2

الفصل الثاني

التشفير المتماثل وسرية الرسائل

محتويات الفصل:

طوال فترة الظهيرة كان مونغو يعمل على شفرة شتيرن (Stern's code)، مستعيناً بدرجة رئيسة بآخر الرسائل التي قام بنسخها في موقع التنصت بميدان نيفين. وكان شتيرن واثقاً جداً. ولابد أنه كان على درايةٍ تامةٍ بأن محطة لندن المركزية على علم بوجود موقع التنصت هذا. وكان واضحاً - من فرط ثقتهم في عدم إمكانية اختراق شفرتهم- أنهم لم يكن يهمهم كم مرة يقوم مونغو بقراءة رسائلهم.

— حديث مع رجال غرباء، روث رينديل

بين قبائل وسط أستراليا يحمل كل رجل وامرأة وطفل اسماً سرياً مقدساً يخلعه عليه شيوخ القبيلة بعد ولادته بفترة وجيزة. ويكون الاسم معروفاً فقط لدى المؤتمنين من أفراد المجموعة ولا يُذكر هذا الاسم السري على الإطلاق إلا في أكثر المناسبات رسمية؛ ويُعد النطق به على مسمع من مجموعة أخرى بمثابة خرقٍ خطير لعرف القبيلة. وإذا حدث وذكر هذا الاسم، فلا يُذكر إلا همساً، وبعد أخذ كل الاحتياطات والتدابير لتجنب سماع الاسم من قِبَل أفراد من خارج المجموعة. يعتقد الفرد في تلك القبائل أن معرفة الغرباء لاسمه السري يمنحهم قدرات خاصة تمكّنهم من إيذائه بفعل السحر.

— الغصن الذهبي، السيد جيمس جورج فريزر

كان التشفير المتماثل، والذي يُعرف أيضاً بالتشفير التقليدي، أو التشفير بالمفتاح السري، أو التشفير بمفتاح واحد، النوع الوحيد من التشفير المستعمل قبل تطوير أنظمة التشفير بالمفاتيح العامة في أواخر السبعينيات.[1] ولا يزال هذا النوع بدرجةٍ كبيرةٍ أكثر نوعي التشفير استخداماً الآن.

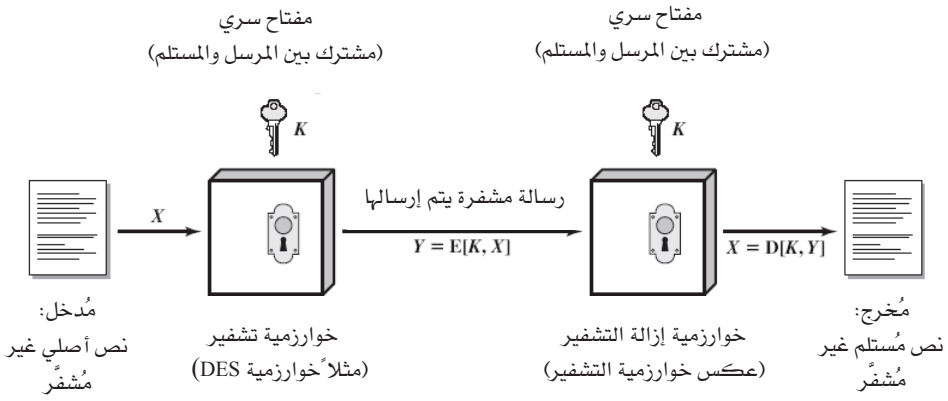
يبدأ هذا الفصل بإلقاء الضوء على نموذج عام لعملية التشفير المتماثل، مما سيمكننا من فهم السياق الذي تُستخدم فيه خوارزميات هذا النوع من التشفير. ونتناول بعد ذلك ثلاث خوارزميات مهمة لتشفير رزم (حزم) البيانات: خوارزمية DES، وخوارزمية DES الثلاثية، وخوارزمية AES. ويقدم الفصل بعد ذلك التشفير المتماثل لتدفق البيانات ويستعرض خوارزمية RC4 المستخدمة على نطاق واسع في هذا المجال. وبعدها نقوم بدراسة استخدام تلك الخوارزميات لتحقيق السرية المنشودة.

