SE - BLOC2 - UE18 - Mathématiques et physiques appliquées

Projet 2023-2024

Collet Marianne et Ludewig François

1 Objectifs

L'objectif de ce laboratoire est la manipulation des concepts mathématiques du cours théorique au travers de la réalisation de programmes en Python.

A la fin du laboratoire l'étudiant sera capable de :

1. réaliser un chiffrement simple de type symétrique moderne

2 Énoncé

Le projet consiste à rendre un programme codé en *Python* qui permet d'implémenter un chiffrement et un déchiffrement *CAST-256* suivant un mode d'opération ECB ou CBC au choix. Le programme doit pouvoir être appelé sur des fichiers *".txt".*

Afin de guider l'étudiant dans la programmation, un ensemble de stubs [1] est fourni. Les stubs sont organisés par fonctionnalité, chacune correspondant à un fichier python :

1. `utils.py` : ensemble de fonction de manipulation binaire et d'arithmétique modulo simple.
2. `key_generator.py` : réalise la génération de clé pour chacun des *round* du CAST-256.
3. `functions_cast256.py` : implémente les fonctions nécessaires au chiffrement et déchiffrement du cast-256.
4. `cast256.py` : met en œuvre le chiffrement et le déchiffrement d'un block de 128 bits
5. `mode_operatoire.py` : effectue le chiffrement via les modes ECB et CBC.

Il est demandé à l'étudiant de remplir chacun des stubs présents.

Étant donné la structure de l'algorithme, certaines fonctionnalités sont à réaliser en priorité. L'ordre suivant est suggéré :

1. `utils`
2. `functions_cast256`
3. `key_generator`
4. `cast256`
5. `mode_operatoire`

Chacun des fichiers est accompagné d'un fichier test (`foo.py` a le fichier associé `foo_test.py`). Ce fichier reprend un ensemble de tests unitaires pour chacune des stubs. Ces tests permettent de vérifier le fonctionnement correct de chacune des fonctions demandées avant de continuer la progression. Il est primordial que TOUS les tests d'un fichier réussissent avant de passer au

1. Fonction sans code ne contenant que la documentation de la fonction

suivant.

Pour rappel, pour faire tourner un test, il faut placer le fichier de test dans le même dossier que votre code et faire un run via *PyCharm*. Si vous voulez exécuter UN SEUL test, vous pouvez appeler en ligne de commande

```
> python3 test_foo.py Test_foo.test_XXX
```

avec XXX le nom de la fonction à tester. Vous pouvez aussi cliquer sur ▶ à côté du test dans *PyCharm*.

3 Évaluation

L'évaluation portera sur le code complété. Le critère de réussite minimal est le passage de tous les tests proposés et une implémentation correcte des différents principes impliqués dans le CAST-256.

La qualité du travail (utilisation intelligente de librairies et clarté/concision efficacité du code) permettra d'augmenter la note. Il en va de même pour l'implémentation du mode d'opération *CBC*.

Si jugé nécessaire, des questions sur le projet peuvent être posées lors de l'oral.

4 Modalités

Les fichiers contenant les stubs **complétés** sont à rendre pour le dimanche suivant la dernière séance de laboratoire à 23h59, dans l'espace du cours.

5 CAST-256

L'algorithme CAST-256 permet de chiffrer des données de manière symétrique. L'algorithme est apparu en 1998 et a été développé sur les bases du cast-128, via un *réseau de Feistel*. Ce chiffre a été candidat lors de l'appel pour l'AES du NIST [2]. Il a été recalé principalement pour sa non efficience. Les sections suivantes rappellent les éléments théoriques permettant d'appliquer le CAST-256.

Tous les détails manquant volontairement dans la description ci-dessous sont à trouver dans les références proposées à la fin de l'énoncé.

2. https://en.wikipedia.org/wiki/Advanced_Encryption_Standard_process

Le CAST-256 chiffre des blocs de 128 bits avec une clé de 256 bits. Même si il est possible d'utiliser des clés de taille plus petite, nous nous limiterons à la taille maximale possible.

