$
$$= P[M = m] \times 1/|\mathcal{K}|$$

Sécurité parfaite du *One Time Pad* (1)

Démonstration

$$P[M = m | C = c] = P[M = m \cap C = c] / P[C = c]$$
$$P[C = c] = 1/|\mathcal{K}|$$

$$\begin{aligned} P[M = m \cap C = c] &= P[M = m \cap K = c \oplus m] \\ &= P[M = m] \times 1/|\mathcal{K}| \end{aligned}$$

$$P[M = m | C = c] = \frac{P[M = m] \times 1/|\mathcal{K}|}{1/|\mathcal{K}|}$$

Sécurité parfaite du *One Time Pad* (1)

Démonstration

$$P[M = m | C = c] = P[M = m \cap C = c] / P[C = c]$$
$$P[C = c] = 1/|\mathcal{K}|$$

$$\begin{aligned} P[M = m \cap C = c] &= P[M = m \cap K = c \oplus m] \\ &= P[M = m] \times 1/|\mathcal{K}| \end{aligned}$$

$$\begin{aligned} P[M = m | C = c] &= \frac{P[M = m] \times 1/|\mathcal{K}|}{1/|\mathcal{K}|} \\ &= P[M = m] \end{aligned}$$

□

Sécurité parfaite du *One Time Pad* (1)

Démonstration

$P[M = m|C = c] = P[M = m \cap C = c]/P[C = c]$
$P[C = c] = 1/|\mathcal{K}|$

$$P[M = m \cap C = c] = P[M = m \cap K = c \oplus m]$$
$$= P[M = m] \times 1/|\mathcal{K}|$$

$$P[M = m|C = c] = \frac{P[M = m] \times 1/|\mathcal{K}|}{1/|\mathcal{K}|}$$
$$= P[M = m] \qquad \square$$

Le *One Time Pad* possède une sécurité parfaite

Limites OTP (1)

Maléabilité

Soit $m \in \mathcal{M}, c \in \mathcal{C}, k \in \mathcal{K}$
$c = E_k(m)$
Et après une attaque, réception de $c_2 \mid c_2 = c \oplus x$
$D_k(c_2) = c_2 \oplus k = m \oplus k \oplus x \oplus k = m \oplus x$
L'attaquant $\oplus$ directement le clair ! !

Réutilisation de la clé impossible

Soit $m_1, m_2 \in \mathcal{M}, c_1, c_2 \in \mathcal{C}, k \in \mathcal{K}$
$c_1 = E_k(m_1)$ et $c_2 = E_k(m_2)$
$c_1 \oplus c_2 = m_1 \oplus k \oplus m_2 \oplus k = m_1 \oplus m_2$
Ou exclusif des clairs ! !

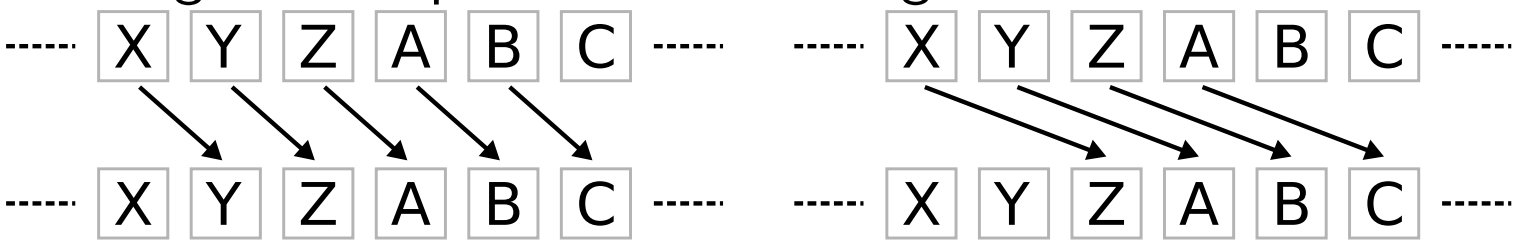
Limites OTP (2)

Réutilisation de la clé impossible 2

Soit $m_1, m_2 \in \mathcal{M}, c_1, c_2 \in \mathcal{C}, k \in \mathcal{K}$
$c_1 = E_k(m_1)$ et $c_2 = E_k(m_2)$
Si m_1 est connu par l'attaquant
$m_1 \oplus c_1 = m_1 \oplus m_1 \oplus k = k$
L'attaquant peut déchiffrer m_2

Chiffre de César (1)

- Décalage de chaque lettre du message clair d'une distance fixe



- Soient n la distance de décalage, x la lettre à chiffrer/déchiffrer

$$\begin{array}{ll} \text{Chiffrement} & E_e(x) = (x + e) \mod 26 \\ \text{Déchiffrement} & E_d(x) = (x - d) \mod 26 \end{array}$$

- Exemple

Message clair	`chiffre de cesar`
Distance 1	`dijggsf ef dftbs`
Distance 2	`ejkhhtg fg eguct`

- Utilisé par Jules César lors de la Guerre des Gaules avec $n = 3$

Chiffre de César (2)

- Analyse fréquentielle possible (fonction de la langue)
  - Certaines lettres sont plus employées que d'autres
  - La fréquence d'une lettre dans un message égale la fréquence de son image dans le message chiffré

$$f(x, M) = f(C_n(x), C_n(M))$$

  - Liste des lettres alphabétiques de la plus fréquente à la moins fréquente dans un texte français

`EAISTNRULODMPCVQGBFJHZXYKW`

  - Exemple

`atnqf zs fzywj hmnkkwj ij hjxfw xn kfhnqj f hfxxjw yjm !`
`voila un autre chiffre de cesar si facile a casser teh !`

Chiffre de César (2)

- Analyse fréquentielle possible (fonction de la langue)
  - Certaines lettres sont plus employées que d'autres
  - La fréquence d'une lettre dans un message égale la fréquence de son image dans le message chiffré

$$f(x, M) = f(C_n(x), C_n(M))$$

  - Liste des lettres alphabétiques de la plus fréquente à la moins fréquente dans un texte français

`EAISTNRULODMPCVQGBFJHZXYKW`

  - Exemple

```
atnqf zs fzywj hmnkkwj ij hjxfw xn kfhnqj f hfxxjw yjm !
voila un autre chiffre de cesar si facile a casser teh !
```

Chiffre de Vigenère (1)

- Blaise de Vigenère (1523 – 1596), diplomate francais
- Amélioration du chiffre de César ⇒ substitution polyalphabétique
  - Une lettre de l'alphabet peut être chiffrée de plusieurs manières différentes
  - La clé est représentée par une chaîne de caractères
  - Un caractère de la clef = une distance

    A=0 B=1 C=2 ...
  - Répétition de la clé, si nécessaire
- Exemple

Message clair	c	h	i	f	f	r	e	d	e	v	i	g	e	n	e	r	e
Clé	u	n	e	c	l	e	u	n	e	c	l	e	u	n	e	c	l
Décalage	20	13	4	2	11	4	20	13	4	2	11	4	20	13	4	2	11
Message chiffré	w	u	m	h	q	v	y	q	i	x	t	k	y	a	i	t	p

Chiffre de Vigenère (2)

Comment obtenir le message clair avec, seulement, le message chiffré ?