[1] ظهر وصف التشفير بالمفاتيح العامة لأول مرة في الأدبيات المفتوحة عام 1976، وتدّعي وكالة الأمن القومي الأمريكي (NSA) أنها اكتشفته قبل ذلك بعدة سنوات.

1-2 مبادئ التشفير المتماثل

يتألف نظام التشفير المتماثل من خمسة عناصر (الشكل 2-1):

- الرسالة الأصلية (غير المشفَّرة): وهي الرسالة أو البيانات الأصلية التي يتم إدخالها إلى خوارزمية التشفير كمُدخل.
- خوارزمية التشفير: تقوم خوارزمية التشفير بإدخال عدة تبديلات وتحويلات مختلفة على الرسالة الأصلية.
- المفتاح السري: المفتاح السري هو مُدخل آخر لخوارزمية التشفير.ويحدد هذا المفتاح التبديلات والتحويلات المختلفة التي تقوم الخوارزمية بإدخالها على الرسالة الأصلية.
- الرسالة المشفَّرة: وهي الرسالة المُحوّرة الناتجة كمُخرج من خوازمية التشفير. وتعتمد تلك الرسالة على الرسالة الأصلية (غير المشفَّرة) وعلى المفتاح السري. ويؤدي استخدام مفتاحين سريين مختلفين لتشفير نفس الرسالة الأصلية إلى إنتاج رسالتين مشفرتين مختلفتين.
- خوارزمية إزالة التشفير: وهي أساساً نفس خوارزمية التشفير ولكن مع تشغيلها بالعكس حيث يتم إدخال كلٍّ من الرسالة المشفَّرة والمفتاح السري إليها لتنتج الرسالة الأصلية.



الشكل 2-1: نموذج مبسط للتشفير المتماثل.

ولضمان تحقيق أمن البيانات باستخدام التشفير المتماثل هناك مطلبان أساسيان:

1. وجود خوارزمية تشفير قوية. كحد أدنى لا ينبغي للخَصْم الذي يعرف الخوارزمية وواحدة أو أكثر من الرسائل المشفَّرة بها أن يتمكن من فك تشفير الرسائل أو استنتاج المفتاح المستخدم في تشفيرها. وبعبارةٍ أخرى: مع معرفته بالخوارزمية المستخدَمة لا ينبغي أن يكون الخَصْم قادراً على فك الشفرة أو استنتاج المفتاح حتى لو كان بحوزته عدد من الرسائل المشفَّرة والرسائل الأصلية المقابلة لها.
2. ينبغي أن يكون كلٌّ من المرسل والمُستلِم قد تلقيا نسخةً من المفتاح السري بطريقةٍ مأمونة، وأن يحتفظا به بطريقةٍ آمنة. حيث إنه إذا استطاع شخصٌ ما معرفة المفتاح مع معرفته بالخوارزمية المستخدَمة، فسيمكنه قراءة أي اتصالات مشفَّرة تتم باستخدام ذلك المفتاح.

ومن المهم الإشارة إلى أن أمن التشفير المتماثل يعتمد على سرية المفتاح، وليس على سرية الخوارزمية. بمعنى أننا نفترض أنه سيكون من غير العملي فك شفرة رسالة بناءً على الرسالة المشفَّرة ومعرفة خوارزمية التشفير/إزالة التشفير. وبعبارةٍ أخرى، لا توجد حاجةٌ للحفاظ على سرية الخوارزمية؛ بل لابد فقط من الحفاظ على سرية المفتاح.

إن هذه الميزة للتشفير المتماثل تجعله صالحاً للاستخدام على نطاق واسع. فعدم الحاجة لسرية الخوارزمية مكّن المنتجين من تطوير شرائح تشفير إلكترونية (chips) منخفضة الكلفة. وهذه الشرائح متاحة حالياً على نطاق واسع ويتم استخدامها في عدد من المنتجات. ولذلك فإن الحفاظ على أمن المفتاح وسريته يمثل المشكلة الرئيسة عند استخدام التشفير المتماثل.