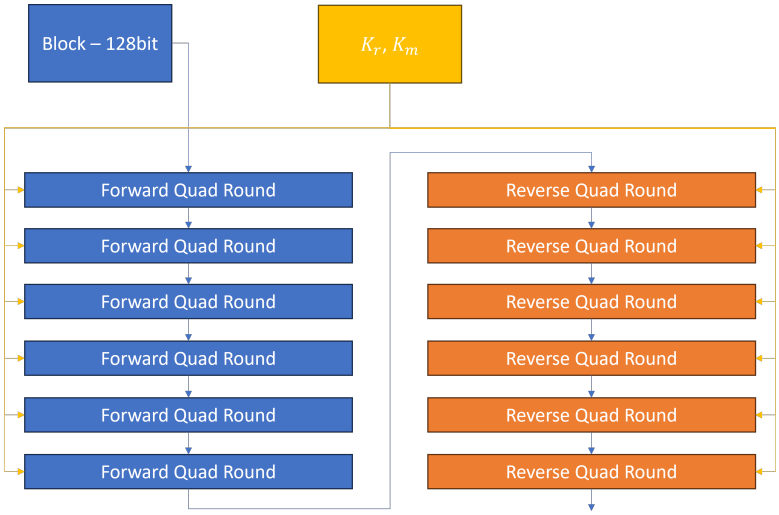

FIGURE 1 – Représentation du principe du CAST-256.

La figure 1 représente les grandes étapes du CAST-256. Le bloc est chiffré par 6 passages consécutifs dans la méthode "Forward Quad Round) suivis de 6 passages dans la méthode "Reverse Quad Round". Au total, la CAST-256 effectue 48 rounds. Des clés de rotation et de masque sont utilisées lors de l'appel de chaque fonction.

5.1 Quad Round

Les fonctions Forward et Reverse Quad Round sont illustrées sur la figure 2. Ces fonctions composent un réseau de Feistel. Ce type d'approche diffère du schéma classique de Feistel mais possède les mêmes propriétés essentielles.

La fonction applique le principe du schéma de Feistel étendu à 4 parties au lieu de 2. Pour ce faire, elle utilise 3 fonctions différentes : f_1, f_2, f_3, avec jeu de clé de rotation et de masque spécifique à chaque appel. Toutes ces clés sont dérivées de la clé de chiffrement.

La fonction "Forward" peut être modélisée par les équations :

$$c' = c \oplus f_1(k_r^0, k_m^0, d) \tag{1}$$
$$b' = b \oplus f_1(k_r^1, k_m^1, c') \tag{2}$$
$$a' = a \oplus f_1(k_r^2, k_m^2, b') \tag{3}$$
$$d' = d \oplus f_1(k_r^3, k_m^3, a') \tag{4}$$
$$\tag{5}$$

les lettres a, b, c et d représentent les parties 32 bits de départ et les lettres a', b', c' et d' les parties 32 bits de sortie. La fonction "Reverse" effectue les mêmes opérations dans l'ordre

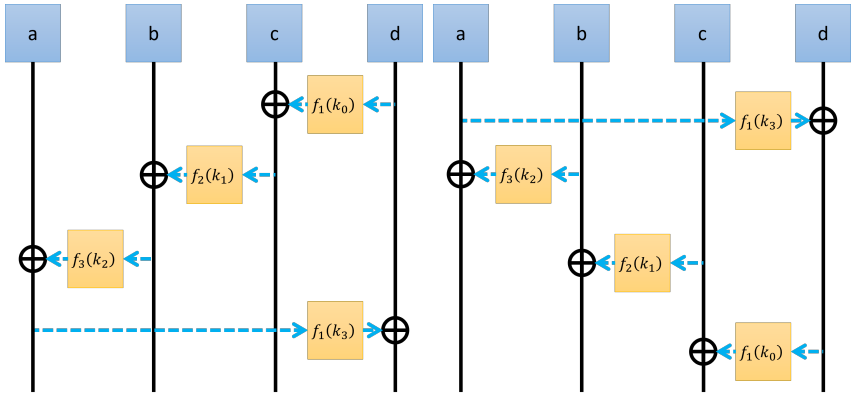

FIGURE 2 – Représentation des fonctions : Forward Quad Round (à gauche) et Reverse Quad Round (à droite).

inverse :

$$d'' = d' \oplus f_1(k_r^3, k_m^3, a') \tag{6}$$
$$a'' = a' \oplus f_1(k_r^2, k_m^2, b') \tag{7}$$
$$b'' = b' \oplus f_1(k_r^1, k_m^1, c') \tag{8}$$
$$c'' = c' \oplus f_1(k_r^0, k_m^0, d'') \tag{9}$$
$$\tag{10}$$

Ici a', b', c' et d' sont les valeurs d'entrée et a", b", c" et d" sont les valeurs de sortie. Il peut être montré trivialement que l'application de ces fonctions successivement avec le même jeux de clés, permet de retrouver les valeurs d'origine.

