فصل–3 التشفير الكتلي

3.1 المقدمة

- ذكرنا في الفصل السابق كيفية استخدام دوال البعثرة شبه العشوائية (او ماتسمى اصطلاحا بالتشفير الكتلي) في تصميم مناهج التشفير الآمنة ، بحيث ان أمنية منهج التشفير تعتمد على "افتراض" وجود تلك الدوال.
- في هذا الفصل سوف نعرض كيفية بناء هذه الدوال عمليا. تصطلح الكثير من الكتب والمصادر على اعتبار التشفير الكتلي مناهج تشفير مستقلة ، والواقع ان التشفير الكتلي ليس كذلك بل هو مجرد دوال بعثرة شبه عشوائية ، تستخدم في تصميم مناهج التشفير بانماط عمليات معيّنة. للبساطة ، سوف نتّفق على تسمية دوال البعثرة شبه العشوائية بالتشفير الكتلي.
- نؤكد هنا ، على ان هياكل التشفير الكتلي التي سوف نذكرها خالية من اثباتات بكونها آمنة ، ولا تعتمد ايضا على "افتراضات" صعبة رياضيا بحيث يمكن برهنة امنيتها عند تحقق تلك الافتراضات. كل مافي الامر ان هياكل التشفير الكتلي العملية صمدت لمدة طويلة بوجه محاولات كسر الشفرة. وهذه الحقيقة بحد ذاتها تعطينا سببا معقولا لاعتماد تلك الهياكل كأدوات آمنة. وقد يتسائل البعض كيف يمكن استخدام تلك الهياكل –غير المثبت امنيتها- في مناهج التشفير التي تتطلب اثباتات صارمة كما ذكرنا من قبل. للأجابة على هذا التساؤل يمكن القول ان تلك الهياكل هي ادوات واطئة المستوى بحيث تكون قابلة للكسر بسهولة من قبل الخصم ، وبالتالي فإن فشل الخصم في مهاجمتها يعد بحد ذاته سببا للاعتقاد بأمنيتها. كما يمكن القول ان تلك الهياكل يمكن استبدالها باخرى –في حال كسرها- بصورة اسهل من استبدال منهج التشفير ككل.
- في الواقع ، توجد طريقتان رئيستان لصميم هياكل التشفير الكتلي ، هما: شبكات (التعويض والبعثرة)، و شبكات Feistel. سوف نبدأ بعرض هاتين الطريقتين ومن ثم نقدّم شفرتي DES و AES ، اللتان تعتبران دوال بعثرة شبه عشوائية.

3.2 شبكات التعويض– البعثرة

- يفترض بالتشفير الكتلي ان يسلك سلوك البعثرة العشوائية بحيث لايمكن لاي خصم ان يميّز سلوكه عن دالة بعثرة عشوائية حقيقية.
- قدّم Shannon مفاهيم اساسية لغرض تصميم التشفير الكتلي.
  - الفكرة الاساسية هي ان يتم تقسيم المدخل الى اجزاء ، كل جزء يمرّر على دالة عشوائية صغيرة. تمزج مخرجات الدوال للاجزاء المختلفة مع بعضها البعض. تكرّر هذه العملية عدد من المرات (**دورات** rounds). تعمل الدوال العشوائية على **تشويش** المدخلات confusion بينما تعمل عملية المزج على **نشر** diffusion هذا التشويش على الدوال الاخرى. يضمن الاستخدام المتكرر لمفهومي التشويش و النشر لعدد من الدورات على جعل المخرجات التي مدخلاتها متشابه تبدو مختلفة تماما.
- تعتمد شبكات (التعويض-البعثرة) substitution-permutation network على المفاهيم اعلاه ، حيث ان التعويض substitution يقابل الدوال العشوائية الصغيرة ، بينما تقابل البعثرة permutation عملية المزج لمخرجات الدوال العشوائية.
- لاحظ ان هذه العملية تخضع لمنهج التشفير الضربي product cipher الذي اشرنا اليه في الفصل الاول ، والذي يشفّر النص بطرق تشفير مختلفة ، وهي تتمثل بتطبيق الطريقة التعويضية ومن ثم طريقة البعثرة ولعدد من الدورات.
- تسمّى دوال التعويض الصغيرة **بصناديق التعويض** S-boxes بينما تعرف عملية البعثرة بـmixing permutations.
- في هذا الهيكل ، يتم استلام المفتاح الخاص master-key ويتم اشتقاق **مفاتيح فرعية** sub-keys منه لكل دورة.
- تعرف عملية الاشتقاق **بجدولة المفتاح** key schedule.
- بعد نهاية كل دورة يتم تطبيق عملية XOR بين المفتاح الفرعي لتلك الدورة مع مدخلات تلك الدورة.
- وبذلك فإن العمليات الاساسية للدورة الواحدة في شبكات (التعويض-بعثرة) هي: عملية XOR بين المدخلات والمفتاح الفرعي ، تمرير الناتج على صناديق التعويض S-boxes ، واخيرا مزج النتائج بـ mixing permutation. تختلف هياكل التشفير الكتلي باختلاف صناديق التعويض وعملية المزج.
- نقدّم الأن الوصف الرياضي لشبكات (التعويض-البعثرة). ليكن لدينا عددين صحيحين موجبين t وx. نفترض ان النص الصريح والنص المشفر هما متجهات ثنائيات بطول tx. ليكن

$$S: \{0,1\}^t \rightarrow \{0,1\}^t$$

هو صندوق التعويض ، و

$$P: \{1, \dots, tx\} \rightarrow \{1, \dots, tx\}$$

هو عملية البعثرة. يعمل S على استبدال t من الثنائيات بثنائيات t اخرى. اما P فيبعثر tx من الثنائيات. تمثل nr عدد الدورات ، اما k فيمثل مفتاح التشفير الذي نشتق منه $(k_1, \dots, k_{nr+1})$ من المفاتيح الفرعية.

ليكن لدينا كتلة من البيانات بشكل خيط رمزي ثنائي $m = (m_1, \dots, m_{tx})$. يتم تشفير هذه الكتلة بتقسيمها الى x من الاجزاء $m_{<1>}, \dots m_{<x>}$ ، كل جزء بطول t من الثنائيات $m_{(i)} = (m_{(i-1)t+1}, \dots, m_{it})$. وبالتالي يكون $m = m_{<1>} || \dots || m_{<x>}$.