Message chiffré esmhqvgxipeexpgnpgjtjhcifpzkrippvg

- Trois motifs : vg ($\Delta = 27$), ip ($\Delta = 21$) et pg ($\Delta = 3$)
- La distance entre les répétitions d'un motif est multiple de 3

$\Rightarrow$ La clé a vraissemblablement une taille de 3

Analyse fréquentielle	e h g p x n j h f k p g s q x e p p t c p r p m v i e g g j i z i v
Message clair	chiffrementaveclechiffredevigenere
Clé	cleclecleclecleclecleclecleclecleclec
Message chiffré	esmhqvgxipeexpgnpgjtjhcifpzkrippvg

Plan du cours

Signatures [4] 1.6

Signature

Permet d'associer l'identité d'une entité à de l'information
Signer une information consiste a transformer un message $m \in \mathcal{M}$ et une information secrète que possède l'entité en un **tag**, $s \in \mathcal{S}$, appelé signature.

Services de sécurité

La signatures une primitive pilier pour les services de sécurité suivants

- Authentification des entités (*authentication of entities*)
- Non répudiation

Signatures [4] 1.6

- $\mathcal{M}$ est l'ensemble des message pouvant être signés
- $\mathcal{S}$ est l'ensemble des **tags** ou signatures

Transformation de signature de l'entité A

- $S_A : \mathcal{M} \to \mathcal{S}$
- Gardée secrète par A

Trans. de vérification des signatures de l'entité A

- $V_A : \mathcal{M} \times \mathcal{S} \to \{true, false\}$
- Connues par les entités devant vérifier les signatures créées par A
- V_A est **publique**, i.e. connue de l'adversaire

Rappel : principes de Kerckhoffs $\to$ usage de clés pour V_A et S_A.

Schéma de signature (1) [4] 1.6

Un schéma de signature **pour** A se compose de :

- De la transformation de signature de A : S_A
  S_A est caractérisée par la clé k_A secrète.
- De la transformation de vérification signatures créées par A : V_A
  V_A est caractérisée par la clé l_A publique.

Schéma de signature (2) [4] 1.6

Propriétés d'un schéma de signature

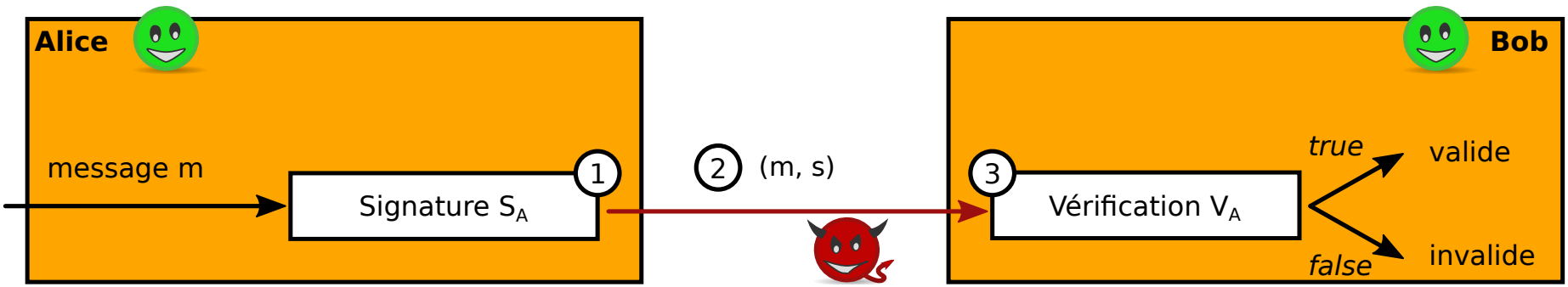
Avec les propriétés suivantes :

1. $s \in \mathcal{S}$ est une signature de A valide sur $m \in \mathcal{M}$ si et seulement si $V_A(m, s) = true$
2. Il est impossible pour une entité $B \neq A$ de calculer $\forall m \in \mathcal{M}$, un tag $s \in \mathcal{S}$ tel que $V_A(m, s) = true$

Remarque

- Pas de preuve formelle de l'existence de schémas satisfiant 1

Utilisation de la signature



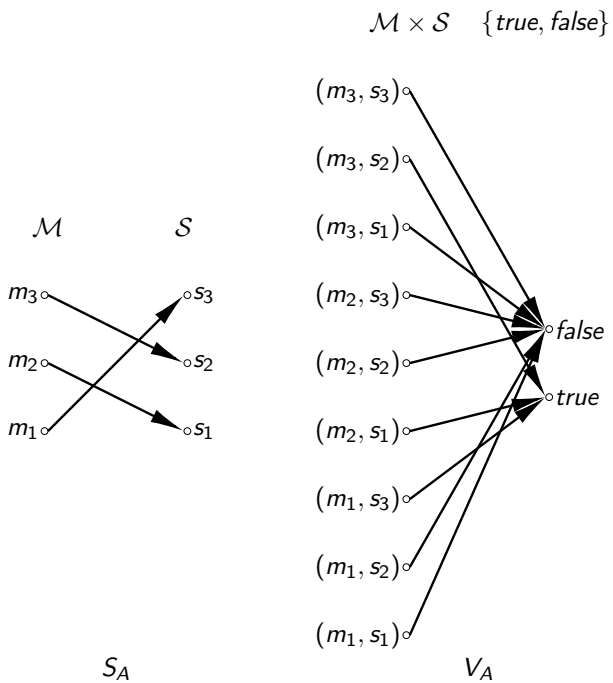
1. Alice signe m avec S_A secrète : $s = S_A(m)$
2. Alice publie sa (m, s) sur un canal non sécurisé à l'adversaire et à Bob
3. Bob vérifie la signature d'Alice avec V_A

Signatures

Exemple de schéma de signature pour l'entité A

- $\mathcal{M} = \{m_1, m_2, m_3\}$
- $\mathcal{S} = \{s_1, s_2, s_3\}$

Remarque : S_A n'est pas forcément une bijection



Plan du cours

Authentification (1)

Indépendant de la confidentialité

Définition informelle

L'authentification permet de façon générale de :

- S'assurer que des entités sont réellement qui elles prétendent être
- S'assurer l'information n'a pas été modifiée de façon non autorisée

Services de sécurité liés

- Authentification des entités
- Authentification de messages
- Intégrité / authenticité de messages
- Non répudiation

Authentification (2)

Scénarios utilisant des aspects de l'authentification

Alice (A) et Bob (B) sont deux entités communiquant sur un réseau :