2-1-1 التشفير

عادةً ما تُصنَّف نظم التشفير بناءً على ثلاثة عوامل مستقلة هي:

1. نوع العمليات المُستخدَمة لتحويل الرسالة الأصلية إلى رسالةٍ مشفَّرة: تعتمد جميع خوارزميات التشفير على مبدأين أساسيين:
   - التعويض: حيث يُستبدل كل عنصر من عناصر الرسالة الأصلية (بت، أو حرف، أو مجموعة من البتات أو مجموعة من الحروف) بعنصر آخر.
   - النقل: حيث يُعاد ترتيب عناصر الرسالة الأصلية.

   وفي كلتا الحالتين يُشترط عدم فقدان أية معلومة متضمنة بالرسالة الأصلية (أي أن جميع العمليات يمكن عكسها). وتشتمل معظم نظم التشفير المستخدَمة، والتي تُعرف بالنظم الضربيّة (product systems) على عدة مراحل من التعويض والنقل.
2. عدد المفاتيح المستخدمة: إذا كان كلٌّ من المرسل والمستلِم يستخدم نفس المفتاح، يُعرف مثل هذا النظام بنظام التشفير المتماثل، أو أحادي المفتاح، أو ذي المفتاح السري، أو التقليدي. إذا كان المرسل والمستلم يستخدمان مفتاحين مختلفين، فإن النظام يُعرف بنظام التشفير غير المتماثل، أو ذي المفتاحين، أو نظام التشفير بالمفاتيح العامة.
3. الطريقة التي تتم بها معالجة الرسالة الأصلية: هناك نظم تشفير الكتلة (block cipher) حيث تُقسَّم الرسالة الأصلية إلى كتل ويتم تشفير كل كتلة منها على حِدَة لإنتاج الكتلة المناظرة من الرسالة المشفَّرة. أما في نظم تشفير التدفق (stream cipher) فتتم معالجة عناصر البيانات الداخلة في الرسالة الأصلية عنصراً عنصراً، بحيث تُنتج لكل عنصر من العناصر الداخلة نظيره من الرسالة المشفَّرة بشكل مستمر.

2-1-2 تحليل الشفرة

يُطلق على عملية محاولة اكتشاف الرسالة الأصلية أو مفتاح التشفير "تحليل الشفرة". وتعتمد الاستراتيجية التي يتبعها محلل الشفرة على طبيعة نظام التشفير ونوعية المعلومات المتوفرة لديه. ويلخص الجدول 2-1 الأنواع المختلفة من هجمات تحليل الشفرة على أساس كمية المعلومات المعروفة لدى محلل الشفرة. أصعب

مشكلة يمكن مواجهتها ستكون عند توفر الرسالة المُشفَّرة فقط. في بعض الأحيان حتى خوارزمية التشفير قد لاتكون معروفة، ولكن بشكلٍ عام يمكننا افتراض معرفة الخَصْم للخوارزمية المُستخدَمة للتشفير. أحد أنواع الهجوم المحتملة في ظل تلك الظروف هو الهجوم الاستقصائي (brute force) حيث يتم تجربة كل المفاتيح الممكنة. إذا كان عدد المفاتيح الممكنة كبيراً جداً، فإن هذه الطريقة تصبح غير عملية. وعليه يتعين على الخَصْم الاعتماد على تحليل الرسائل المشفَّرة نفسها، وعادةً بإخضاعها لعدد من الاختبارات الإحصائية. لاستخدام هذه الطريقة، ينبغي أن تتوافر للخَصْم فكرة عامة عن نوع الرسالة الأصلية، مثلاً هل هي نص بالإنجليزية أو الفرنسية، أو ملف تنفيذي (EXE file)، أو سرد لسطور برنامج مصدري بلغة جافا، أو ملف محاسبة، وهكذا. ويُعدُّ الهجوم المبني على معرفة الرسالة المشفَّرة فقط أسهل أنواع الهجوم من حيث الدفاع ضده، نظراً لأن الخَصْم يتوافر لديه أقل قدر من المعلومات يمكنه استغلالها. غير أنه في كثير من الحالات يتوافر لدى محللي الشفرات معلومات أكثر من ذلك. فقد يتمكن المحلل من الحصول على واحدة أو أكثر من الرسائل الأصلية غير المشفَّرة مع ما يقابلها من الرسائل المشفَّرة. أو قد يعرف المحلل أن أنماطاً بعينها من النصوص ستوجد بالرسالة الأصلية. على سبيل المثال، وتبدأ الملفات المكوّدة بصيغة Postscript دائماً بنفس النمط، كما أن رسائل التحويل الإلكتروني للأموال قد تتضمن في بدايتها شعاراً قياسياً موحداً، وهكذا. كل هذه أمثلة لنصوص غير مشفَّرة معروفة (known plaintext). وعن طريق هذه المعرفة قد يتمكن المحلل من استنتاج المفتاح بناءً على الطريقة التي تم بها تحويل ذلك النص الأصلي في الرسالة المشفَّرة.