- يوضّح المنهج (3.1) كيفية عمل شبكات (تعويض-بعثرة). لاحظ ، ان آخر دورة لاتتضمن عملية تعويض ، كما ان اول واخر عملية على النص الصريح هي عملية Xor مع المفاتيح الفرعية وهي تعرف بعملية **التبييض** whitening ، والتي تهدف لمنع الخصم من البدء بعمليتي التشفير او فك الشفرة عندما لايكون المفتاح معلوما.

Construction (3.1): Substitution–permutation network

Input: plaintext block, m, S-box, S, permutation, P, number of rounds, nr, private key, k, part size,t

Output: ciphertext block, c

$state = m$
$x = |m|/t$
$(k_1, \dots, k_{nr+1}) = KeySchedule(k, nr)$
for $r = 1$ to $nr - 1$:
  $state = state \oplus k_r$
  for $i = 1$ to x:
    $u = S(state_{<i>})$
  $state = (P(state_1), \dots, P(state_{tx}))$
$state = state \oplus k_{nr}$
for $j = 1$ to x:
  $u = S(state_{<j>})$
$c = state \oplus k_{nr+1}$
return c

تكون خوارزمية اشتقاق المفاتيح الفرعية KeySchedule مختلفة باختلاف منهج التشفير المستخدم.

صناديق التعويض.يشترط في صناديق التعويض توفّر شرطين مهمين: يضمن الشرط الاول خاصية البعثرة وفك الشفرة ، اما الشرط الثاني فيضمن الأمنية.

الشرط الاول: شرط قابلية العكس. يشترط في صناديق التعويض ان تكون قابلة للعكس ، بمعنى ان تكون دوال من نوع 1-1 ، بحيث ان كل ادخال يقابله اخراج واحد فقط ، وهذا يضمن امكانية معرفة الادخال عند توفّر الاخراج.

الشرط الثاني: تاثير الانهيار avalanche effect. ويعنى ان التغيير البسيط في المدخلات سوف يتراكم خلال الدورات المتكررة مما يسبب ظهور مخرجات مختلفة تماما. تصمّم صناديق التعويض بحيث ان المدخلات التي تختلف بثنائية واحدة فقط على الاقل سوف تختلف مخرجاتها على الاقل بثنائيتين. فلو كان لدينا مدخلان مختلفان بثانية واحدة فقط ، فإن مخرجاتهما ستختلف بثانئيتين بعد الدورة الأولى ، اربع ثنائيات بعد الدورة الثانية ، ثمان ثنائيات بعد الدورة الثالثة ، وهكذا. بصروة عامة تؤثر الدورة i على 2^i من الثنائيات ، وبذلك فإن كتلة البيانات التي طولها 128 ثنائية تحتاج الى7 دورات لاكمال شرط الانهيار.

3.3 شبكات Feistel

- تشبه شبكة فيستل شبكة (التعويض-بعثرة) من حيث احتوائها على عمليات (صناديق التعويض ، المزج ، و اشتقاق المفاتيح الفرعية) ولكن تختلف عنها من حيث الهيكل العام.
- تمتاز صناديق التعويض S-boxes لشبكة Feistel بانها غير قابلة للعكس بالضرورة ، وبالتالي ، فإن شبكة Feistel تكون قابلة للعكس رغم استخدامها مكونات اصغر غير قابلة للعكس.
- تستخدم شبكة Feistel دالة داخلية f ، لايشترط فيها-كما ذكرنا- ان تكون قابلة للعكس. يتم تقسيم الادخال m لشبكة Feistel الى نصفين ، L و R ، وكل نصف يمرر على الدالة f. عندما يكون طول الادخال n من الثنائيات ، فإن ادخلات واخراجات الدالة f تكون بطول $n/2$.
- يمكن التعبير عن عمل دورة واحدة لشبكة Feistel كما يلي:
  1. قسّم الادخال m الى نصفين ، L و R.
  2. لكل قيمة $i = 1, \dots, nr$ (nr هو عدد الدورات)

أ. ليكن $w_1 \leftarrow R$ و $w_2 \leftarrow L \oplus f_i(R)$

ب. ليكن $L = w_1$ و $R = w_2$.

3. الاخراج هو (L, R).

لاتتطلب عملية فك الشفرة في شبكة Feistel تواجد معكوس الدالة الداخلية f وانما تتم بتطبيق الدالة f ايضا.

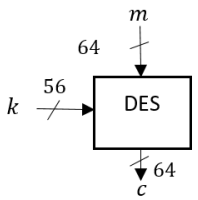
3.4 تشفير البيانات القياسي DES

تم تطوير معيار تشفير البيانات Data Encryption Standard (DES) عام 1970في IBM (وبمساعدة من قبل وكالة الامن الوطني National security agency) وتم تبنّيه من قبل مؤسسة FIPS (Fediral Information Processing Standard) الاميركية عام 1976.

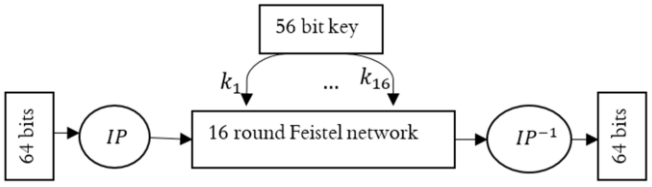
- تمتاز DES بقصر مفتاحها (56 ثناية) مما يجعلها غير آمنة حاليا ، حيث تستخدم بدلا عنها طريقة DES الثلاثية (Triple DES) التي تعتمد اساسا على DES ،كما سنرى. قدّمت DES أمنية متقدمة ، ولحد الأن فإن افضل طريقة لمهاجمتها هي مهاجمة القوة القاسية brute force وذلك بتجربة جميع احتمالات المفتاح.
- مؤخرا ، تم استبدال DES بمعيار التشفير المتقدم AES. في هذا المقطع ، سوف نستعرض كيفية عمل DES بصورة مختصرة.