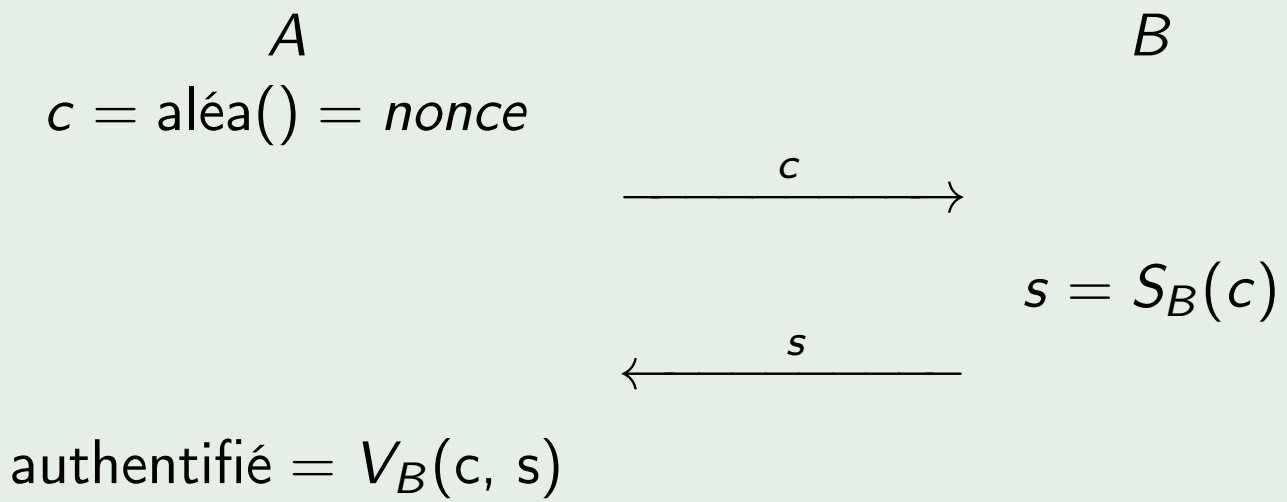
1. A et B sont actives en même temps avec un délai négligeable
   exemple : connexion TCP, datagrammes UDP, ...
2. A et B échangent des messages avec un délai non négligeable temps
   exemple : mails, SMS, ...

Authentification (3)

Scénario 1 : authentification des entités

- Vérification des identités de A et B "en temps réel"
- Éventuellement "dans les deux sens"

Protocole possible : A envoie à B un challenge auquel seulement B peut répondre correctement.



Authentification (4)

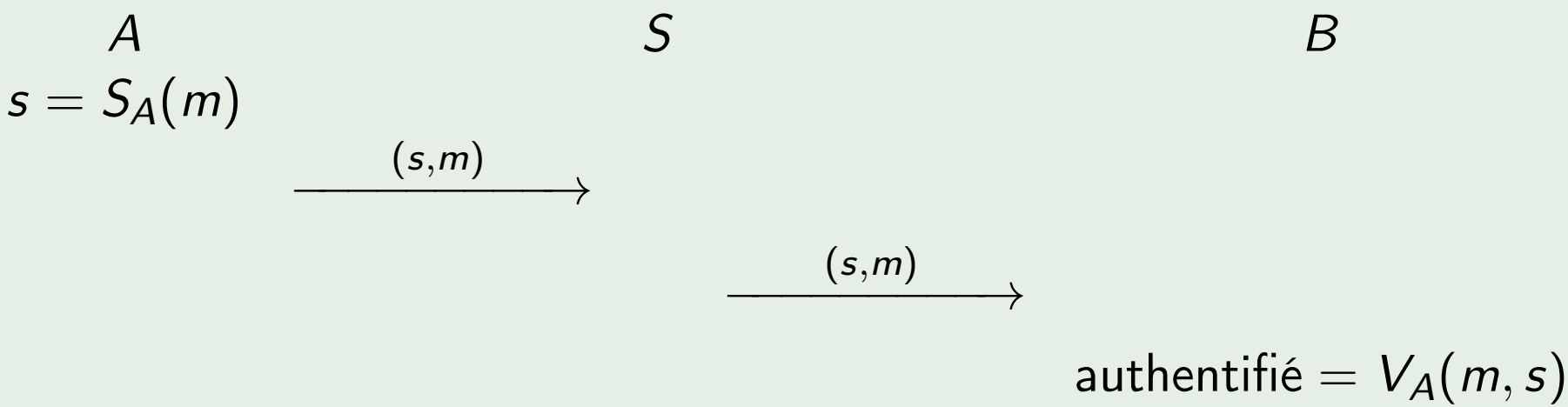
- HTTPS (TLS)
- SSH (SSH-USERAUTH, SSH-TRANS)
- IPSEC/IKE
- ...

Authentification (5)

Scénario 2 : authentification de message

- Vérification de l'origine des données en temps différé

Protocole possible : A envoie au serveur S un message avec sa signature $S_A(m)$. B Récupère le message depuis S et vérifie l'origine du message à l'aide de V_A.



Authentification (6)

- PGP
- SMIME
- ...

Plan du cours

Schéma de chiffrement à clé publique

Définition

Un schéma de chiffrement aux transformations $\{E_e \mid e \in \mathcal{K}\}$ et $\{D_d \mid d \in \mathcal{K}\}$ est dit à clé publiqué si pour toute paire (E_e, D_d) et pour tout $c \in \mathcal{C}$ il est "impossible" de trouver le $m \in \mathcal{M}$ tel que $E_e(m) = c$.
Autrement dit il est "impossible" d'inverser la transformation de chiffrement.
E_e est une fonction à sens unique avec trappe et d est la trappe.

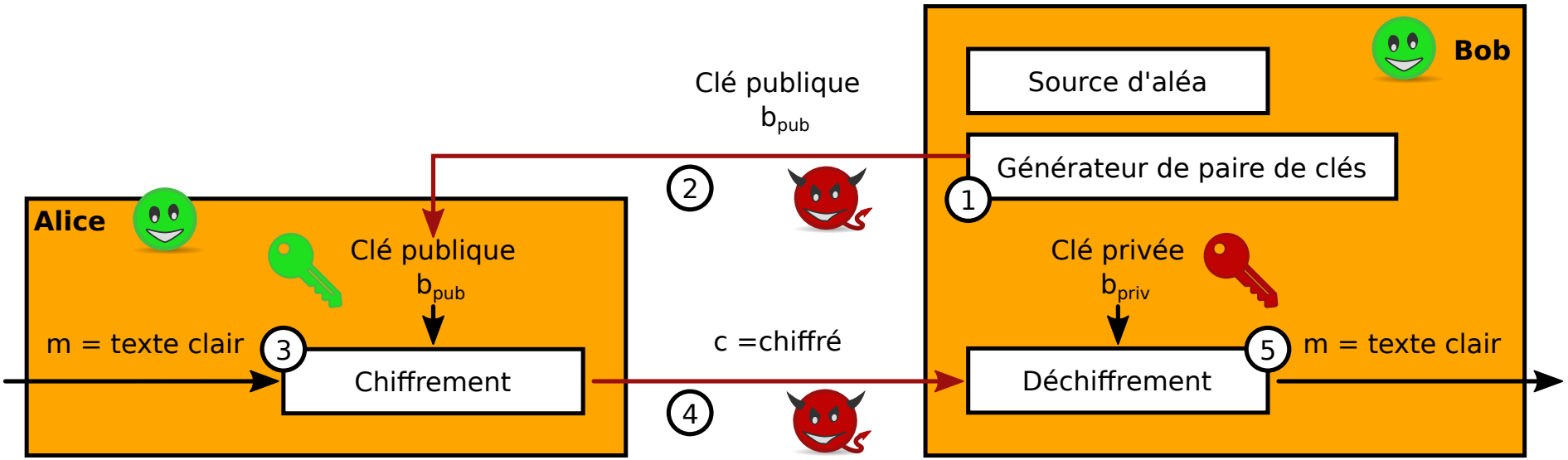
Notation clé publique et clé privée

Par convention, on apelle e la clé publique et d la clé privée.
$e = k_{\text{pub}} \quad d = k_{\text{priv}}$