الجدول 2-1: أنواع الهجوم على الرسائل المشفَّرة.

نوع الهجوم	المعروف لدى محلل الشفرة
النص المشفَّر	– خوارزمية التشفير – النص المُشفَّر المطلوب فك تشفيره
النص الأصلي المعروف	– خوارزمية التشفير – النص المشفَّر المطلوب فك تشفيره – زوج أو أكثر من النص الأصلي والنص المشفَّر المقابل والناتج عن استخدام المفتاح السري
النص الأصلي المختار	– خوارزمية التشفير – النص المشفَّر المطلوب فك تشفيره – رسالة أصلية يختارها محلل الشفرات، مع النص المشفَّر المقابل والناتج من تشفيرها باستخدام المفتاح السري
النص المشفَّر المختار	– خوارزمية التشفير – النص المشفَّر المطلوب فك تشفيره – رسالة يُفترض أنها مشفَّرة يختارها محلل الشفرات، مع النص العادي المقابل والناتج من إزالة تشفيرها باستخدام المفتاح السري
النص المختار	– خوارزمية التشفير – النص المشفَّر المطلوب فك تشفيره – رسالة أصلية يختارها محلل الشفرات، مع النص المشفَّر المقابل والناتج من تشفيرها باستخدام المفتاح السري – رسالة يُفترض أنها مشفَّرة يختارها محلل الشفرات، مع النص العادي المقابل الناتج من إزالة تشفيرها باستخدام المفتاح السري

من أنواع الهجوم وثيقة الصلة بهجوم النص الأصلي المعروف الهجوم الذي يمكن تسميته "هجوم الكلمة المحتملة" (probable-word attack). إذا كان الخَصْم يتعامل مع تشفير رسالة نثرية عامة فقد لا يعرف إلا القليل عن محتوى الرسالة. أما إذا كان يستهدف معلومات محددة للغاية، فقد تصبح أجزاء من الرسالة معروفة. فمثلاً عند إرسال ملف محاسبة كامل قد يستطيع الخَصْم تحديد مواضع بعض الكلمات الأساسية في عناوين الأعمدة. وكمثال آخر، قد يتضمن الكود المصدري

لبرنامج من إنتاج شركة معينة بياناً عن حقوق التأليف والنشر في موضع محدد بشكل قياسي.

وإذا استطاع محلل الشفرات بشكلٍ أو بآخر جعل نظام المصدر يُدخِل في النظام رسالة من اختيار المحلل فإننا نحصل على هجوم نص أصلي مختار (chosen-plaintext attack). وبصفةٍ عامة إذا كان المحلل قادراً بطريقة ما على اختيار الرسائل التي يقوم النظام بتشفيرها فقد يتعمد المحلل اختيار أنماط البيانات التي يمكن أن تفيده في الكشف عن بنية المفتاح.

تتضمن القائمة في الجدول 2-1 نوعين آخرين من الهجوم: هجوم نص مشفَّر مختار (chosen-ciphertext attack) وهجوم النص المختار (chosen-text attack). ورغم أن هذين النوعين أقل استخداماً كأسلوبين لتحليل الشفرة، فإنهما يمثّلان سبلاً ممكنة للهجوم.

