Introduction

Utilisation de l'algorithme FF1

Version 1.0

Auteurs:
Emmanuel Coste : Lead Developer
Thomas Ramahefasolo : Head of Customer Projects
Bruno Grieder : CTO

Table des matières

 © Copyright 2018-2022 Cosmian. All rights reserved

Ce document étudie l'utilisation du **FPE** pour le chiffrement d'une base de données *Oracle*, et en particulier l'utilisation de l'algorithme **FF1** pour le chiffrement des champs de type *entier*, *décimal* et *date*.

© Copyright 2018-2022 Cosmian. All rights reserved

1 Présentation succinte

En cryptographie, le **chiffrement préservant le format**, dit **FPE** pour *Format-Preserving Encryption*, fait référence à un algorithme de chiffrement dont la sortie, **le texte chiffré**, présente le même format que l'entrée, **le texte en clair**. La signification de **format** varie. Généralement, seuls des ensembles finis de caractères sont utilisés ; numérique, alphabétique ou alphanumérique. L'exemple fréquemment donné semble celui du chiffrement d'un numéro de carte de crédit à 16 chiffres : le chiffré qui résulte du chiffrement de ce numéro de carte est composé également de 16 chiffres.

© Copyright 2018-2022 Cosmian. All rights reserved

2 Motivation du FPE

L'une des motivations de l'utilisation de FPE est relative aux problèmes d'intégration du chiffrement dans une application existante, avec un modèle de données déjà défini.

L'ajout du chiffrement à une telle application peut être difficile si le modèle de données doit être modifié, car cela implique généralement de modifier les limites de longueur des champs ou les types de données. Par exemple, la sortie d'un **chiffrement par bloc** transformerait le numéro de carte de crédit en une valeur hexadécimale dont la taille excède celle du champ d'origine.

De plus, l'utilisation d'un mode de chiffrement classique entraîne l'utilisation (**et le stockage**) d'un ou plusieurs vecteurs d'initialisation, ce qui peut être problématique dans 1 modèle de données ce stockage n'a pas été prévu initialement.

Le **FPE** tente de simplifier ce processus de transition en préservant le formatage et la longueur des données d'origine, permettant un remplacement instantané des valeurs en clair par leurs textes chiffrés dans l' application concernée.

© Copyright 2018-2022 Cosmian. All rights reserved

3 Comparaison avec un chiffrement par bloc

Un chiffrement par bloc de *n* bits est techniquement un **FPE** sur l'ensemble $0, ..., 2^{n-1}$. Si un FPE est nécessaire sur l'un de ces ensembles de taille standard (par exemple, $n = 64$ pour DES et $n = 128$ pour AES), un chiffrement par bloc de la bonne taille peut être utilisé.

Cependant, dans une utilisation classique, un chiffrement par bloc est utilisé dans un mode de fonctionnement qui lui permet de chiffrer des messages arbitrairement longs, et en utilisant un vecteur d'initialisation. Dans ce mode, un chiffrement par bloc **n'est pas** un FPE.

L'utilisation de **FPE** implique généralement l'usage de "petits" domaines où n peut même être inférieur ou égal à 10 [1].

Algorithme FF1

 © Copyright 2018-2022 Cosmian. All rights reserved

4 Standardisation

Plusieurs constructions de **FPE** sont apparues depuis 1981 [2] mais seules 2 ont été standardisées *FF1* et *FF3* [3].

Ces 2 méthodes sont basées sur un schéma de Feistel et utilisent l'AES[1] comme PRF[2].