Exemple

Le chiffrement *RSA*

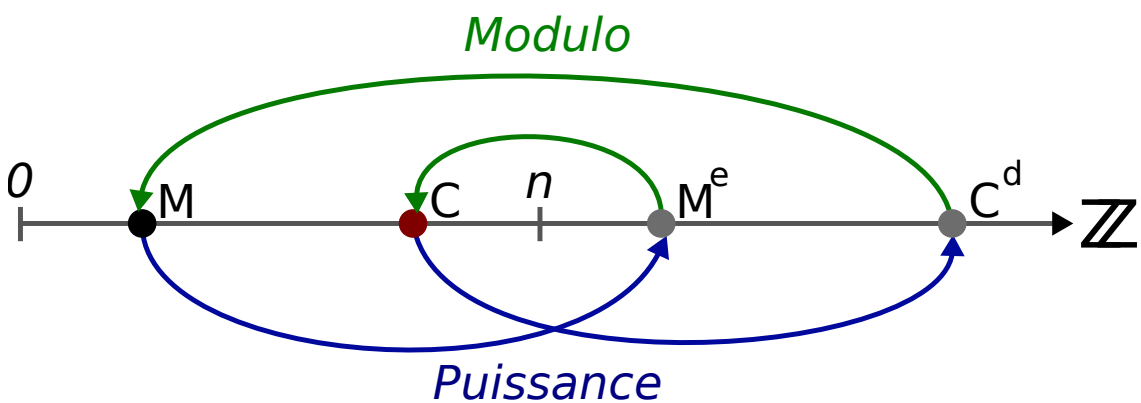
Confidentialité et schéma de chiffrement à clé publique



1. Génération des clés (b_{pub}, b_{priv}) avec b_{priv} non déductible de b_{pub}
2. Distribution de b_{pub} à Alice et à l'adversaire
3. Alice chiffre le clair m avec la transformation $E_{b_{pub}}(m) = c$
4. Alice transmet à Bob le chiffré c sur un canal non sécurisé
5. Bob déchiffre c pour retrouver $m = D_{b_{priv}}(c)$

RSA – Rivest, Shamir, Adleman (1)

- Création des clés
  - Choisir p et q deux nombres premiers distincts
  - $\Rightarrow$ Calculer le *module de chiffrement* n, $n = p \cdot q$
  - $\Rightarrow$ Calculer l'*indicatrice d'Euleur* de n, $\phi(n) = (p-1) \cdot (q-1)$
  - Choisir l'*exposant de chiffrement* e, un entier premier avec $\phi(n)$,
  - $\Rightarrow$ Calculer l'*exposant de déchiffrement* d, $e \cdot d \equiv 1 \pmod{\phi(n)}$
    Algorithme d'Euclide étendu
- Chiffrement : $c = m^e \pmod{n}$, avec $m < n$
- Déchiffrement : $m = c^d \pmod{n}$
- Décomposition de n en produit de facteurs premiers $\rightarrow$ p et q $\mathcal{O}(e^n)$



RSA – Rivest, Shamir, Adleman (2)

- Exemple
  - Création des clés
    - $p = 5,\ q = 11$
    - $\Rightarrow n = p \cdot q = 5 \cdot 11 = 55$
    - $\Rightarrow \phi(n) = (p-1) \cdot (q-1) = (5-1) \cdot (11-1) = 4 \cdot 10 = 40$
    - $e = 3$
    - $\Rightarrow d = 27 \quad$ Vérification : $d \cdot e \stackrel{?}{\equiv} 1 \pmod{\phi(n)}$
      $d \cdot e = 3 \cdot 27 = 81 = 2 \cdot 40 + 1 \equiv 1 \mod 40$
    - $\Rightarrow k_{\text{pub}} = \{n, e\} = \{55, 3\} \quad k_{\text{priv}} = \{n, d, p, q\} = \{55, 27, 5, 11\}$
  - Chiffrement de $m = 19$
    - $c = m^c \mod n = 19^3 \mod 55 = 6859 \mod 55 = 39$
  - Déchiffrement de $c = 39$
    - $m = c^d \mod n = 39^{27} \mod 55 = 39$
    - $39^{27} = 909377887614652551975371341130628025063947 9$

RSA – Rivest, Shamir, Adleman (3)

Définition

$k_{\text{pub}} = (e, n) \in \mathbb{N}^2$
$k_{\text{priv}} = (d, n, p, q) \in \mathbb{N}^4$
$\mathcal{K} = \{(k_{\text{pub}}, k_{\text{priv}}) \mid (k_{\text{pub}}, k_{\text{priv}}) \text{ telles que RSA cohérent}\}$

Transformation de chiffrement $E_{k_{\text{PUB}}}$

$E_{k_{\text{pub}}} : (\mathbb{Z}/n\mathbb{Z})^* \to (\mathbb{Z}/n\mathbb{Z})^*$
$E_{k_{\text{pub}}} : m \mapsto m^e \pmod{n}$

Transformation de déchiffrement $D_{k_{\text{PRIV}}}$

$D_{k_{\text{priv}}} : (\mathbb{Z}/n\mathbb{Z})^* \to (\mathbb{Z}/n\mathbb{Z})^*$
$D_{k_{\text{priv}}} : c \mapsto c^d \pmod{n}$

RSA – Rivest, Shamir, Adleman (4)

On veut montrer que : $c^d \equiv m \pmod{n}$

Démonstration de cohérence (1)

$n = pq$ p, q deux nombres premiers
Soit $m \in (\mathbb{Z}/n\mathbb{Z})^* \Rightarrow p$ et q ne divisent pas m

RSA – Rivest, Shamir, Adleman (4)

On veut montrer que : $c^d \equiv m \pmod{n}$

Démonstration de cohérence (1)

$n = pq$ p, q deux nombres premiers
Soit $m \in (\mathbb{Z}/n\mathbb{Z})^* \Rightarrow p$ et q ne divisent pas m

---

D'après le petit théorème de Fermat :
$m^{p-1} \equiv 1 \pmod{p}$ et $m^{q-1} \equiv 1 \pmod{q}$

RSA – Rivest, Shamir, Adleman (4)

On veut montrer que : $c^d \equiv m \pmod{n}$

Démonstration de cohérence (1)

$n = pq$ p, q deux nombres premiers
Soit $m \in (\mathbb{Z}/n\mathbb{Z})^* \Rightarrow p$ et q ne divisent pas m

---

D'après le petit théorème de Fermat :
$m^{p-1} \equiv 1 \pmod{p}$ et $m^{q-1} \equiv 1 \pmod{q}$

$c^d \equiv (m^e)^d \equiv m^{ed} \pmod{n}$ sur $(\mathbb{Z}/n\mathbb{Z})^*$

RSA – Rivest, Shamir, Adleman (4)