3.4.1 تصميم DES

تعتمد DES على هيكل شبكة Feistel ذي 16 دورة ، حيث تشفّر الكتلة m التي طولها 64 ثنائية باستخدام مفتاح طوله 56 ثنائية لنحصل على كتلة مشفّرة c بطول 64 ثنائية ايضا.

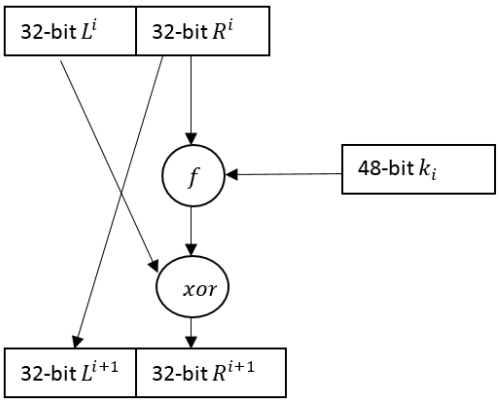


- قبل ان تمرّر الكتلة m الى شبكة Feistel ، تقوم DES ببعثرتها عن طريق البعثرة الاولية Initial permutation (IP).
- يطبّق معكوس هذه البعثرة IP^{-1} على المخرجات للحصول على الكتلة المشفّرة c.
- لاتضيف البعثرة الاولية اي اهمية أمنية لخوارزمية DES ، كما انها تعمل على ابطاء التنفيذ البرامجي لطريقة DES ، لذا عادة مايتم اهمالها عند مناقشة أمنية DES. راجع الملحق للاطلاع على جميع الجداول المستخدمة في شفرة DES. يوضح الشكل (3.1) الهيكل العام لشفرة DES.



شكل (3.1): المخطط العام لشفرة DES

- تقسم الكتلة المدخلة m الى نصفين L و R كل واحد بطول 32 ثنائية.
- تشفّر هذه الكتلة بـ16 دورة.
- تستخدم جميع الدورات نفس الدالة f ، ولكن بمفاتيح فرعية مختلفة.
- كل مفتاح فرعي طوله 48 ثنائية ويتم اشتقاقهم جميعا من المفتاح الرئيسي ذي 56 ثنائية.
- تتم عملية الاشتقاق عن طريق عملية key-schedule لانتطرق لتفاصيلها في هذا الكتاب. يوضّح الشكل (3.2) الهيكل العام لدورات DES.

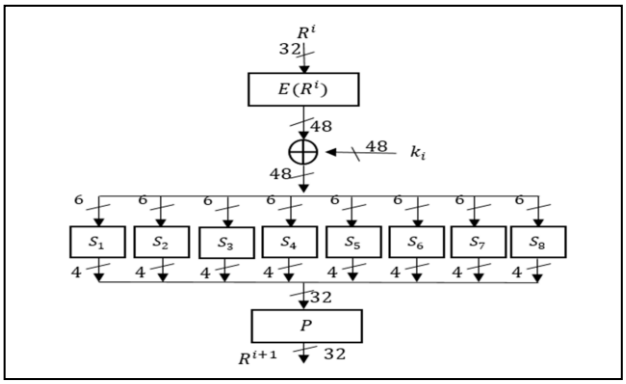


شكل (3.2): دورة DES

عمل الدالة f. تستلم الدالة f مفتاح فرعي بطول 48 ثنائية وادخال R بطول 32 ثنائية.

- تعمل الدالة اولا على توسيع الادخال R من 32 ثنائية الى 48 ثنائية عن طريق تكررا بعض الثانئيات بجدول التوسيع $E(R)$ ،
- ومن ثم تجري عملية XOR بين نتيجة التوسيع والمفتاح الفرعي$E(R)\oplus k$.
- يتم تقسيم الناتج الذي طوله 48 ثنائية الى 8 كتل بطول 6 ثنائية لكل كتلة.
- تمرّر كل كتلة على صندوق تعويض S-box خاص بها ، بحيث يستلم 6 ثنائيات ويعطي 4 ثنائيات.
- توجد هناك ثمان صناديق تعويضية $S_1, \dots, S_8$ غير قابلة للعكس (لعدم تطابق طول المدخلات مع طول المخرجات ، راجع الملحق للاطلاع على تصميم هذه الصناديق). ت
- عمل صناديق التعويض هذه على تشويش المدخلات confusion. يكون اجمالي اخراجات الصناديق الثمانية 32 ثنائية.
- الخطوة الاخيرة ، هي نشر diffusion اخراج الصناديق بمزجها معا عن طريق عملية بعثرة Pيكون اخراجها 32 ثنائية ايضا ، والذي يمثل الاخراج النهائي للدالة.

يوضّح الشكل (3.3) عمل الدالة f.



شكل (3.3): التصميم الداخلي للدالة f

صناديق التعويض S-boxes. تم اختيار قيم صناديق التعويض بعناية كي يصعب مهاجمته وخصوصا تجاه **مهاجمة التباين** differential cryptanalysis. وعلى الرغم من تأخر التوصل الى هذه المهاجمة (عام 1980) الّا ان فريق IBM كان ملتفتا الى هذا النوع من المهاجمة اثناء فترة تصميم DES.

- يتمثل كل صندوق بشكل جدول له 4 صفوف و16 عمود. تكون عناصر الجدول بشكل قيم بطول 4 ثنائيات وتمثّل الارقام الصحيحة $\{0, \dots, 15\}$. كل صف في الجدول هو بعثرة للارقام 1...15 عند استلام الخيط $B_j = b_1b_2b_3b_4b_5b_6$ فإنه يتم حساب $S_j(B_j)$له كما يلي: تمثل الثنائيات b_1b_6 الصيغة الثنائية

لرقم الصف r في $S_j()$(بحيث $r = 0, \dots, 3$)، اما الثنائيات الاربعة الاخرى $b_2b_3b_4b_5$ فتمثل الصيغة الثنائية لرقم العمود o في $S_j()$(بحيث $o = 0, \dots, 15$). وبذلك يكون اخراج الصندوق هو العنصر $S_j(r, o)$ بشكل اربع ثنائيات.