5.2 Sous-fonctions

De fait, ce réseau permet de déchiffrer sans jamais inverser les sous fonctions f_1, f_2 et f_3. Ces fonctions peuvent donc être à sens unique, non inversibles.

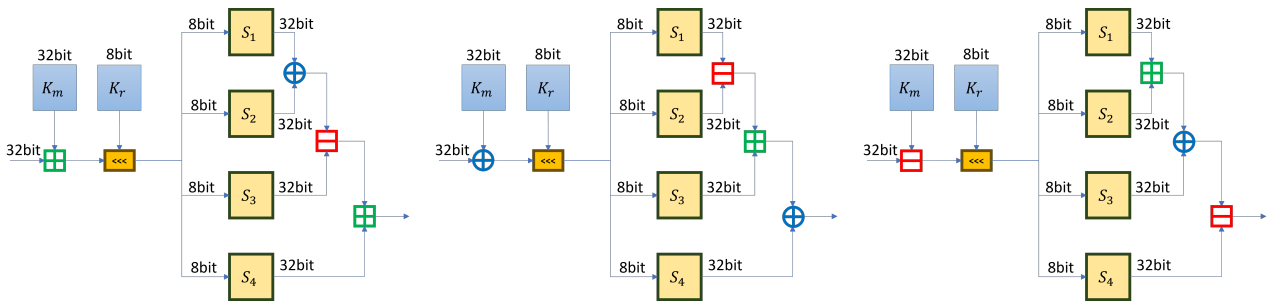

FIGURE 3 – Représentation des sous fonctions : f_1 (à gauche), f_2 (au milieu) et f_3 (à droite).

La figure 3 illustre schématiquement les sous-fonctions. Les symboles + et − dans un carré signifient que l'opération est effectuée dans un espace modulo. Les trois fonctions font appel aux mêmes boîtes de substitution.

5.3 Substitution

Les boîtes de substitution sont des tableaux dont les valeurs sont fixées par la documentation du chiffre. La taille de chaque élément est aussi définei et fixe. Pour le CAST-256, les boîtes de substitution sont composées de 256 valeurs codées sur 32 bits comme illustré sur la figure 4

32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

FIGURE 4 – Représentation des boîtes de substitution comme une structure composée d'un nombre fixe d'éléments de même taille (à gauche). Chaque élément est identifié par sa position (à droite)

La position des valeurs dans le tableau est numérotée de 0 à 255 (figure 4 à droite). La valeur d'entrée est alors codée sur 8 bits. La boîte retourne alors les 32 bits enregistrés à la position de la valeur d'entrée. On dit que les boîtes effectuent une expansion en passant de 8 bits à 32 bits.

5.4 Génération des clés

Les clés pour chacun des *rounds* sont obtenues à partir d'une clé globale. Pour les obtenir, la clé de chiffrement codée sur 256 bit subit un cycle de traitement présenté sur la figure 5.

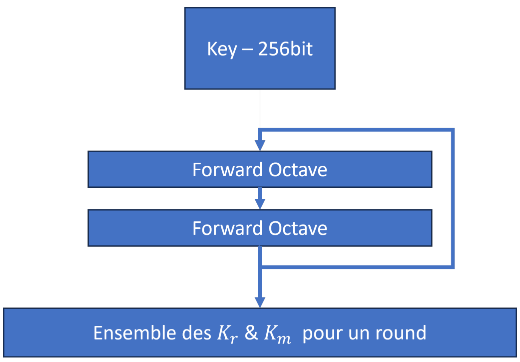

FIGURE 5 – Génération des clés pour chacun des *rounds*.

Tous les deux passages dans la fonction "Forward Octave", les clés de rotation et de masque sont extraites des 256 bits obtenus. Cette fonction est similaire à la fonction "Forward Quad Round" et fonctionne sur 256 bits au lieu de 128 bits. Son déroulé est illustré sur la figure 6

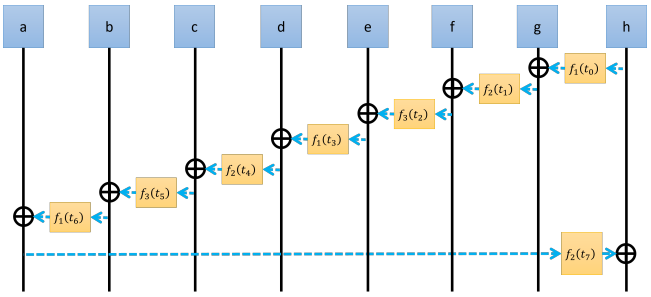

FIGURE 6 – Génération des clés pour chacun des *rounds*.