On veut montrer que : $c^d \equiv m \pmod{n}$

Démonstration de cohérence (1)

$n = pq$ p, q deux nombres premiers
Soit $m \in (\mathbb{Z}/n\mathbb{Z})^* \Rightarrow p$ et q ne divisent pas m

---

D'après le petit théorème de Fermat :
$m^{p-1} \equiv 1 \pmod{p}$ et $m^{q-1} \equiv 1 \pmod{q}$

$c^d \equiv (m^e)^d \equiv m^{ed} \pmod{n}$ sur$(\mathbb{Z}/n\mathbb{Z})^*$

$ed \equiv 1 \pmod{\phi(n) = (p-1)(q-1)}$
$ed = 1 + k(p-1)(q-1)$

RSA – Rivest, Shamir, Adleman (5)

On veut montrer que : $c^d \equiv m \pmod{n}$

Démonstration de cohérence (2)

$$m^{p-1} \equiv 1 \pmod{p}$$

$$(m^{p-1})^{k(q-1)} \equiv 1 \pmod{p} \quad \text{puissance } k(q-1)$$

$$m^{1+k(p-1)(q-1)} \equiv m \pmod{p} \quad \text{multiplication par } m$$

$$m^{ed} \equiv m \pmod{p} \quad \Rightarrow \quad (1)\ p \text{ divise } m^{ed} - m$$

RSA – Rivest, Shamir, Adleman (5)

On veut montrer que : $c^d \equiv m \pmod{n}$

Démonstration de cohérence (2)

$$m^{p-1} \equiv 1 \pmod{p}$$

$$(m^{p-1})^{k(q-1)} \equiv 1 \pmod{p} \quad \text{puissance } k(q-1)$$

$$m^{1+k(p-1)(q-1)} \equiv m \pmod{p} \quad \text{multiplication par } m$$

$$m^{ed} \equiv m \pmod{p} \quad \Rightarrow \quad (1)\ p \text{ divise } m^{ed} - m$$

De la même manière on montre que :

$$m^{ed} \equiv m \pmod{q} \quad \Rightarrow \quad (2)\ q \text{ divise } m^{ed} - m$$

RSA – Rivest, Shamir, Adleman (5)

On veut montrer que : $c^d \equiv m \pmod{n}$

Démonstration de cohérence (2)

$$m^{p-1} \equiv 1 \pmod{p}$$

$$(m^{p-1})^{k(q-1)} \equiv 1 \pmod{p} \quad \text{puissance } k(q-1)$$

$$m^{1+k(p-1)(q-1)} \equiv m \pmod{p} \quad \text{multiplication par } m$$

$$m^{ed} \equiv m \pmod{p} \quad \Rightarrow \quad (1)\ p \text{ divise } m^{ed} - m$$

De la même manière on montre que :

$$m^{ed} \equiv m \pmod{q} \quad \Rightarrow \quad (2)\ q \text{ divise } m^{ed} - m$$

---

p et q sont premiers et d'après (1) et (2) :

$m^{ed} - m$ est aussi divisé par $p \times q \Rightarrow m^{ed} \equiv m \pmod{n}$ □

RSA – Rivest, Shamir, Adleman (6)

Maléabilité

Alice envoie à Bob c_1 sur un canal non sécurisé tel que :

$$m_1^e \equiv c_1 \pmod{n}$$

Eve en homme dans le milieu transforme c_1 en c tel que :

$$c = c_1 c_2 \mid c_2 \equiv m_2^e \pmod{n} \quad m_2 \text{ inconnu d'Eve}$$

Bob reçoit c et déchiffre m' tel que :

$$m' \equiv c^d \equiv (c_1 c_2)^d \equiv c_1^d c_2^d \equiv m_1^{ed} m_2^{ed} \equiv m_1 m_2 \pmod{n}$$

Par associativité de la multiplication et RSA

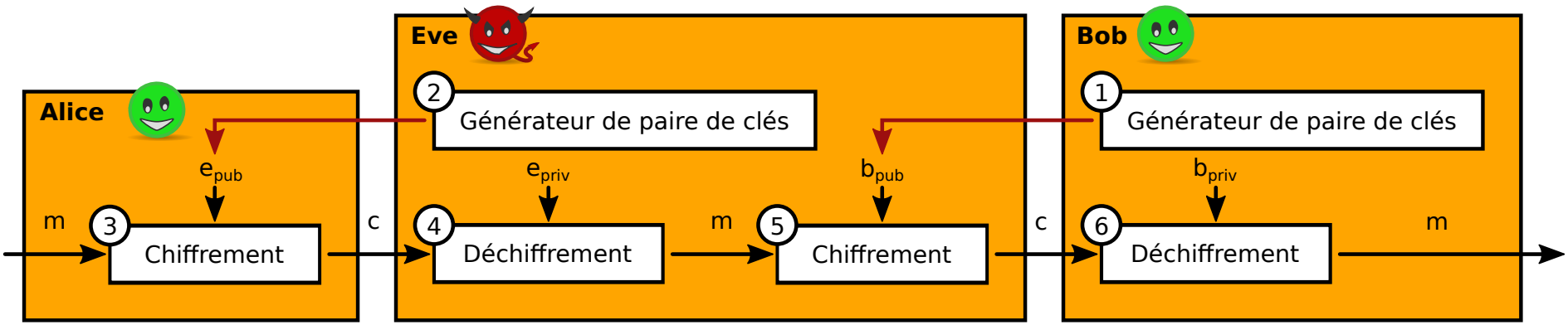
RSA – Rivest, Shamir, Adleman (7)

```
>>> p = 5
>>> q = 11
>>> n = p * q
>>> e = 3
>>> d = 27
>>> m1 = 2
>>> m2 = 4
>>> c1 = m1 ** e % n
>>> c2 = m2 ** e % n
>>> c = c1 * c2 % n
>>> c ** d % n
8
>>> m1 * m2
8
```

Avantages des chiffres à clé publique

- Pas de confiance mutuelle entre émetteur et récepteur
- Gestion de clé "facile"
  - Répertoire public de clés publiques ou distribution entre pairs
  - La clé privée ne doit "jamais" être transmise
- Possibilité d'utilisations nouvelles : distribution de clés symétriques, signatures, certificats, etc.

Attaque par homme dans le milieu



1. Génération des clés $(b_{\text{pub}}, b_{\text{priv}})$
   Envoi de b_{pub} intercepté par Eve
2. Génération des clés $(e_{\text{pub}}, e_{\text{priv}})$
   Envoi de e_{pub} intercepté à Alice
3. Alice chiffre le clair m avec e_{pub}
   et envoie c à Eve
4. Eve déchiffre m avec e_{priv}
5. Eve chiffre le clair m avec b_{pub}
   et envoie c' à Bob
6. Bob déchiffre c' avec b_{priv}

Signatures (1)

Il est possible de construire une classe de schéma de signatures à l'aide de schéma de chiffrement à clé publique particuliers.

