Crypto $\sim$
Algorithmes de chiffrement par bloc

Pierre Karpman

Visite ENS de Rennes/KL @ Inria/IRISA

2013–09–04

Contexte

- “La crypto, c’est un truc qui sert à établir des communications (sûres) en la présence d’adversaires.”
- “La crypto, c’est un truc qui sert uniquement parce qu’il y a des gens malhonnêtes et belliqueux.”

Contexte

- “La crypto, c’est un truc qui sert à établir des communications (sûres) en la présence d’adversaires.”
- “La crypto, c’est un truc qui sert uniquement parce qu’il y a des gens malhonnêtes et belliqueux.”

Exemple

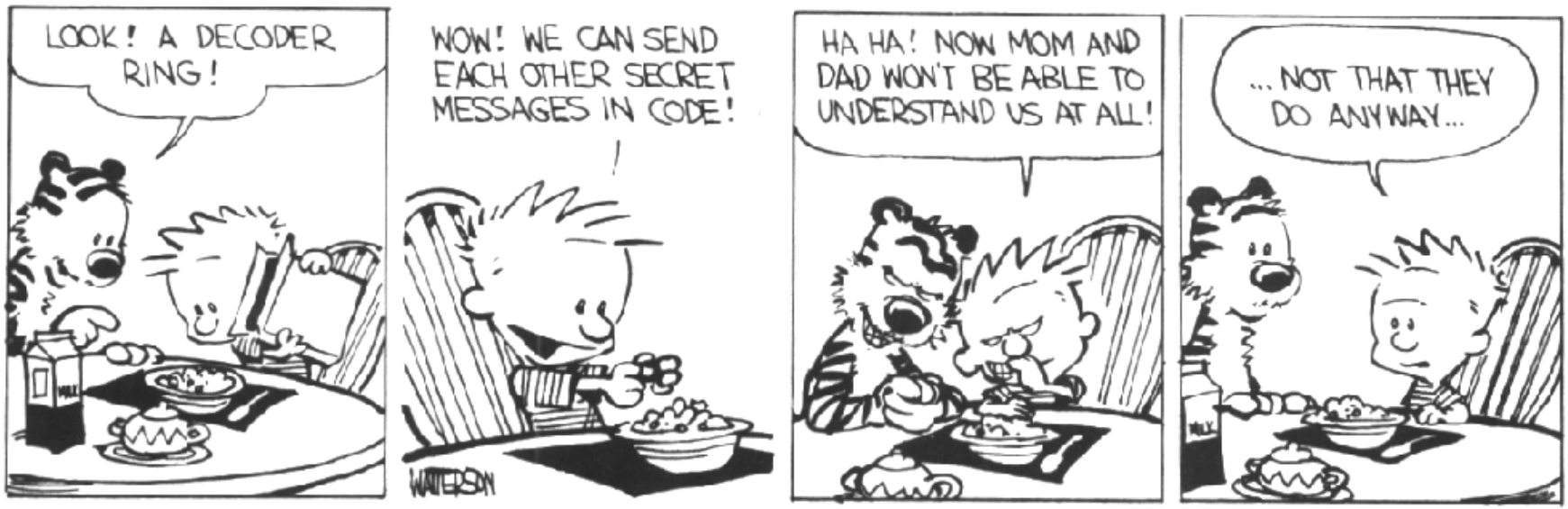

FIGURE : Utilisation de cryptographie pour se protéger contre des adversaires non étatiques.

Grands domaines en crypto

- Primitives à clef publique.
- Primitives à clef secrète.
- Protocoles, modes d'opération, études théoriques.
- Attaques physiques.
- Modèles et théories exotiques.

Zoom sur la crypto symétrique

FIGURE : Voyons ça de plus près !

Primitives symétriques

- Chiffres par bloc (par ex. AES, PRESENT).
- Chiffres par flot (par ex. RC4, SNOW 3G).
- Fonctions de hachage (par ex. SHA{0,1,2,3}).
- Codes d'authentification (par ex. HMAC-*).