- تم اختيار قيم الجداول بحيث تضمن انه عند تغيير ثنائية واحدة في الادخال ، ستتغير على الاقل ثنائيتين في الاخراج.
- تعتبر صناديق التعويض هي الجزء الاساسي لأمنية شفرة DES وذلك لاضافتها **مبدأ اللّاخطية** non-linearity للشفرة ، بمعنى

$$S(a) \oplus S(b) \neq S(a \oplus b)$$

- بدون استخدام التشفير اللاخطي يستطيع الخصم التعبير عن مدخلات ومخرجات DES بشكل نظام خطّي ويحاول ان يحسب المفتاح المجهول.
- لغرض التوضيح ، نذكر محتوى صندوق التعويض S_1 في الشكل (3.4).

S_1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
1	00	15	07	04	14	02	13	01	10	06	12	11	09	05	03	08
2	10	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
3	15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13

شكل (.34): صندوق التعويض S_1 لخوارزمية DES

مثال(3.1): افترض ان ادخال الصندوق S_1 هو (101000). عندها تمثل 10 رقم الصف وهو 2 ، اما الثنائيات 0100 فتمثل رقم العمود وهو الرقم 4. عندها يكون اخراج الصندوق هو العنصر 13 ، والذي له الصيغة الثنائية 1101.

فك شفرة DES. تكون عملية فك الشفرة في DES هي نفس عملية التشفير وذلك لاعتمادها على شبكة Feistel. اثناء عملية فك الشفرة يتم اشتقاق المفاتيح الفرعية بصورة معكوسة ، حيث ان الدورة الأولى تستخدم المفتاح الفرعي رقم 16 ، والدورة الثانية تستخدم المفتاح الفرعي رقم 15 ، وهكذا.

3.4.2 أمنية DES

- افضل طريق لمهاجمة DES هي عن طريق تجربة جميع قيم المفتاح بما يعرف مهاجمة brute force وبما ان طول مفتاح DES هو 56 ثنائية ، فإن هذه المهاجمة تتطلب 2^{56} من المحاولات.
- اصبحت كمية المحاولات الضخمة هذه ممكنة في الوقت الحالي.
- تطلبت اول محاولة لكسر شفرة DES ومعرفة المفتاح 96 يوما عن طريق مشروع DESCHALL عام 1997.
- في عام 1998 تم كسر الشفرة بـ41 يوم بمشروع distributed.net.
- شهد نفس العام قفزة نوعية وذلك بكسر DES بوقت 56 ساعة. تم هذا الانجاز عن طريق استخدام جهاز خاص لكسر شفرة DES يعرف بـ Deep Crack والذي كلّف في حينها 250 الف دولار.
- بعد ان تشارك مشروع distributed.net مع جهاز Deep Crack تم كسر الشفرة بـ22 ساعة 15 دقيقة فقط. نتيجة لذلك فإن DES لم تعد آمنة.
- تجدر الاشارة الى ان مهاجمة brute force لم تعتمد على تنظيم DES الداخلي بل عن طريق مهاجمة مفتاحها الصغير ، لذا تم توظيف DES في تصميم منهج تشفير كتلي اكثر أمنية (يعرف بـTriple DES) وبمفتاح اطول.
- هناك محاولات لاستغلال تنظيم DES الداخلي وبالتحديد صناديق التعويض S-boxes. على سبيل المثال ، في نهاية 1980 طوّر الباحثان Biham و Shamir اسلوب مهاجمة جديد يعرف بمهاجمة التباين Differential cryptanalysis يتطلب عند تطبيقه على DES 2^{47} من الحسابات ، وقد اعترف مصممي DES انّ هذا الاسلوب من المهاجمة كان مشخّصا من قبلهم منذ البداية.
- في بداية 1990 طوّر Matsui اسلوب مهاجمة جديد يدعى **المهاجمة الخطية** Linear cryptanalysis يستطيع مهاجمة DES بـ2^{43} من الحسابات. على الرغم من ان هذه المهاجمات المعقدة تستطيع ان تكسر DES بوقت اقل من brute force الّا انها تعتبر مهاجمات نظرية لكونها تحتاج الى كميات كبيرة من ازواج النصوص الصريحة والمشفرّة (كما سنوضّح لاحقا) لاكتشاف مفتاح التشفير ، وبالتالي تبقى مهاجمة brute force هي المهاجمة الاكثر كفاءة من الناحية العملية.

3.4.3 زيادة طول المفتاح لطريقة DES

في هذا الجزء ، سوف نستعرض مناهج تشفير آمنة تعتمد على منهج DES الاساسي ولكن بمفتاح اطول.

الاستدعاء المزدوج

ليكن لدينا منهج التشفير الكتلي F وليكن k_1 و k_2 مفتاحين مستقلين. يمكن تصميم منهج تشفير كتلي جديد له مفتاح اطول بمقدار مرتين من منهج التشفير الاصلي كما يلي:

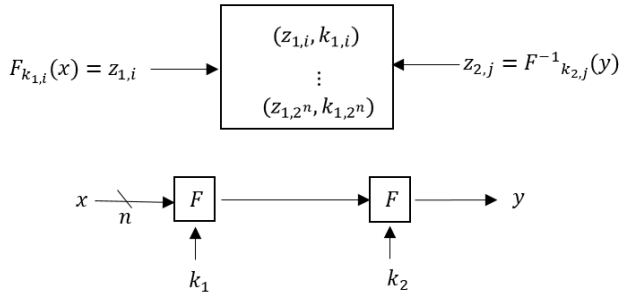
$$F'_{k_1,k_2}(x) = F_{k_2}(F_{k_1}(x)).$$

عندما تكون DES=F فإن المنهج الناتج يكون مفتاحه 112 (ويعرف بـ Double DES)، وهو اكبر بكثير من ان يتم كسر شفرته بطريقة brute force.