Schéma réversible

Soit E_e une transformation de chiffrement à clé publique sur $\mathcal{M}$ et $\mathcal{C}$.
Suposons que $\mathcal{C} = \mathcal{M}$. Soit D_d est la transformation de déchiffrement associée à E_e.
Comme E_e et D_d sont toutes deux des permutations, on a :

$$D_d(E_e(m)) = E_e(D_d(m)) = m, \ \forall m \in \mathcal{M}$$

Signatures (2)

Schéma de signature sur chiffrement réversible

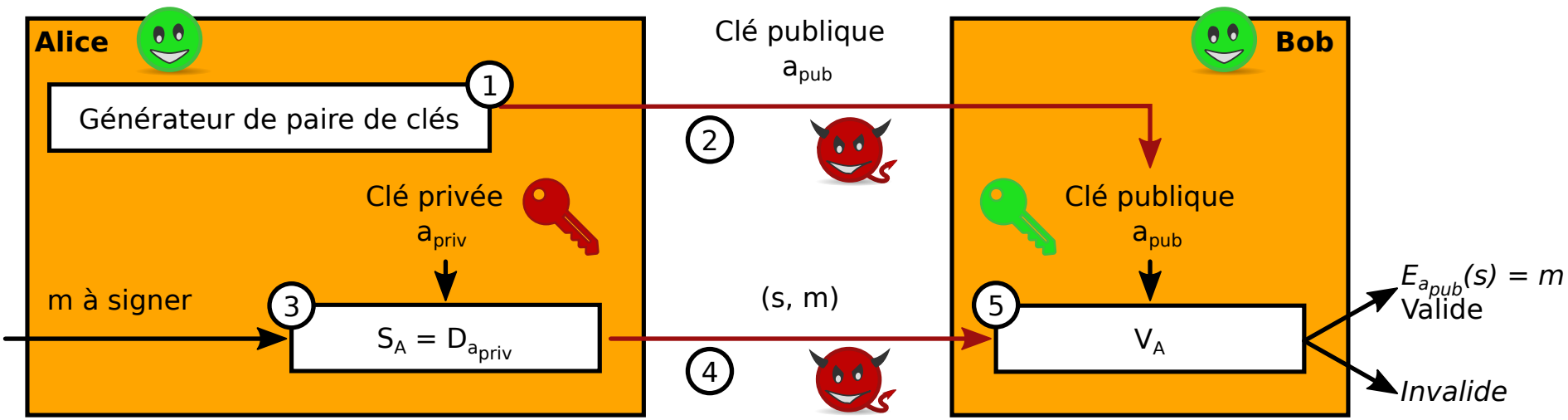
- $\mathcal{M}$ est l'espace des messages
- $\mathcal{C} = \mathcal{M}$ est l'espace des signatures $\mathcal{S}$
- Soit $(a_{\text{pub}}, a_{\text{priv}})$ la paire de clés de chiffrement de l'entité A.
- Définir la fonction de signature de l'entité A telle que $S_A = D_{a_{\text{priv}}}$:

$$\text{Signature de } m \in M \ : \ s = D_{a_{\text{priv}}}(m)$$

- Définir la fonction de vérification de signature de l'entité A telle que :

$$V_A(m, s) = \begin{cases} \mathit{valide}, & \text{si } E_{a_{\text{pub}}}(s) = m, \\ \mathit{invalide}, & \text{dans les autres cas.} \end{cases}$$

Signatures (3)



1. Génération des clés (a_{pub}, a_{priv})
2. Distribution de a_{pub} à Bob et à l'adversaire
3. Alice signe le clair m avec la transformation $S_a = D_{a_{priv}}(m) = s$
4. Alice transmet à Bob la signature (s, m) sur un canal non sécurisé
5. Bob Vérifie (s, m) en déchiffrant $m' = E_{a_{pub}}(s)$ et en comparant m' à m

Sécurité des signatures à chiffrement réversibles (1)

Maléabilité de RSA

Le produit de deux signatures sera une signature valide du produit des clairs !

$$s_1 \times s_2 = S_A(m_1 \times m_2)$$

$$V_A(s_1 \times s_2) = V_A(S_A(m_1 \times m_2))$$

Forge de signatures

V_A est publique, par conséquent E_e aussi.
L'adversaire peut choisir un $s_{\text{forgé}} \in \mathcal{M}$ et calculer $m_{\text{forgé}} = E_e(s_{\text{forgé}})$
L'adversaire peut donc forger des signatures pour A car :

$$V_A(m_{\text{forgé}}, s_{\text{forgé}}) = \text{valide}$$

Sécurité des signatures à chiffrement réversibles (2)

Forge : signature avec récupération de message

Choisir un ensemble des message signables $\mathcal{M}' \subset \mathcal{M}$ avec $|\mathcal{M}'| << |\mathcal{M}|$
Définir vérification telle que :

$$V_A(s) = \begin{cases} \textit{valide}, & \text{si } E_{a_{\text{pub}}}(s) \in \mathcal{M}', \\ \textit{invalide}, & \text{dans les autres cas.} \end{cases}$$

Pour choisir $s_{\text{forgé}}$ valide, tel que $E_e(s_{\text{forgé}}) \in \mathcal{M}'$, l'adversaire doit maintenant inverser E_e, ce qui est "impossible"

Remarque : si on transmet $m' \in \mathcal{M}'$ avec s, on peut aussi vérifier l'égalité en plus de l'inclusion.

Sécurité des signatures à chiffrement réversibles (3)

Intégrité de la signature

on peut utiliser des fonctions à sens unique "impossible" à inverser telles que les fonctions de hashage $\mathcal{H}$ pour calculer $m' \in \mathcal{M}'$ depuis des m de taille arbitraire. On signe et vérifie $\mathcal{H}(m) = m' \in \mathcal{M}'$

Conséquence sur la maléabilité de RSA :

$$V_A(s_1 s_2) = V_A(D_{k_{\text{priv}}}(\mathcal{H}(m_1)) \times D_{k_{\text{priv}}}(\mathcal{H}(m_2)))$$

Sécurité des signatures à chiffrement réversibles (3)

Intégrité de la signature

on peut utiliser des fonctions à sens unique "impossible" à inverser telles que les fonctions de hashage $\mathcal{H}$ pour calculer $m' \in \mathcal{M}'$ depuis des m de taille arbitraire. On signe et vérifie $\mathcal{H}(m) = m' \in \mathcal{M}'$

Conséquence sur la maléabilité de RSA :

$$\begin{aligned} V_A(s_1 s_2) &= V_A(D_{k_{\text{priv}}}(\mathcal{H}(m_1)) \times D_{k_{\text{priv}}}(\mathcal{H}(m_2))) \\ &= V_A(\mathcal{H}(m_1)^d) \times \mathcal{H}(m_2)^d) \end{aligned}$$

Sécurité des signatures à chiffrement réversibles (3)

Intégrité de la signature

on peut utiliser des fonctions à sens unique "impossible" à inverser telles que les fonctions de hashage $\mathcal{H}$ pour calculer $m' \in \mathcal{M}'$ depuis des m de taille arbitraire. On signe et vérifie $\mathcal{H}(m) = m' \in \mathcal{M}'$