Zoom sur les chiffres par bloc

FIGURE : Voyons ça de plus près bis !

Chiffre par bloc, définition

- Principe : on chiffre les données par blocs de taille fixe.

Chiffre par bloc

Un chiffre par bloc est une application $\mathcal{E} : \{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}^n$ t.q. $\forall k \in \{0,1\}^\kappa$, $\mathcal{E}(k,\cdot)$ est une permutation de $\{0,1\}^n$ (est inversible).

Chiffre par bloc, définition

- Principe : on chiffre les données par blocs de taille fixe.

Chiffre par bloc

Un chiffre par bloc est une application $\mathcal{E} : \{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}^n$ t.q. $\forall k \in \{0,1\}^\kappa$, $\mathcal{E}(k,\cdot)$ est une permutation de $\{0,1\}^n$ (est inversible).

Chiffre par bloc idéalisé

- Un chiffre par bloc de clefs de taille κ définit 2^κ permutations.
- Pour un chiffre “idéal”, chacune de ces permutations est tirée aléatoirement parmi les $2^n!$ permutations de blocs de n bits possibles.

Chiffre par bloc idéalisé

- Un chiffre par bloc de clefs de taille κ définit 2^κ permutations.
- Pour un chiffre “idéal”, chacune de ces permutations est tirée aléatoirement parmi les $2^n!$ permutations de blocs de n bits possibles.

Construction itérative d'un chiffre par bloc

- Un chiffre par bloc est généralement construit autour :
  - d'une fonction de tour ;
  - d'un algorithme de génération de sous-clefs.
- Le chiffre final n'est que la composition de la fonction de tour un certain nombre de fois (de l'ordre de 10-100 en fonction des chiffres).

Exemple de structure : les réseaux de substitution-permutation

- La fonction de tour est composée :
  - D'une couche de substitution non-linéaire,
  - D'une couche de diffusion linéaire.

Exemple de couche non-linéaire

Substitution explicite par groupes de 4 bits *via* une table.

Exemple (PRESENT) :

x	0	1	2	3	4	5	6	7
S(x)	c	5	6	b	9	0	a	d
x	8	9	a	b	c	d	e	f
S(x)	3	e	f	8	4	7	1	2

Exemple de couche linéaire

Permutation de bits. Exemple (PRESENT) :
$P(i) = i \lll 2$, $i \in \{0 \dots 63\}$.

Exemple de structure : les réseaux de substitution-permutation

- La fonction de tour est composée :
  - D'une couche de substitution non-linéaire,
  - D'une couche de diffusion linéaire.

Exemple de couche non-linéaire

Substitution explicite par groupes de 4 bits *via* une table.

Exemple (PRESENT) :

x	0	1	2	3	4	5	6	7
S(x)	c	5	6	b	9	0	a	d
x	8	9	a	b	c	d	e	f
S(x)	3	e	f	8	4	7	1	2

Exemple de couche linéaire

Permutation de bits. Exemple (PRESENT) :
$P(i) = i \lll 2$, $i \in \{0 \dots 63\}$.

Exemple de structure : les réseaux de substitution-permutation

- La fonction de tour est composée :
  - D'une couche de substitution non-linéaire,
  - D'une couche de diffusion linéaire.

Exemple de couche non-linéaire

Substitution explicite par groupes de 4 bits *via* une table.

Exemple (PRESENT) :

x	0	1	2	3	4	5	6	7
S(x)	c	5	6	b	9	0	a	d
x	8	9	a	b	c	d	e	f
S(x)	3	e	f	8	4	7	1	2

Exemple de couche linéaire

Permutation de bits. Exemple (PRESENT) :
$P(i) = i \lll 2$, $i \in \{0 \dots 63\}$.

Exemple de structure : les réseaux de substitution-permutation

- La fonction de tour est composée :
  - D'une couche de substitution non-linéaire,
  - D'une couche de diffusion linéaire.