فقط خوارزميات التشفير الضعيفة نسبياً هي التي لا تصمد أمام هجوم نص مشفَّر مختار. بصفةٍ عامة، تصمم خوارزميات التشفير بحيث تصمد أمام هجمات النصوص الأصلية المعروفة.

يُعدُّ نظام التشفير آمناً حسابياً (computationally secure) إذا تحقق في النص المشفَّر الذي ينتجه النظام أحد الشرطين الآتيين أو كلاهما:

- تكلفة فك الشفرة تتجاوز قيمة المعلومات المشفَّرة.
- الوقت اللازم لفك الشفرة يتجاوز العمر المفيد للمعلومات المشفَّرة.

وللأسف من الصعب جداً تقدير حجم الجهد اللازم للنجاح في تحليل شفرة نص مشفَّر. ولكن بافتراض عدم وجود نقاط ضعف أصيلة في رياضيات الخوارزمية، فيمكننا اللجوء إلى أسلوب الحل الاستقصائي (brute-force approach) حيث يمكن حساب تقديرات معقولة للتكلفة والوقت اللازمين.

ويتلخص أسلوب الحل الاستقصائي في تجربة كل مفتاح ممكن إلى أن يتم التوصل إلى ترجمة مفهومة للرسالة المشفَّرة والتي تمثل الرسالة الأصلية . وفي

المتوسط ينبغي تجربة نصف العدد الكلي للمفاتيح الممكنة لتحقيق النجاح المنشود. يبين الجدول 2-2 الوقت اللازم لتلك العملية لأطوال مختلفة للمفاتيح، ويلاحظ أن خوارزمية DES تستخدم مفتاحاً بطول 56 بتاً. وتفترض النتائج المدرجة بالجدول أن الزمن المطلوب لفك الشفرة باستخدام مفتاح معين هو 1 ميكروثانية، وهو تقدير معقول بالنسبة لقدرة الحاسبات العادية المتوفرة حالياً. أما في حال استخدام أنظمة تضم عدداً كبيراً من المعالجات الدقيقة تعمل على التوازي (massively parallel organizations of microprocessors)، فيمكن تحقيق معدلات معالجة أعلى من ذلك بكثير. ويبيّن العمود الأخير من الجدول 2-2 النتائج في حالة نظام يمكنه معالجة مليون مفتاح كل ميكروثانية. وتوضح هذه النتائج أنه بمثل هذا المستوى من الأداء فلن يكون بالإمكان اعتبار خوارزمية DES آمنةً حسابياً.

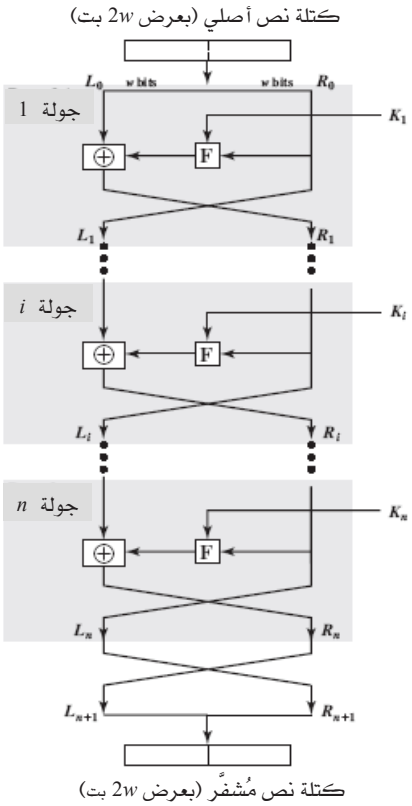
الجدول 2-2: متوسط الوقت اللازم للتوصل للمفتاح السري بأسلوب البحث الاستقصائي.