[1] Advanced Encryption Standard
[2] Pseudo Random Function

© Copyright 2018-2022 Cosmian. All rights reserved

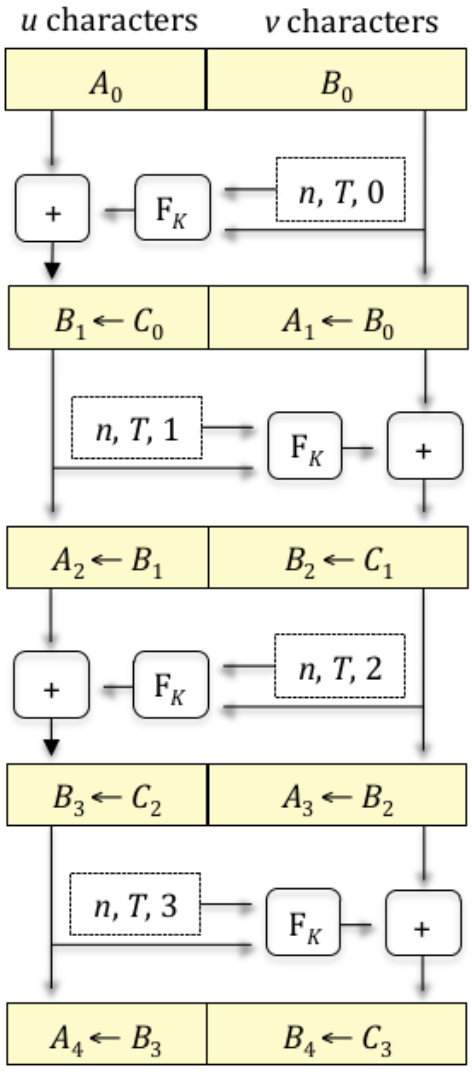
5 Schéma de Feistel équilibré

L'algorithme *FF1* repose sur l'utilisation d'un schéma de Feistel à 10 tours (8 pour *FF3*). Le choix d'un schéma de Feistel est détaillé dans [1] Annexe D. Ce schéma se déroule en plusieurs **tours** où chaque tour effectue la transformation réversible suivante :

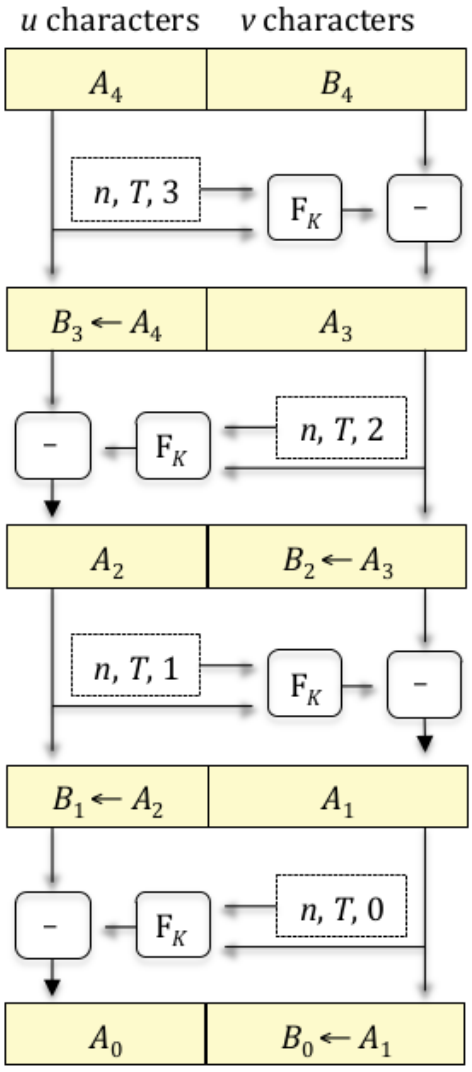
- la donnée est découpée en deux parties (de même taille, le schéma est équilibré)
- une fonction de tour (exploitant une clé de tour) est appliquée à une des deux parties de la donnée
- le rôle des deux parties est permuté au tour suivant

La figure suivante illustre 4 tours.

 © Copyright 2018-2022 Cosmian. All rights reserved



Encryption



Decryption

© Copyright 2018-2022 Cosmian. All rights reserved

6 Paramètres de FF1

Les paramètres de FF1 sont les suivants :

- *radix* : le nombre de caractères dans l'alphabet
- *minlen*: la taille minimale d'un message à chiffrer
- *maxlen*: la taille maximale d'un message à chiffrer

Ces 3 paramètres influent sur la sécurité du système et doivent respecter les conditions suivantes du standard :

- $radix \in [2..2^{16}]$,
- $radix^{minlen} \geq 100$,
- $2 \leq minlen \leq maxlen < 2^{32}$.

Le seuil donné dans la deuxième équation est d'ailleurs remonté à $1.000.000$ dans [3] Annexe A, soit $radix^{minlen} \geq 1000000$. Cette mesure est particulièrement prudente voire très conservatrice ([1] Annexe H).

6.1 Tweak

Le paramètre "tweak" du FPE, dont la taille n'est pas encadrée dans le standard, améliore la sécurité en particulier quand les clairs sont de petite taille. Ce paramètre public sert à contextualiser le chiffrement du clair en fonction de tout ce qui peut se rapporter au clair et minimise les attaques par corrélation.