Conséquence sur la maléabilité de RSA :

$$\begin{aligned} V_A(s_1 s_2) &= V_A(D_{k_{\text{priv}}}(\mathcal{H}(m_1)) \times D_{k_{\text{priv}}}(\mathcal{H}(m_2))) \\ &= V_A(\mathcal{H}(m_1)^d) \times \mathcal{H}(m_2)^d) \\ &= E_{k_{\text{pub}}}(\mathcal{H}(m_1)^d) \times \mathcal{H}(m_2)^d) \end{aligned}$$

Sécurité des signatures à chiffrement réversibles (3)

Intégrité de la signature

on peut utiliser des fonctions à sens unique "impossible" à inverser telles que les fonctions de hashage $\mathcal{H}$ pour calculer $m' \in \mathcal{M}'$ depuis des m de taille arbitraire. On signe et vérifie $\mathcal{H}(m) = m' \in \mathcal{M}'$

Conséquence sur la maléabilité de RSA :

$$\begin{aligned} V_A(s_1 s_2) &= V_A(D_{k_{\text{priv}}}(\mathcal{H}(m_1)) \times D_{k_{\text{priv}}}(\mathcal{H}(m_2))) \\ &= V_A(\mathcal{H}(m_1)^d) \times \mathcal{H}(m_2)^d) \\ &= E_{k_{\text{pub}}}(\mathcal{H}(m_1)^d) \times \mathcal{H}(m_2)^d) \\ &= (\mathcal{H}(m_1)^d \times \mathcal{H}(m_2)^d)^e \end{aligned}$$

Sécurité des signatures à chiffrement réversibles (3)

Intégrité de la signature

on peut utiliser des fonctions à sens unique "impossible" à inverser telles que les fonctions de hashage $\mathcal{H}$ pour calculer $m' \in \mathcal{M}'$ depuis des m de taille arbitraire. On signe et vérifie $\mathcal{H}(m) = m' \in \mathcal{M}'$

Conséquence sur la maléabilité de RSA :

$$\begin{aligned} V_A(s_1 s_2) &= V_A(D_{k_{\text{priv}}}(\mathcal{H}(m_1)) \times D_{k_{\text{priv}}}(\mathcal{H}(m_2))) \\ &= V_A(\mathcal{H}(m_1)^d) \times \mathcal{H}(m_2)^d) \\ &= E_{k_{\text{pub}}}(\mathcal{H}(m_1)^d) \times \mathcal{H}(m_2)^d) \\ &= (\mathcal{H}(m_1)^d \times \mathcal{H}(m_2)^d)^e \\ &= \mathcal{H}(m_1)^{de} \times \mathcal{H}(m_2)^{de} \end{aligned}$$

Sécurité des signatures à chiffrement réversibles (3)

Intégrité de la signature

on peut utiliser des fonctions à sens unique "impossible" à inverser telles que les fonctions de hashage $\mathcal{H}$ pour calculer $m' \in \mathcal{M}'$ depuis des m de taille arbitraire. On signe et vérifie $\mathcal{H}(m) = m' \in \mathcal{M}'$

Conséquence sur la maléabilité de RSA :

$$\begin{aligned} V_A(s_1 s_2) &= V_A(D_{k_{\text{priv}}}(\mathcal{H}(m_1)) \times D_{k_{\text{priv}}}(\mathcal{H}(m_2))) \\ &= V_A(\mathcal{H}(m_1)^d) \times \mathcal{H}(m_2)^d) \\ &= E_{k_{\text{pub}}}(\mathcal{H}(m_1)^d) \times \mathcal{H}(m_2)^d) \\ &= (\mathcal{H}(m_1)^d \times \mathcal{H}(m_2)^d)^e \\ &= \mathcal{H}(m_1)^{de} \times \mathcal{H}(m_2)^{de} \\ &= \mathcal{H}(m_1) \times \mathcal{H}(m_2) \neq \mathcal{H}(m_1 m_2) \end{aligned}$$

Sécurité des signatures à chiffrement réversibles (3)

Intégrité de la signature

on peut utiliser des fonctions à sens unique "impossible" à inverser telles que les fonctions de hashage $\mathcal{H}$ pour calculer $m' \in \mathcal{M}'$ depuis des m de taille arbitraire. On signe et vérifie $\mathcal{H}(m) = m' \in \mathcal{M}'$

Conséquence sur la maléabilité de RSA :

$$\begin{aligned} V_A(s_1 s_2) &= V_A(D_{k_{\text{priv}}}(\mathcal{H}(m_1)) \times D_{k_{\text{priv}}}(\mathcal{H}(m_2))) \\ &= V_A(\mathcal{H}(m_1)^d) \times \mathcal{H}(m_2)^d) \\ &= E_{k_{\text{pub}}}(\mathcal{H}(m_1)^d) \times \mathcal{H}(m_2)^d) \\ &= (\mathcal{H}(m_1)^d \times \mathcal{H}(m_2)^d)^e \\ &= \mathcal{H}(m_1)^{de} \times \mathcal{H}(m_2)^{de} \\ &= \mathcal{H}(m_1) \times \mathcal{H}(m_2) \neq \mathcal{H}(m_1 m_2) \\ &= \text{invalide} \end{aligned}$$

Plan du cours

Fonction de hashage

Définition

Une fonction de hashage est une fonction qui associe efficacement des chaînes binaires de taille arbitraire à des chaînes de taille t, fixée.

Propriété : uniformité de la sortie

Soit la fonction de hashage $\mathcal{H}$ et n la taille de ses chaînes de sortie. Soit m une chaîne de taille arbitraire tirée au hasard et $c = \mathcal{H}(m)$:
$P[C = c|M = m] = 2^{-n}$

Propriété : effet avalanche

Soit la fonction de hashage $\mathcal{H}$ de taille de sortie n. Soit $c = \mathcal{H}(m)$. Soit m' tel que $\text{Hamming}(m, m') = 1$. Alors en moyenne :

$$\text{Hamming}(\mathcal{H}(m), \mathcal{H}(m')) = n/2$$

Propriétés

Résistance aux préimages

Connaissant une image c par $\mathcal{H}$ il est "impossible" de calculer m tel que $\mathcal{H}(m) = c$