طول المفتاح	عدد المفاتيح الممكنة	الوقت اللازم بافتراض عملية تشفير كل 1 ميكروثانية	الوقت اللازم، بافتراض 10^6 عملية تشفير كل 1 ميكروثانية
32 بتاً	$2^{32} = 4.3 \times 10^9$	2^{31} ميكروثانية = 35.8 دقيقة	2.15 ميللي ثانية
56 بتاً	$2^{56} = 7.2 \times 10^{16}$	2^{55} ميكروثانية = 1142 عاماً	10.01 عاماً
128 بتاً	$2^{128} = 3.4 \times 10^{38}$	2^{127} ميكروثانية = 5.4×10^{24} عاماً	5.4×10^{18} عاماً
168 بتاً	$2^{168} = 3.7 \times 10^{50}$	2^{167} ميكروثانية = 5.9×10^{36} عاماً	5.9×10^{30} عاماً
26 حرفاً (تباديل)	$26! = 4 \times 10^{26}$	2×10^{26} ميكروثانية = 6.4×10^{12} عاماً	6.4×10^{6} عاماً

2-1-3 هيكل فيستل لخوارزميات التشفير

لكثير من خوارزميات التشفير المتماثل لكتل البيانات، بما في ذلك خوارزمية DES، هيكلٌ مشترك قام بوصفه لأول مرة هورست فيستل من شركة IBM في عام 1973 [FEIS73] ويبينه الشكل 2-2. تضم مدخلات خوارزمية التشفير كتلة بيانات من الرسالة الأصلية بطول $2w$ بت ومفتاح K، وتقسم كتلة الرسالة الأصلية

إلى نصفين: أيسر L_0 وأيمن R_0، ويمر هذان النصفان من البيانات عبر عددٍ n من جولات المعالجة، ثم يتم الجمع بينهما لتكوين الكتلة المشفَّرة. يدخل إلى كل جولة معالجة i مُدخلان أيسر L_{i-1} وأيمن R_{i-1} الناتجين من الجولة السابقة، وكذلك مفتاح فرعي K_i مستمد من المفتاح الكلي K. بشكلٍ عام، تختلف المفاتيح الفرعية K_i عن المفتاح الكلي وعن بعضها بعضاً، ويتم توليدها من المفتاح الكلي بواسطة خوارزمية فرعية لتوليد المفاتيح.



الشكل 2-2: شبكة فيستل التقليدية.

كل جولات المعالجة لها نفس الهيكل، حيث تجري عملية تعويض على النصف الأيسر من البيانات ويتم ذلك بإعمال الدالة F للتقريب (rounding) على النصف الأيمن من البيانات ثم إعمال وظيفة "أو-الحصرية" ((Exclusive-OR (XOR) على ناتج دالة التقريب مع النصف الأيسر من البيانات. ولدالة التقريب نفس الهيكل العام في كل الجولات، ولكن في كل جولة يتم تعديل متغيراتها حسب مفتاح التقريب الفرعي K_i. وبعد عملية التعويض تلك تُجرى عملية تبديل ما بين نصفي البيانات (بإحلال النصف الأيمن محل الأيسر والعكس بالعكس).

يُعدُّ هيكل فيستل أحد الأمثلة على هيكل عام تستخدمه جميع أنظمة التشفير المتمائل للكتل. وبشكلٍ عام، يتألف التشفير من سلسلة من الجولات تتم في كل جولة منها عمليات تعويض وتبديل تعتمد على المفتاح السري. وتعتمد تفاصيل خوارزمية معيَّنة للتشفير المتماثل للكتل على اختيار المتغيرات وسمات التصميم الآتية:

- مقاس الكتلة: توفِّر الكتلة الكبيرة مزيداً من الأمن (بافتراض تساوي العوامل الأخرى) ولكنها تقلل من سرعة التشفير وإزالته. تحقق الكتلة التي حجمها 128 بتاً توازناً معقولاً بين هذين العاملين وتُستخدم في معظم التصاميم الحديثة لتشفير الكتل تقريباً.
- طول المفتاح: توفِّر المفاتيح الطويلة المزيد من الأمن ولكن ذلك يمكن أن يقلل من سرعة التشفير وإزالته. والطول الأكثر شيوعاً في خوارزميات التشفير الحديثة هو 128 بتاً.
- عدد الجولات: من أساسيات التشفير المتماثل للكتل أن جولة واحدة لا تكفي لتوفير الأمن المطلوب، ولكن جولات متعددة توفِّر أمناً أكبر. والعدد المستخدم عادةً هو 16 جولة.
- الخوارزمية الفرعية لتوليد المفاتيح: تؤدي زيادة التعقيد في تلك الخوارزمية إلى زيادة صعوبة عملية تحليل الشفرة.
- دالة التقريب: مرة أخرى، تؤدي زيادة التعقيد هنا إلى المزيد من القدرة على مقاومة عمليات تحليل الشفرة.