Idéalement ce paramètre doit être unique pour chaque message en clair.

Sécurité

 © Copyright 2018-2022 Cosmian. All rights reserved

7 Définition de la sécurité

Dans la littérature cryptographique, la mesure d'un "bon" FPE est de savoir si un attaquant peut distinguer le FPE d'une permutation vraiment aléatoire. Différents types d'attaquants sont modélisés, selon leur faculté à disposer d'un oracle ou d'un ensemble de paires de texte chiffré/texte clair.

© Copyright 2018-2022 Cosmian. All rights reserved

8 Cryptanalyse

La cryptanalyse basée sur les schémas de Feistel [4] a attiré notre attention (nous n'avons d'ailleurs pas trouvé d'autres attaques sur FF1 dans la littérature). L'article décrit comment construire un distingueur d'une permutation aléatoire avec une variable de tour du schéma de Feistel. Cette construction est d'ailleurs possible pour les autres variantes du FPE comme le montre la figure suivante :

Algorithm	Rounds	Block size	Keysize	Complexity Time	Complexity Data
FEA-1	12	8	128	2^{36}	2^{32}
FEA-1	14	8	192	2^{44}	2^{40}
FEA-1	16	8	256	2^{52}	2^{48}
FEA-2	18	8	128	2^{60}	2^{56}
FEA-2	21	8	192	2^{72}	2^{68}
FEA-2	24	8	256	2^{84}	2^{80}
FF1	10	20	128	2^{70}	2^{60}
FF3-1	8	40	128	2^{100}	2^{80}
Generic[3]	$2r$	$2n$	-	$2^{2n(r-1.5)}$	$2^{2n(r-1.5)-n}$

Table 1: Comparison of Distinguishing Attacks.

Dans le cas de FEA-1 et FEA-3, l'article fournit également une attaque supplémentaire permettant la reconstruction de la clef de chiffrement (de 192 ou 256 bits) avec une complexité de 2^{136}. En ce qui concerne FF1, la construction du distingueur FF1 a complexité en espace de 2^60 (pour des clairs de 20 bits) et une complexité en temps de 2^70 comme le détaille l'article :

> As discussed in Section 3 we have a distinguisher on FFX with a data complexity of $2^{2n((r-1)-\frac{1}{2})-n}$ and a time complexity of $2^{2n((r-1)-\frac{1}{2})}$. Given that FF1 has 10 rounds ($r = 5$), and a minimum block size of 20 bit ($n = 10$), the data complexity is $2^{20 \cdot 3.5-10} = 2^{60}$ and the time complexity is $2^{20 \cdot 3.5} = 2^{70}$.

Enfin l'article donne deux contre-mesures à cette attaque sur FF1 : l'une consiste à réduire la taille du tweak afin de réduire le volume de données que l'attaquant peut obtenir et l'autre consiste à augmenter le nombre de tours de Feistel à 18. Cette dernière contre-mesure est d'après les auteurs de l'article la meilleure option. La calcul du nombre de tour **18** est expliqué dans [1] Annexe H.

Ainsi, avec cette augmentation du nombre de tours, la complexité en espace pour la construction du distingueur FF1 devient 2^{150} et en temps de 2^{160} (et donc au-delà du niveau de sécurité de 128 bits minimum attendu).

Chiffrement de base de données

 © Copyright 2018-2022 Cosmian. All rights reserved

L'étude se concentre sur le chiffrement d'une base de données **Oracle**, et en particulier les champs de type *entier*, *décimal* et *date*.

 © Copyright 2018-2022 Cosmian. All rights reserved

9 Recensement des champs concernés

Les types concernés sont listés (à minima) dans la documentation Oracle à ces adresses :

- https://docs.oracle.com/cd/B19306_01/server.102/kkb14200/sql_elements001.htm
- https://docs.oracle.com/en/database/oracle/oracle-database/19/refrn/datatype-limits.html#GUID-963C79C9-9303-49FE-8F2D-C8AAF04D3095.