- لسوء الحظ، لا يوفّر الاستدعاء المزدوج للتشفير الكتلي أمنية عالية، حيث يمكن مهاجمته بطريقة مهاجمة خاصة تدعى "الالتقاء عند المنتصف" meet-in-the-middle. تتلخص فكرة هذه المهاجمة بمايلي: يعطى الخصم الزوج (x, y)، حيث ان

$$y = F'_{k_1,k_2}(x) = F_{k_2}(F_{k_1}(x))$$

يبدأ الخصم بناء قائمتين من الازواج. تتضمن القائمة الأولى جميع الازواج التي شكلها (k'_1, z_1) حيث $z_1 = F'_{k'_1}(x)$ لجميع قيم k'_1. تتضمن القائمة الثانية جميع الازواج بالشكل (k'_2, z_2) بحيث $z_2 = F^{-1}_{k'_2}(y)$. لاحظ، انه توجد قيمة z بحيث $F_{k_1}(x) = z = F^{-1}_{k_2}(y)$، وان k_1 و k_2 هي قيم المفاتيح التي يبحث عنها الخصم. يهدف الخصم الى ايجاد الازواج المتطابقة بين القائمتين والتي لها نفس قيمة z. يعتبر اي تطابق نحصل عليه مفتاح محتمل للتشفير الكتلي F'. يوضّح الشكل (3.5) كيفية هذه المهاجمة.



شكل (3.5): مهاجمة الالتقاء عند المنتصف

- عندما يكون طول المفتاح للمنهج Fهو n، فإن تشكيل القوائم لهذه المهاجمة يتطلب وقت $\mathcal{O}(2^n)$، بينما تحتاج تلك القوائم الى حيّز خزني كبير جدا $n.2^n$.

الاستدعاء الثلاثي

للتخلص من مهاجمة الالتقاء عند المنتصف، يتم استخدام طريقة الاستدعاء الثلاثي لمنهج التشفير الكتلي. توفّر هذه الطريقة مستوى أمنية عالي ضد مهاجمة brute force. هناك صيغتان للاستدعاء الثلاثي:

أ. الصيغة الأولى- ثلاث مفاتيح مختلفة: نختار ثلاث مفاتيح مختلفة k_1، k_2، k_3 ونحسب
$y = F'_{k_1,k_2,k_3}(x) = F_{k_3}(F^{-1}_{k_2}(F_{k_1}(x)))$.

ب. الصيغة الثانية- مفتاحين مختلفين: نختار مفتاحين مختلفين k_1، k_2ونحسب $y = F'_{k_1,k_2}(x) = F_{k_1}(F^{-1}_{k_2}(F_{k_1}(x)))$.

- يدعى الاستدعاء الثلاثي لتشفير DES الكتلي بـ(3DES) Triple-DES وقد حلّ محل DES بصورة رسمية منذ عام 1999.
- سلبية 3DES الاساسية هي صغر حجم الكتلة وحقيقة كونها بطيئة لأنها تتطلب ثلاث عمليات تشفير لكل كتلة، وبما ان كل عملية تشفير تتطلب 16دورة، فإن مجموع الدورات لتشفير كتلة واحدة هو 48 دورة!.
- دعت هذه السلبيات الى استبدال 3DES عام 2001 بمعيار التشفير المتقدم AES.

3.5 معيار التشفير المتقدم

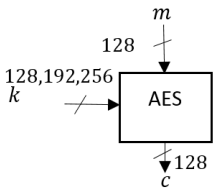
- اعلن معهد National Institute of Standards and Technology (NIST) في الولايات المتحدة عام 1999 عن رغبته في ايجاد تشفير كتلي جديد ليستبدل DES. يدعى هذا التشفير بمعيار التشفير المتقدم Advanced Encryption Standard (AES).
- تم تقديم 21 طريقة تشفير من مختلف انحاء العالم. تتضمن الطرق المتقدمة للمنافسة عمل افضل مصمّمي خوارزميات التشفير وفك الشفرة في حينها.
- بعد المؤتمر الاول الخاص بمعيار AES عام 1998، اعلنت NIST عن ترشيح 15 طريقة تشفير.
- بعد المؤتمر الثاني عام 1999، تم اختيار 5 طرق فقط وهي: MARS, RC6, Rijndael, Serpent, Twofish.

- في مؤتمر AES الثالث عام 2000 ، تم مناقشة هذه الطرق الخمس المتبقية وبعده اعلنت NIST اخيرا عن فوز **خوارزمية** Rijndael بمعيار التشفير المتقدم للتشفير الكتلي ، حيث لم يعثر افضل كاسري الشفرات في العالم على ابسط نقطة ضعف في تلك الخوارزمية رغم عملهم المركّز.

3.5.1 طريقة عمل AES

في هذا الجزء ، سوف نتكلم عن طريقة عمل AES بصورة عامة وبدون الدخول في التفاصيل. قبل التعرض لعمل AES نود الاشارة الى ان AES هو اسم معيار التشفير ويطلق مسامحة على اسم الخوارزمية ، التي في حقيقة الامر اسمها هو Rijndael.

- تعتمد AES على هيكل شبكة (التعويض- البعثرة) على العكس من DES التي تعتمد على شبكة Feistel. يكون ادخال AES مكّون من 128 ثنائية والذي هو تماما 16 بايت اما مفتاحها فيكون اما 128 ثنائية ، 192 ثنائية ، او 256 ثنائية.



- تقوم AES بتخزين هذه البيانات في مصفوفة بالابعاد 4×4 والتي تعرف **بالحالة** state.
- تجري عمليات التشويش- النشر على مصفوفة الحالة عن طريق اربع عمليات اساسية ولعدد من الدورات nr.
- يعتمد عدد الدورات على طول المفتاح المستخدم. حيث يكون عدد الدورات 10 عندما يكون طول المفتاح 128 ، 12 عندما يكون المفتاح 192 ، و 14 عندما يكون المفتاح 256.
- نستعرض الأن بصورة اجمالية كيفة عمل خوارزمية AES:

1. عند استلام كتلة الادخال m ، نجعل الحالة state هي قيمة m ، وننفذ عملية AddRoundKey بين مفتاح الدورة RoundKey و state
2. لكل دورة من $nr-1$ ، ننفذ على الحالة state العمليات التالية بصورة متتالية: SubBytes، ShiftRows ، MixColumns ، ومن ثم AddRoundKey.
3. ننفذ في الدورة الاخيرة العمليات:SubBytes ، ShiftRows ، و AddRoundKey.
4. الاخراج النهائي c هو الحالة state.

لاحظ ، انه الدورة الاخيرة من الخوارزمية لاتستخدم عملية MIxColumns.