بالإضافة لذلك، هناك الاعتباران الآتيان فيما يتعلق بتصميم أنظمة التشفير المتماثل للكتل:

- التشفير وإزالة التشفير السريع بالبرمجيات: في كثيرٍ من الأحيان يتم دمج التشفير ضمن التطبيقات أو وظائف الخدمة البرمجية مما لا يُحتاج معه إلى استخدام وحدات عتاد بنائية خاصة بذلك (hardware). وفي هذه الحالة لابد من أخذ سرعة تشغيل تلك البرامج كأحد الاعتبارات المهمة.
- سهولة التحليل: رغم أننا نود جعل خوارزميتنا أصعب مايمكن كي تستعصي على محاولات تحليل الشفرة، فإن هناك فائدة كبيرة في جعل الخوارزمية سهلة التحليل. فعندما يمكننا وصف الخوارزمية بإيجاز وشرحها بوضوح فإنه يسهل علينا تحليل الخوارزمية فيما يتعلق بنقاط ضعفها إزاء محاولات تحليل الشفرة، ومن ثم التوصل إلى مستوى أعلى من الضمان لقوتها. وجدير بالذكر أن خوارزمية DES، على سبيل المثال، لا تتصف بسهولة تحليل وظائفها.

عملية إزالة التشفير في التشفير المتماثل للكتل هي أساساً نفس عملية التشفير. والقاعدة هنا على النحو الآتي: استخدم الرسالة المشفَّرة كمُدخل للخوارزمية، ولكن مع استعمال المفاتيح الفرعية K_i بترتيب عكسي، بمعنى استعمل المفتاح K_n في الجولة الأولى وK_{n-1} في الجولة الثانية، وهكذا إلى أن تستعمل K_1 في الجولة الأخيرة. تلك ميزة لطيفة، حيث إننا لا نحتاج لتنفيذ خوارزميتين مختلفتين – واحدة للتشفير وأخرى لإزالته.

2-2 خوارزميات التشفير المتماثل للكتل

تُعدُّ خوارزميات تشفير الكتل أكثر أنواع خوارزميات التشفير المتماثل شيوعاً. تقوم عملية تشفير الكتل بمعالجة بيانات النص الأصلي على شكل كتل ذات مقاس ثابت وتنتج كتلة واحدة من النص المشفَّر (بنفس الحجم) لكل كتلة من كتل النص الأصلي. ويركّز هذا الجزء على أهم ثلاث خوارزميات للتشفير المتماثل

للكتل، وهي: معيار تشفير البيانات (DES)، ومعيار DES الثلاثي (3DES)، ومعيار التشفير المتقدم (AES).

2-2-1 معيار تشفير البيانات

تأسس نظام التشفير الأكثر استخداماً في العالم على معيار تشفير البيانات (DES) المعتمد عام 1977 من قِبل المكتب الوطني الأمريكي للمعايير (National Bureau of Standards) – وهو الآن المعهد الوطني الأمريكي للمعايير والتقنية (NIST) – كمعيار فيدرالي لمعالجة المعلومات رقم 46 (FIPS PUB 46). ويُطلق على الخوارزمية نفسها خوارزمية تشفير البيانات (DEA).[2]

❖ وصف الخوارزمية:

يبلغ طول الرسالة الأصلية 64 بتاً بينما يبلغ طول المفتاح 56 بتاً. أما الرسائل الأصلية الأطول من ذلك فيتم معالجتها على شكل كتل حجم كلٍّ منها 64 بتاً. يشبه هيكل خوارزمية DES هيكل شبكة فيستل المبين في الشكل 2-2 مع بعض الاختلافات البسيطة. وتستخدم خوارزمية DES عدد 16 جولة من جولات المعالجة. من المفتاح الأصلي (بطول 56 بتاً) يتم توليد 16 مفتاحاً فرعياً يُستخدم واحدٌ منها لكل جولة معالجة.