En reprenant ces documentations, on identifie :

- des nombres entiers signés ou non-signés :
  - LONG
  - NUMERIC (ANSI Datatype)
  - INTEGER (ANSI Datatype)
  - INT (ANSI Datatype)
  - SMALLINT (ANSI Datatype)
- des nombres décimaux à précision variable :
  - NUMBER : valeur minimum : $999...(389's) \times 10^{125}$ et valeur maximum : $-999...(389's) \times 10^{125}$
  - BINARY_FLOAT : valeur minimum : $1.17549e^{-38}F$ et valeur maximum : $3.40282e^{+38}F$
  - BINARY_DOUBLE : valeur minimum : $2.22507485850720e^{-308}$ et valeur maximum : $1.79769313486231e^{+308}$
  - DECIMAL (ANSI Datatype)
  - FLOAT (ANSI Datatype)
  - DOUBLE PRECISION (ANSI Datatype)
  - REAL (ANSI Datatype)
- des dates :
  - DATE compris entre le 1er janvier 4712 avant JC au 31 décembre 9999 après JC
  - TIMESTAMP
  - INTERVAL

 © Copyright 2018-2022 Cosmian. All rights reserved

10 Propositions d'implémentation

La sécurité du chiffrement FPE est relative à la représentation sous forme de chaîne de caractères de l'entier à chiffrer.

Comme vu précédemment, le seuil de sécurité suggèré par le standard [3] Annexe A est fixé par l'équation $radix^{minlen} \geq 1000000$.

Cette sécurité impose donc que le chiffrement d'un nombre n'est possible que si ce nombre se représente sous forme de chaîne de caractères par au moins 6 chiffres.

10.1 Chiffrement des entiers

Une solution pour respecter cette contrainte peut être l'utilisation si nécessaire d'un padding "à gauche" dont la taille est calculée en fonction de la taille du type en base de données.

Par exemple, pour le type *NUMBER(38)*, ce padding peut être le suivant :

- compléter par des zéros à gauche les nombres dont la représentation a une taille inférieure à 38 (jusqu'à obtenir une chaîne de 37 caractères, 37 pour ne pas créer de débordement en base)

Pour les nombres signés, il paraît risqué de chiffrer le signe : en effet des contrôles applicatifs sur la valeur du champ pourraient générer des erreurs inattendues.

10.2 Chiffrement des nombres décimaux

Concernant le chiffrement des nombres décimaux, le chiffrement FPE doit s'appliquer à minima sur la mantisse et l'exposant du nombre en respectant, tout comme le chiffrement des entiers par FPE, la taille maximale possible du type de données. Un padding sur la mantisse peut-être utilisé comme précédemment. Dans la notation scientifique, la virgule peut être ignorée pour le chiffrement et réinsérée une fois le nombre chiffré afin de respecter le format d'origine.

10.3 Chiffrement des dates

Le format DATE est un format très contraint et l'utilisation d'un padding paraît impossible. De plus, le chiffrement de ce champ (en respectant les valeurs limites) peut générer probablement des erreurs applicatives. S'il faut le chiffrer, disons plutôt le brouiller, nous conseillons plutôt une obfuscation avec des valeurs limites "attendues" par l'application (par exemple fixer un interval restreint pour l'année). Le FPE ne paraît pas adapté pour le chiffrement de ces champs.

 © Copyright 2018-2022 Cosmian. All rights reserved

10.4 Tweak

Le tweak pourrait être le résultat d'un hash d'une combinaison relative aux informations du nombre en base (par exemple nom de la table, de la colonne, etc). Ce calcul dynamique ne vient pas sans inconvénient mais a le mérite de ne pas nécessiter un stockage de l'ensemble des tweaks.

© Copyright 2018-2022 Cosmian. All rights reserved

References

[1] T. S. Mihir Bellare Phillip Rogaway, 'The FFX mode of operation for format-preserving encryption, draft 1.1'. 2010.

[2] N. B. of Standards, 'FIPS PUB 74. Guidelines for implementing and using the NBS data encryption'. National Bureau of Standards, 1981.

[3] M. D. NIST, 'Recommendation for block cipher modes of operation: Methods for format-preserving encryption'. Special Publication (NIST SP), National Institute of Standards; Technology, Gaithersburg, MD, 2016, doi: https://doi.org/10.6028/NIST.SP.800-38G.

[4] O. Dunkelman, A. Kumar, E. Lambooij, and S. K. Sanadhya, 'Cryptanalysis of feistel-based format-preserving encryption'. Cryptology ePrint Archive, Report 2020/1311, 2020.

 © Copyright 2018-2022 Cosmian. All rights reserved