نبيّن الأن العمليات الاربع الاساسية لخوارزمية AES:

1. عملية 1- AddRoundKey: في كل دورة من دورات AES يتم اشتقاق مفتاح فرعي RoundKey بطول 16 بايت من المفتاح الرئيسي. يخزّن هذا المفتاح الفرعي في مصفوفة 4×4. يتم تطبيق عملية XOR بين كل بايت في مصفوفة المفتاح مع البايت المناظر له في مصفوفة الحالة . ليكن $a_{i,j}$ هو البايت الموجود في الصف رقم i والعمود رقم j في مصفوفة الحالة، كذلك الحال بالنسبة للبايت $k_{I,j}$ في مصفوفة المفتاح. تقوم عملية AddRoundKey بحساب $a_{i,j} = a_{i,j} \oplus k_{i,j}, \forall i,j = 1,..,4$
2. عملية 2-SubBytes: يتم في هذه العملية استخدام صندوق تعويض واحد S-Box، مكوّن من 16 بايت مرتّبة بشكل مصفوفة 4×4 ، لتعويض كل بايت موجود في مصفوفة الحالة الى بايت اخر. يمتاز صندوق التعويض بكونه قابل للعكس ومن نوع 1-1 ، بمعنى ان كل ادخال له اخراج واحد فقط ويمكن بسهولة معرفة الادخال في حالة وجود الاخراج. تجري عملية SubBytes كما يلي:

$$a_{i,j} = S\_Box(a_{i,j}) \forall i,j = 1,..4.$$

يوضّح الشكل (3.6) صندوق التعويض لخوارزمية AES. لتعويض قيمة بايت في جدول التعويض ، تستخدم اول اربع ثنائيات من ذلك الادخال لتحدد رقم الصف ، اما الثنائيات الاربع الاخرى فتحدد رقم العمود. يكون الاخراج هو عنصر الصندوق الذي يلتقي عنده رقم الصف ورقم العمود. يوفّر صندوق التعويض S-Box خاصية اللّاخطية لخوارزمية AES ايضا كما ذكرنا في خوارزمية DES.

مثال(3.2): عندما يكون الادخال $(c2)_{hex}$ فإن الاخراج هو $(25)_{hex}$.

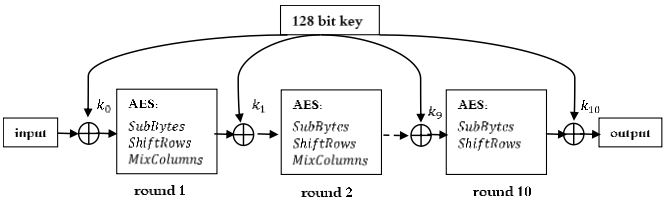
x \ y	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

شكل (3.6): صندوق التعويض S-Box لخوارزمية AES

3. عملية 3-ShiftRows: في هذه العملية يتم تدوير صفوف مصفوفة الحالة نحو اليسار كما يلي: الصف الاول يبقى بدون تدوير ، الصف الثاني يدوّر مرة واحدة ، الصف الثالث يدوّر مرّتان ، والصف الثالث يدوّر ثلاث مرات.
4. عملية 4-MixColumns: تتضمن هذه العملية مزج كل عمود عن طريق عملية تحويل خطي قابل للعكس. يتم التعبير عن كل عمود بشكل متعددة حدود معاملاتها قيم ذلك العمود.

تمثل العمليتان الاخيرة (3 و4) عملية البعثرة permutation في حين تمثل العملية رقم 1 عملية التعويض substitution من عمليات شبكة بعثرة-تعويض.

يوضّح الشكل (3.7) الهيكل العام لتشفير كتلة بيانات باستخدام AES.



شكل (3.7): مخطط شفرة AES

3.5.2 تصميم S–Box

على العكس من طريقة DES ، التي تستخدم قيم عشوائية لصناديق التعويض ، صُمّم صندوق التعويضS-box لطريقة AES بصورة جبرية ، حيث يتضمن تشكيل هذا الجدول عمليات في حقول منتهية finite fields. لتوضيح كيفية تشكيل الجدول يجب ان نقدّم بعض التعاريف.

تعريف(3.1): يعرف **الحقل** $\mathbb{F}$ بأنه مجموعة من العناصر التي تحقق مايلي:

1. جميع عناصر الحقل $\mathbb{F}$ تشكّل مجموعة جمعية additive group لعملية الجمع والعنصر المحايد 0.
2. جميع عناصر الحقل $\mathbb{F}$ باستثناء 0 تشكّل مجموعة ضربية بعملية الضرب والعنصر المحايد 1.
3. عند مزج هاتين العمليتين ، فيجب تحقق قانون التوزيع

$$\forall a, b, c \in F: a(b + c) = (ab) + (ac)$$

- نهتم في مجال التشفير بحقول محدودة العناصر تعرف بالحقول المنتهية. ويمثل عدد عناصر تلك الحقول رتبة الحقل field order.
- توجد الحقول المنتهية فقط عندما تكون رتبتها qبالشكل p^n ، بحيث p هو عدد اولي و n عدد صحيح موجب. مثال على ذلك ، الحقل $\mathbb{F}(2)$ الذي يتكون من عنصرين والحقل $\mathbb{F}(2^8)$ الذي يتكون من 256 عنصر.
- تستخدم AES الحقل $\mathbb{F}(2^8)$ حيث تمثل كل بايت من البيانات بشكل عنصر في ذلك المجال. عموما ، عدد عناصر الحقل $\mathbb{F}(2^8)$ هو عدد غير اولي (256) وبالتالي لايمكن اجراء عمليات الجمع والضرب في حالة $\mod 2^8$. لذا يتحتم استخدام صيغة اخرى لتمثيل عناصر هذا الحقل. الصيغة المستخدمة هنا هي تمثيل كل عنصر بشكل متعددة حدود polynomial ، بحيث تجري العمليات الحسابية على عناصر الحقل باجراء العمليات الحسابية لمتعددات الحدود التي تناظر تلك العناصر. يتمثل كل عنصر في الحقل $\mathbb{F}(2^8)$ بشكل متعددة حدود

$$A(x) = a_7x^7 + \cdots + a_1x + a_0, a_i \in F(2) = \{0,1\}$$

- لاحظ انه لتخزين متعددة الحدود هذه فأننا نخزّن فقط معاملاتها $(a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0)$ والتي تتطلب 8 ثنائيات فقط.
- نفترض ان لدينا الدالة Binary2Field التي تحوّل الثنائيات الى متعددة حدود ، الدالة المعاكسة Field2Binary التي تحوّل متعددة الحدود الى ثنائيات ، والدالة FieldInv التي تحسب المعكوس الضربي لكل عنصر من عناصر الحقل.