Exemple de couche non-linéaire

Substitution explicite par groupes de 4 bits *via* une table.

Exemple (PRESENT) :

x	0	1	2	3	4	5	6	7
S(x)	c	5	6	b	9	0	a	d
x	8	9	a	b	c	d	e	f
S(x)	3	e	f	8	4	7	1	2

Exemple de couche linéaire

Permutation de bits. Exemple (PRESENT) :
$\mathrm{P(i)} = i \lll 2$, $i \in \{0 \ldots 63\}$.

Rôle des composants d'un SPN

Couche non-linéaire

- Donner un aspect "aléatoire" aux données.
- Par ex., pour une différence fixée en entrée de deux blocs, on peut observer beaucoup de différences en sortie.

Couche linéaire

- Diffuser l'aléa créé par la couche non-linéaire à travers le bloc entier.
- Par ex., s'assurer qu'une différence se propage à travers un nombre important de boîtes de substitution.

Rôle des composants d'un SPN

Couche non-linéaire

- Donner un aspect "aléatoire" aux données.
- Par ex., pour une différence fixée en entrée de deux blocs, on peut observer beaucoup de différences en sortie.

Couche linéaire

- Diffuser l'aléa créé par la couche non-linéaire à travers le bloc entier.
- Par ex., s'assurer qu'une différence se propage à travers un nombre important de boîtes de substitution.

Au fait...

- Un chiffre ne s'utilise jamais tel quel et jamais seul, mais avec un mode d'opération et un code d'authentification.
- Par ex. GCM (Galois Counter Mode).
- C'est un autre domaine d'étude (cf. introduction).

Au fait...

- Un chiffre ne s'utilise jamais tel quel et jamais seul, mais avec un mode d'opération et un code d'authentification.
- Par ex. GCM (Galois Counter Mode).
- C'est un autre domaine d'étude (cf. introduction).

Au fait...

- Un chiffre ne s'utilise jamais tel quel et jamais seul, mais avec un mode d'opération et un code d'authentification.
- Par ex. GCM (Galois Counter Mode).
- C'est un autre domaine d'étude (cf. introduction).

Et les attaques dans tout ça ?

Paramètres d'une attaque

La sévérité d'une attaque se caractérise par un certain nombre de paramètres comme :

- L'objectif de l'attaque (distinguer le chiffre de données aléatoires, retrouver la clef, retrouver les textes clairs depuis les chiffrés correspondant....)
- Sa complexité en temps (généralement évaluée en nombre équivalent de calcul du chiffre).
- Sa complexité en mémoire.
- Sa complexité en données.
- Le type de données nécessaires (clair connu/choisi, chiffré choisi...).
- Le modèle d'attaque (clef unique, clefs liées....)
- Sa probabilité de succès.

Paramètres d'une attaque

La sévérité d'une attaque se caractérise par un certain nombre de paramètres comme :

- L'objectif de l'attaque (distinguer le chiffre de données aléatoires, retrouver la clef, retrouver les textes clairs depuis les chiffrés correspondant....)
- Sa complexité en temps (généralement évaluée en nombre équivalent de calcul du chiffre).
- Sa complexité en mémoire.
- Sa complexité en données.
- Le type de données nécessaires (clair connu/choisi, chiffré choisi...).
- Le modèle d'attaque (clef unique, clefs liées....)
- Sa probabilité de succès.

Paramètres d'une attaque

La sévérité d'une attaque se caractérise par un certain nombre de paramètres comme :

- L'objectif de l'attaque (distinguer le chiffre de données aléatoires, retrouver la clef, retrouver les textes clairs depuis les chiffrés correspondant....)
- Sa complexité en temps (généralement évaluée en nombre équivalent de calcul du chiffre).
- Sa complexité en mémoire.
- Sa complexité en données.
- Le type de données nécessaires (clair connu/choisi, chiffré choisi...).
- Le modèle d'attaque (clef unique, clefs liées....)
- Sa probabilité de succès.