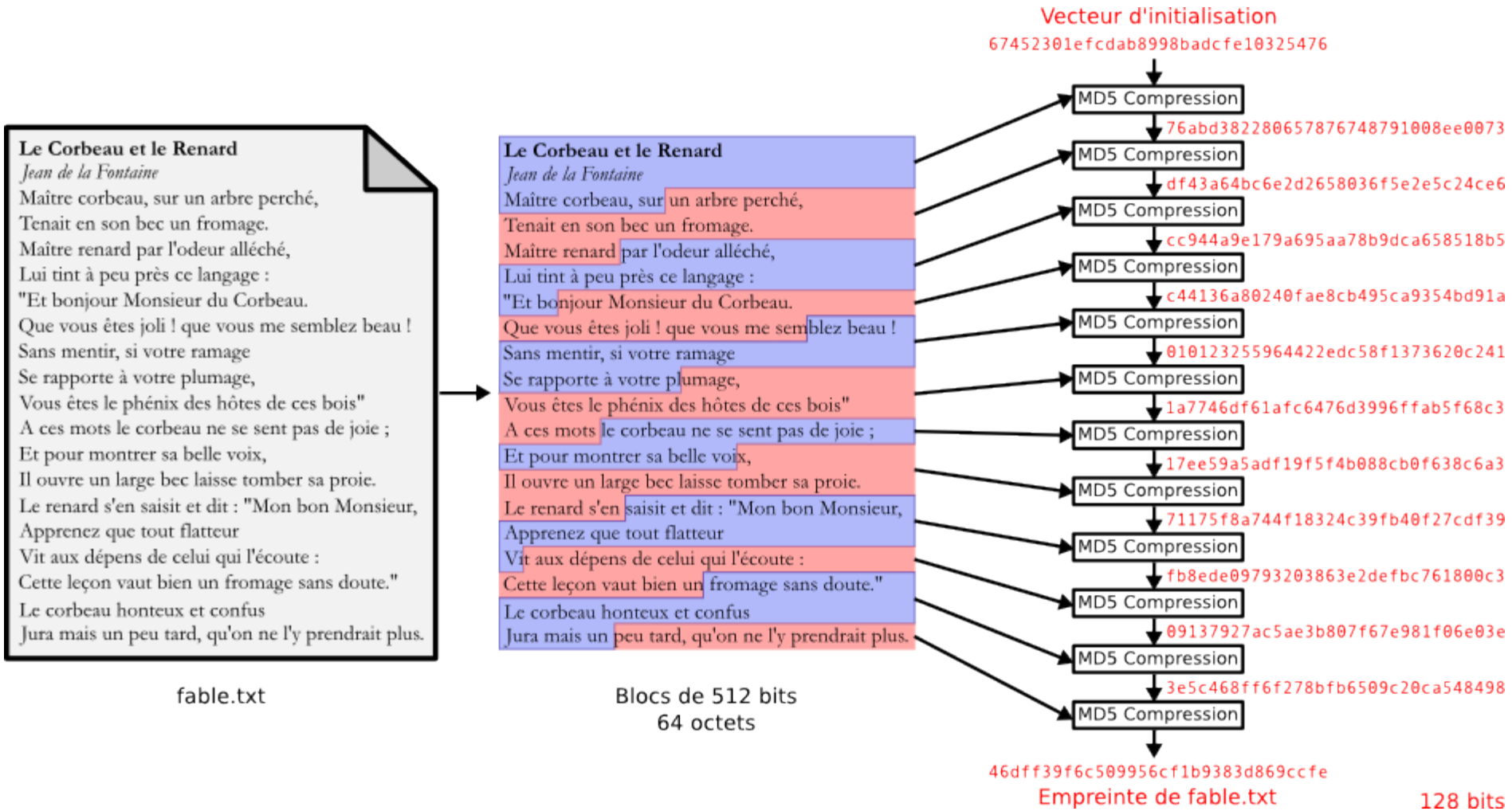
Résistance aux secondes préimages

Connaissant une image (m, c) tels que $\mathcal{H}(m) = c$ est "impossible" de calculer $m' \neq m$ que $\mathcal{H}(m') = c$

Résistance aux collisions

Il est impossible de calculer m et m' tel que $\mathcal{H}(m) = \mathcal{H}(m')$.
Paradoxe des anniversaires :
$P[\text{trouver } m \text{ et } m' \mid \mathcal{H}(m) = \mathcal{H}(m') \text{ après } 2^{n/2} \text{ essais}] \geq 1/2$

MD5


Le Corbeau et le Renard
Jean de la Fontaine
Maître corbeau, sur un arbre perché,
Tenait en son bec un fromage.
Maître renard par l'odeur alléché,
Lui tint à peu près ce langage :
"Et bonjour Monsieur du Corbeau.
Que vous êtes joli ! que vous me semblez beau !
Sans mentir, si votre ramage
Se rapporte à votre plumage,
Vous êtes le phénix des hôtes de ces bois"
A ces mots le corbeau ne se sent pas de joie ;
Et pour montrer sa belle voix,
Il ouvre un large bec laisse tomber sa proie.
Le renard s'en saisit et dit : "Mon bon Monsieur,
Apprenez que tout flatteur
Vit aux dépens de celui qui l'écoute :
Cette leçon vaut bien un fromage sans doute."
Le corbeau honteux et confus
Jura mais un peu tard, qu'on ne l'y prendrait plus.
fable.txt
Blocs de 512 bits
64 octets
Vecteur d'initialisation
67452301efcdab8998badcfe10325476
MD5 Compression
76abd38228065787674879 1008ee0073
df43a64bc6e2d2658036f5e2e5c24ce6
cc944a9e179a695aa78b9dca658518b5
c44136a80240fae8cb495ca9354bd91a
010123255964422edc58f1373620c241
1a7746df61afc6476d3996ffab5f68c3
17ee59a5adf19f5f4b088cb0f638c6a3
71175f8a744f18324c39fb40f27cdf39
fb8ede09793203863e2defbc761800c3
09137927ac5ae3b807f67e981f06e03e
3e5c468ff6f278bfb6509c20ca548498
46dff39f6c509956cf1b9383d869ccfe
Empreinte de fable.txt
128 bits

Intégrité

H-based MAC

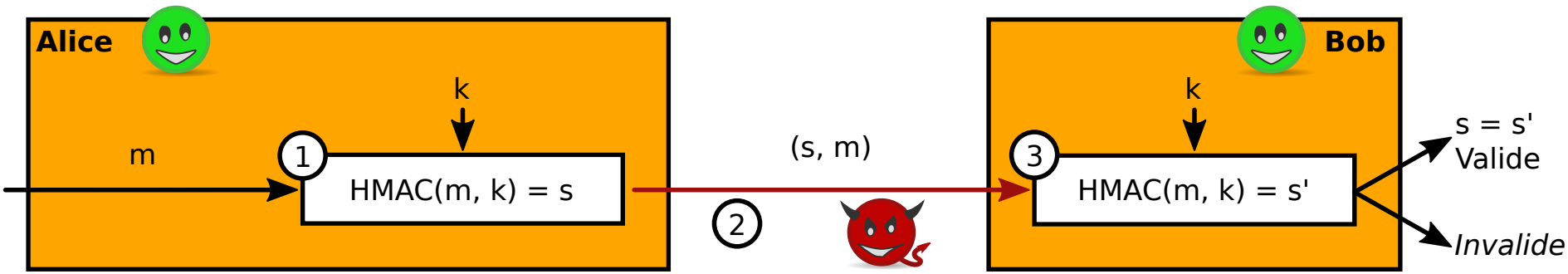
$$s = \mathcal{H}(k||m) \stackrel{?}{=} s' = \mathcal{H}(k||m)$$

- Vulnérabilités au attaques de *length extension* : *SHA-1, MD5*
- $\mathcal{H}(k||m||\text{bad}) = \mathcal{H}(\mathcal{H}(k||m)||\text{bad})$
- Faiblesse

Variante (HMAC)

$$s = \mathcal{H}(k||\mathcal{H}(k||m)) \stackrel{?}{=} s' = \mathcal{H}(k||\mathcal{H}(k||m))$$

Intégrité



1. Alice calcule le tag s sur le clair m avec k et envoie (s, m) à Bob et à l'adversaire
2. Bob calcule le tag s' sur le clair m avec k
4. Bob calcule si $s = s'$ et en déduit la validité

Plan du cours

Plan du cours