الجمع والضرب. تجري عمليات الجمع والضرب على متعددات الحدود وفق قوانين متعددات الحدود. تتم عملية الجمع بين متعددتي حدود بإجراء عملية Xor بين ثنائيات عناصر تلك المتعددات. ينتج عن عملية ضرب متعددتي الحدود فائض احيانا ، بحيث تكون درجة متعددة الحدود الناتجة اكبر من 8 ، لذا يتحتم تقليص النتيجة بتقسمها على متعددة حدود $P(x)$ معينة واخذ باقي القسمة فقط. استخدمت AES متعددة الحدود $P(x) = x^8 + x^4 + x^3 + x + 1$ لغرض تقليص عملية الضرب.

- بعد هذه المقدمة السريعة نعود الأن لبيان كيفية حساب قيم صندوق التعويض S-box لطريقة AES. توضّح الخوارزمية (3.1)كيفية حساب قيمة صندوق التعويض لأحد المدخلات.

Algorithm (3.1) SubBytes

Input: byte $(a_7a_6a_5a_4a_3a_2a_1a_0)$

Output: byte $(b_7b_6b_5b_4b_3b_2b_1b_0)$

$z = \text{Binary2Field}(a_7a_6a_5a_4a_3a_2a_1a_0)$

if $z \neq 0$:

$z = \text{FieldInv}(z)$

$(a_7a_6a_5a_4a_3a_2a_1a_0)$=Field2Binary(z)

$(c_7c_6c_5c_4c_3c_2c_1c_0) = (01100011)$

for $i = 0$ to 7:

$b_i = (a_i + a_{i+4} + a_{i+5} + a_{i+6} + a_{i+7} + c_i) \bmod 2$

return $b_7b_6b_5b_4b_3b_2b_1b_0$

نذكر الأن مثال بسيط لتوضيح عمل خوارزمية (3.1) للتعويض.

مثال(3.3): افترض ان الادخال هو 53 ، الذي يساوي 01010011 بالنظام الثنائي ، والتي تقابل متعددة الحدود $x^6 + x^4 + x + 1$. المعكوس الضربي لهذا العنصر هو $x^7 + x^6 + x^3 + x$ والذي يقابل بالنظام الثنائي

$$(a_7a_6a_5a_4a_3a_2a_1a_0) = (11001010)$$

نقوم الأن بحساب ثنائيات الاخراج:

$$b_0 = a_0 + a_4 + a_5 + a_6 + a_7 + c_0 \bmod 2$$

$$= 0 + 0 + 0 + 1 + 1 + 1 \bmod 2 = 1$$

$$b_1 = a_1 + a_5 + a_6 + a_7 + a_0 + c_1 \bmod 2$$

$$= 1 + 0 + 1 + 1 + 0 + 1 \bmod 2 = 0$$

وهكذا. تكون النتيجة هي 11101101 ، وهي عدد ED بالنظام السداسي عشر. ويمكن التحقق من صحة النتيجة باستخدام الجدول في شكل (3.4).

3.5.3 خوارزمية توسيع المفتاح

- تعمل خوارزمية **توسيع المفتاح** key expansion algorithm على استلام المفتاح (بطول 128 ، 192 ، او 256) وتشتق منها مفاتيح فرعية.
- لاحظ انه يتم استخدام مفتاح فرعي لعملية AddRoundKey قبل بدء الدورات وبذلك فإن عدد المفاتيح المطلوبه هي عدد الدورات زائد واحد.
- نوضّح الأن كيفية توسيع مفتاح بطول 128 ثنائية.
- تستلم AES مفتاح بطول 16بايت وتعمل على توسيع هذا المفتاح الى 176 بايت ، مما يكفي لتزويد عملية AddRoundKey الابتدائية بـ16بايت وتزويد الدورات العشر الاخرى بواقع 16 بايت لكل دورة ايضا.
- تعمل خوارزمية التوسيع على اشتقاق المفتاح الفرعية وترتيب الاخراجات بصورة 44 كلمة $w[0], \dots, w[43]$ ،كل كلمة بطول 4 بايت.
- تتضمن خوارزمية توسيع المفتاح عمليتين: RotWord ، و SubWord. تقوم عملية $\text{RotWord}(B_0, B_1, B_2, B_3)$ بالتدوير الدائري للبايتات الاربع. بمعنى

$$\text{RotWord}(B_0, B_1, B_2, B_3) = (B_1, B_2, B_3, B_0)$$

- اما عملية $\text{SubWord}(B_0, B_1, B_2, B_3)$ فتطبّق صندوق S-box الخاص بـAES على كل بايت B_0, B_1, B_2, B_3 على حدة. بمعنى:

$$\text{SubWord}(B_0, B_1, B_2, B_3) = (B'_0, B'_1, B'_2, B'_3)$$

بحيث $B'_i = S_{\text{Box}(B_i)}: i = 0, \dots, 3$. RCon مصفوفة 10 عناصر قيمها ثوابت بالنظام السداسي عشر.

توضح الخوارزمية (3.2) خوارزمية توسيع المفتاح.