Afin de faire appel à la fonction "Forward Octave", des clés de rotation et de masque doivent être créées. Leur valeur ne dépend ni de la clé, ni du bloc à chiffrer. Elles sont donc identiques à tous les chiffrements.

5.5 Mode d'opération

Le mode d'opération le plus simple consiste simplement à appliquer le DES à des blocs successifs de 64 bits de données à chiffrer. Ce *mode d'opération* est connu sous le nom *Electronic Code Book* (ECB) et est représenté à la figure 7.

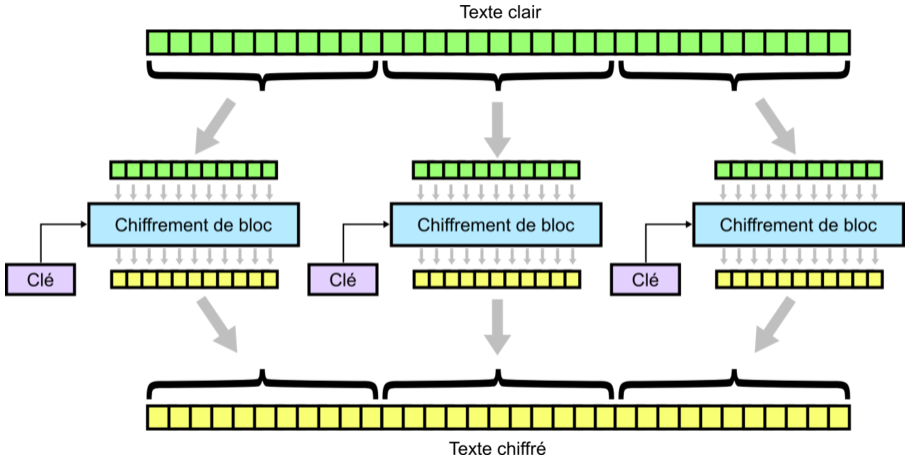

FIGURE 7 – Mode d'opération ECB.

Néanmoins, lorsque ce mode est utilisé, il est possible de voir apparaître certains motifs, ce qui est en contradiction avec les principes de la cryptographie. Ceci vient du fait que des blocs initiaux identiques donneront des blocs chiffrés identiques.

Plusieurs *modes d'opérations* ont été développés afin de palier à ce problème. On pense notamment au *Cipher Block Chaining* (CBC).

Le mode d'opération CBC consiste à effectuer sur chacun des blocs à chiffrer une opération de XOR avec le bloc chiffré précédemment avant de passer dans la "moulinette" DES, comme représenté à la Figure 8. Pour le premier bloc à coder, un vecteur d'initialisation de 64 bits est utilisé.

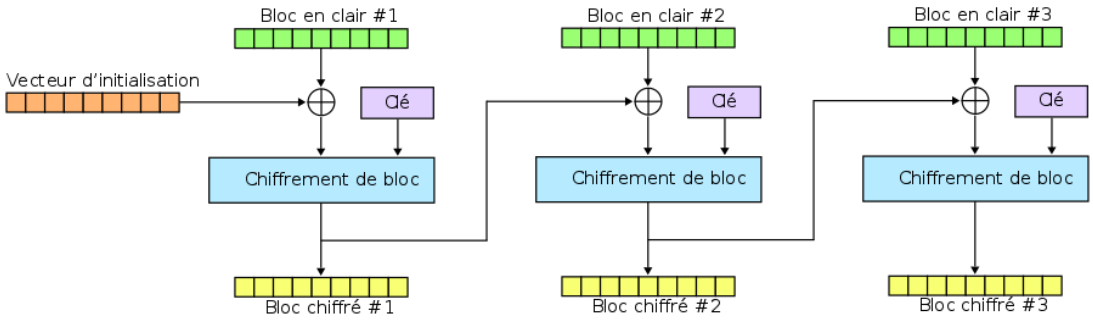

FIGURE 8 – Mode d'opération CBC.

6 Références

Afin de compléter le bref rappel théorique, un ensemble de références est proposé à l'étudiant :

- La norme officielle du CAST-256 : RFC 2144
- La page wikipédia qui présente le CAST-256.
- La page wikipédia qui présente le CAST-128.
- L'autre page wikipédia qui explique les modes d'opérations. (les dernières images viennent d'ici)
- Un article qui présente une partie du dépôt fait auprès du NIST en 1998.