Paramètres d'une attaque

La sévérité d'une attaque se caractérise par un certain nombre de paramètres comme :

- L'objectif de l'attaque (distinguer le chiffre de données aléatoires, retrouver la clef, retrouver les textes clairs depuis les chiffrés correspondant....)
- Sa complexité en temps (généralement évaluée en nombre équivalent de calcul du chiffre).
- Sa complexité en mémoire.
- Sa complexité en données.
- Le type de données nécessaires (clair connu/choisi, chiffré choisi...).
- Le modèle d'attaque (clef unique, clefs liées....)
- Sa probabilité de succès.

Paramètres d'une attaque

La sévérité d'une attaque se caractérise par un certain nombre de paramètres comme :

- L'objectif de l'attaque (distinguer le chiffre de données aléatoires, retrouver la clef, retrouver les textes clairs depuis les chiffrés correspondant....)
- Sa complexité en temps (généralement évaluée en nombre équivalent de calcul du chiffre).
- Sa complexité en mémoire.
- Sa complexité en données.
- Le type de données nécessaires (clair connu/choisi, chiffré choisi...).
- Le modèle d'attaque (clef unique, clefs liées....)
- Sa probabilité de succès.

Paramètres d'une attaque

La sévérité d'une attaque se caractérise par un certain nombre de paramètres comme :

- L'objectif de l'attaque (distinguer le chiffre de données aléatoires, retrouver la clef, retrouver les textes clairs depuis les chiffrés correspondant....)
- Sa complexité en temps (généralement évaluée en nombre équivalent de calcul du chiffre).
- Sa complexité en mémoire.
- Sa complexité en données.
- Le type de données nécessaires (clair connu/choisi, chiffré choisi...).
- Le modèle d'attaque (clef unique, clefs liées....)
- Sa probabilité de succès.

Paramètres d'une attaque

La sévérité d'une attaque se caractérise par un certain nombre de paramètres comme :

- L'objectif de l'attaque (distinguer le chiffre de données aléatoires, retrouver la clef, retrouver les textes clairs depuis les chiffrés correspondant....)
- Sa complexité en temps (généralement évaluée en nombre équivalent de calcul du chiffre).
- Sa complexité en mémoire.
- Sa complexité en données.
- Le type de données nécessaires (clair connu/choisi, chiffré choisi...).
- Le modèle d'attaque (clef unique, clefs liées....)
- Sa probabilité de succès.

Attaques (cont.)

On distingue :

- Les attaques génériques, qui marchent contre n'importe quelle primitive. Par ex. :
  - Attaque par force brute (en temps ou en mémoire) ;
  - Les compromis temps/mémoire génériques ;
  - etc.
- Les attaques dédiées.
- On considère généralement qu'un chiffre est "cassé théoriquement" quand la complexité d'une attaque dédiée est inférieure à la meilleure attaque générique dans un modèle équivalent.

Attaques (cont.)

On distingue :

- Les attaques génériques, qui marchent contre n'importe quelle primitive. Par ex. :
  - Attaque par force brute (en temps ou en mémoire) ;
  - Les compromis temps/mémoire génériques ;
  - etc.
- Les attaques dédiées.
- On considère généralement qu'un chiffre est "cassé théoriquement" quand la complexité d'une attaque dédiée est inférieure à la meilleure attaque générique dans un modèle équivalent.

Attaques (cont.)

On distingue :

- Les attaques génériques, qui marchent contre n'importe quelle primitive. Par ex. :
  - Attaque par force brute (en temps ou en mémoire) ;
  - Les compromis temps/mémoire génériques ;
  - etc.
- Les attaques dédiées.
- On considère généralement qu'un chiffre est "cassé théoriquement" quand la complexité d'une attaque dédiée est inférieure à la meilleure attaque générique dans un modèle équivalent.

Exemples de types d'attaques dédiées ?