Algorithm (3.2):Key Expansion

Input: 16-byte key

Output: 44 words w

$RCon[1] = 01000000, RCon[2] = 02000000,$

$RCon[3] = 04000000, RCon[4] = 08000000,$

$RCon[5] = 10000000, RCon[6] = 20000000,$

$RCon[7] = 40000000, RCon[8] = 80000000,$

$RCon[9] = 1B000000, RCon[10] = 3600000$

for $i = 0$ to 3:

$w[i] = (key(4i), key(4i + 1), key(4i + 2), key(4i + 3))$

for $i = 4$ to 43:

$temp = w[i - 1]$

If $i = 0 \bmod 4$

$temp = SubWord(RotWord(temp)) \oplus Rcon[i. 4]$

$w[i] = w[i - 4] \oplus temp$

return $(w[0], \dots, w[43])$

- يمكن تنفيذ خوارزمية توسيع المفتاح لاشتقاق جميع المفاتيح بصورة مسبقة قبل اجراء عمليات التشفير (فك التشفير). يتطلب هذا الاسلوب وجود خزن كافي للمفاتيح لجميع المفاتيح الفرعية مما يجعله غير ملائم للتطبيق في حالة الاجهزة محدودة الموارد كالبطاقات الذكية.
- هناك اسلوب اخر يتضمن اشتقاق المفاتيح الفرعية عند الحاجة ولكل دورة من دورات التشفير (فك التشفير). لاحظ ان عملية فك التشفير تتطلب العمل بتسلسل معكوس للمفاتيح مما يعني انه يتحتم اشتقاق المفاتيح بصورة معكوسة من البداية ولحين الوصول للدورة المطلوبة ، وبالتالي نلاحظ ان عملية فك الشفرة تكون ابطأ بقليل من عملية التشفير ضمن هذا الاسلوب.

3.5.4 فك شفرة AES

ذكرنا ان AES لاتعتمد على شبكة Feistel لذا يجب توفر جميع عمليتها بصورة معكوسة اثناء فك الشفرة. حيث توجد InvSubBytes بدلا من SubBytes، InvShiftRows بدلا من ShiftRows ، و InvMixColums بدلا من MixColums. كما ان المفاتيح الفرعية يتم اشتقاقها وتقديمها بصورة معكوسة ، المفتاح الاخير يستخدم بالدورة الأولى من عملية فك الشفرة ، الى نصل الى الدورة الاخيرة التي تستخدم المفتاح الاول.

3.5.5 أمنية AES

ذكرنا ان خوارزمية AES تم اختيارها نتيجة لصمودها تجاه كل انواع المهاجمات ويعود السبب في ذلك الى تضمين تصميمها على صفات خاصة ساعدت في توفير الأمنية ضد مختلف المهاجمات. على سبيل المثال ، تم تصميم صندوق التعويض بطريقة تقاوم هجمات Differential cryptanalysis و Linear cryptanalysis التي تعرضت لها خوارزمية DES. كذلك ساعد التحويل الخطي MixColumns على استحالة اجراء المهاجمات الخطية.

3.6 المهاجمات الخطية والتباينية

في هذا المقطع ، سنقدم تقنيتي مهاجمة قويتين: مهاجمة التباين و المهاجمة الخطية.

مهاجمة التباين Differential cryptanalysis . قُدّمت هذه التقنية لاول مرة من قبل Biham و Shamir عام 1980 حيث استعملاها لمهاجمة DES. ليكن x_1 و x_2 هما مدخلان لمنهج التشفير. يعرف الفرق بينهما بالشكل $\Delta_x = x_1 \oplus x_2$. كذلك ، ليكن y_1 و y_2 هما اخراجا منهج التشفير عند تشفير الادخالين x_1 و x_2 ، على التوالي ، بنفس المفتاح. ليكن $\Delta_y = y_1 \oplus y_2$. يعّرف الزوج (Δ_x, Δ_y)بالتباين.

- تهتم هذه التقنية بايجاد نقاط الضعف في منهج التشفير التي تنتج تباينات معينة باحتمالية اعلى مما هو متوقع في الدوال العشوائية.
- لفرض التوضيح ، نقول ان التباين (Δ_x, Δ_y) يظهر باحتمالية p اذا كان للمدخلان العشوائيان x_1 و x_2 القيمة $\Delta_x = x_1 \oplus x_2$ فإن احتمالية ظهور $\Delta_y = y_1 \oplus y_2$ هي p. عندما يكون طول كتلة الادخال هو n فيجب ان لايظهر اي تباين باحتمالية تزيد عن 2^{-n}. مع ذلك ، تظهر بعض التباينات باحتمالية اكبر.

- يعمل الخصم في هذ التقنية باجراء مهاجمة من نوع CPA ، حيث يختار مدخلات لها نفس التباين ومن ثم يطلب تشفيرها. ثم يقوم بعد ذلك بحزر ثنائيات المفتاح.
- تعتبر هذه التقنية من المهاجمات النظرية كونها تتطلب عدد كبير من النصوص الصريحة المختارة (2^{47} لمهاجمة DES).

المهاجمة الخطية Linear cryptanalysis. طورّت المهاجمة الخطية من قبل Matsui بداية 1990. تعتمد هذه المهاجمة على افتراض وجود علاقة خطيّة بين بعض ثنائيات الادخال وثنائيات والاخراج. بمعنى ان ، الضعف المحتمل هو وجود ترابط خطّي بين ثنائيات الادخال والاخراج.

- لتكن $x_1, \dots, x_n$ هي ثنائيات الادخال ، ولتكن $y_1, \dots, y_n$ هي ثنائيات الاخراج ، يعمل التحليل الخطي على ايجاد معادلات بالشكل $x_{i_1} \oplus \dots x_{i_l} \oplus y_{i'_1} \oplus \dots \oplus y_{i'_1} = 0$. من الواضح ، انه اذا كانت دالة التشفير عشوائية فإن نسبة تحقق المعدلات اعلاه هي فقط $1/2$. لذا فإن الخصم يتعامل مع المعادلات التي تتحقق اكثر من $1/2$.
- يتم استخدام هذه التقنية لإيجاد ثنائيات المفتاح. حيث يقوم الخصم باختيار عدد كبير من النصوص و يشفّرها ويطبّق عليها المعادلات اعلاه. اذا كانت اكثر التجارب تعطي 0 فإن قيم المفتاح الموجودة في المعادلات تأخذ اصفار ، واذا كانت اكثر التجارب تعطي 1 ، فإن قيم ثنائيات المفتاح الموجودة في المعادلات تأخذ احاد.

ملحق شفرة DES

تتضمن شفرة DES استخدام العديد من الجداول. نذكر الأن هذه الجداول:

- جدول البعثرة 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

- جدول البعثرة العكسية IP^{-1}

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

- جدول التوسيع E

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

- جدول البعثرة P

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

- صناديق التعويض:

S1

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

S2

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

S3

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

S4

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

S5

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

S6

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

S7

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

S8

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