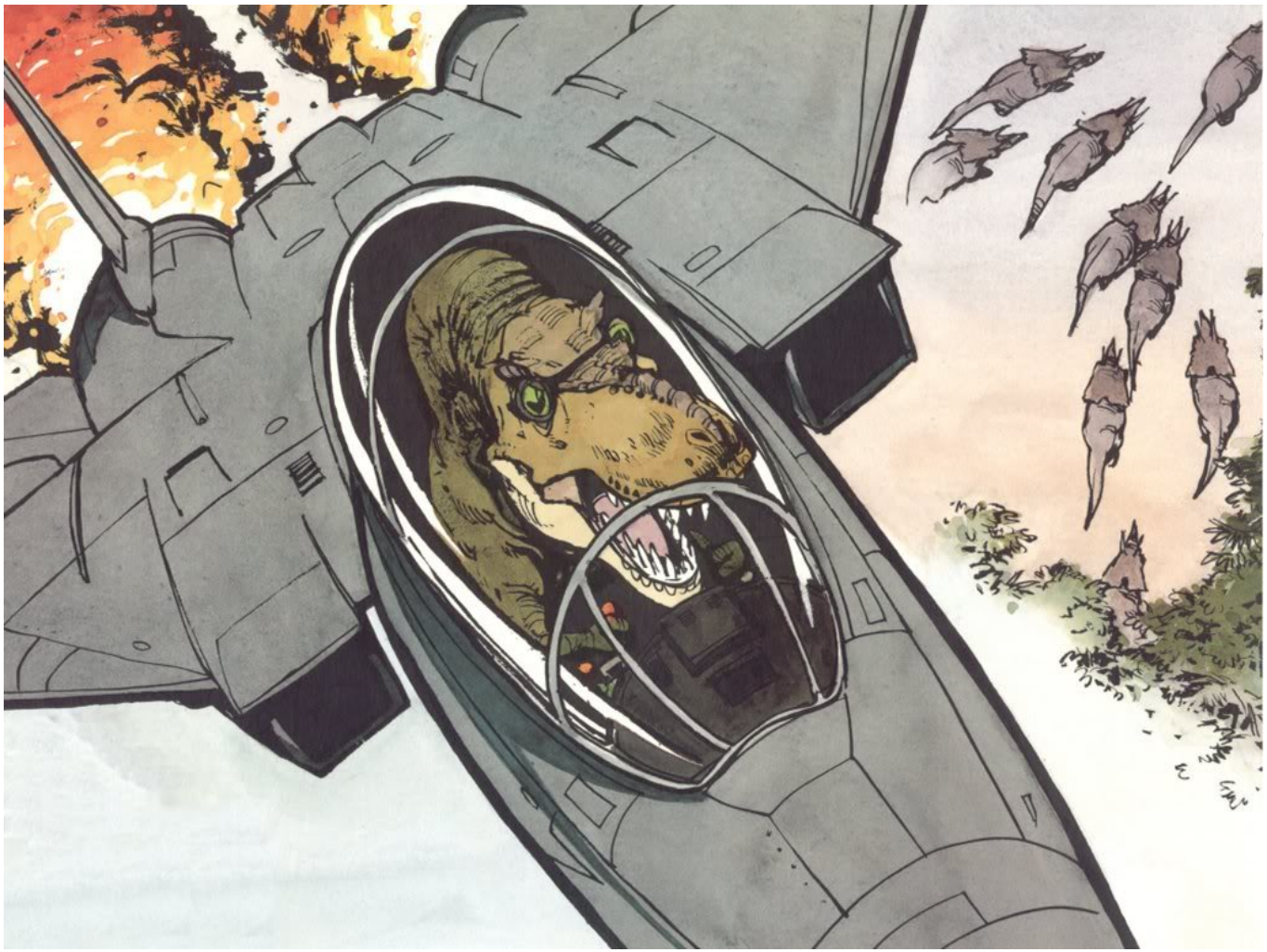

FIGURE : Une attaque brutale....

Tout a une fin....

Des questions ?

En tout cas on recrute (France, Singapour...) ! ! !

$\rightsquigarrow$ pierre.karpman@gmail.com / TDs de PROG1