عملية إزالة التشفير بخوارزمية DES هي أساساً نفس عملية التشفير. القاعدة هنا على النحو الآتي: استخدم الرسالة المشفَّرة كمُدخل للخوارزمية، ولكن مع استعمال المفاتيح الفرعية K_i بترتيب عكسي، بمعنى استعمل المفتاح K_{16} في الجولة

---

[2] المصطلحات هنا مربكة بعض الشيء. فحتى وقتٍ قريبٍ كان من الممكن استخدام المصطلحين DES وDEA بنفس المعنى. غير أن أحدث طبعة من وثيقة DES تتضمن مواصفات خوارزمية DEA الواردة هنا بالإضافة إلى خوارزمية (3DES) التي سنتناولها فيما بعد. وتمثل كلٌّ من DEA و3DES جزءاً من معيار تشفير البيانات. وعلاوة على ذلك – إلى أن تم اعتماد الرسمي لمصطلح 3DES مؤخراً – كانت خوارزمية DEA الثلاثية يشار إليها عادةً بـ triple DES بينما تُكتب كـ 3DES. لتسهيل الأمر هنا سوف نستخدم 3DES.

الأولى وK_{15} في الجولة الثانية، وهكذا إلى أن تستعمل K_1 في الجولة الأخيرة (رقم 16).

❖ قوة خوارزمية DES:

تنقسم المخاوف بشأن قوة DES إلى فئتين: مخاوف حول الخوارزمية نفسها، ومخاوف حول طول المفتاح المستخدم (56 بتاً). تتركز المخاوف من الفئة الأولى حول إمكانية النجاح في تحليل الشفرة من خلال استغلال خصائص خوارزمية DES. على مر السنين، كانت هناك محاولات عديدة لاكتشاف واستغلال نقاط الضعف في الخوارزمية، مما جعل DES أكثر خوارزميات التشفير التي دُرست. ورغم قيام العديد من تلك المحاولات، فإنه لم ينجح أحد حتى الآن في اكتشاف نقطة ضعف قاتلة في DES.[3]

يُعدُّ طول المفتاح أكثر مدعاةٍ للقلق. بمفتاح طوله 56 بتاً يكون عدد المفاتيح الممكنة 2^{56}، أي حوالي 7.2×10^{16} مفتاحاً. وهكذا فظاهرياً يبدو أن الهجوم الاستقصائي غير عملي. فبافتراض أنه في المتوسط يحتاج الأمر إلى تجربة نصف عدد المفاتيح أثناء عملية البحث، فإن آلة تشفير DES واحدة تنفذ عملية تشفير كل 1 ميكروثانية ستستغرق أكثر من ألف سنة لفك الشفرة (انظر الجدول 2-2).

غير أن افتراض ميكروثانية واحدة كسرعة تشفير يُعد افتراضاً مفرطاً في التحفظ. أخيراً وفي يوليو 1998 ثبت بشكلٍ قاطع أن خوارزمية DES غير آمنة، عندما أعلنت مؤسسة Electronic Frontier Foundation (EFF) أنها تمكنت من فك شفرة DES باستخدام آلة صُمّمت خصيصاً لفك شفرات DES التي تم إنتاجها بكلفة لا تتجاوز 250 ألف دولار، حيث استغرق ذلك أقل من ثلاثة أيام. وقد نشرت مؤسسة EFF وصفاً تفصيلياً للآلة، مما مكّن الآخرين من بناء آلاتهم الخاصة لفك شفرات DES [EFF98]. وبطبيعة الحال ستستمر أسعار عتاد الحاسبات في الانخفاض بينما تتزايد سرعاتها، مما سيجعل خوارزمية DES عديمة القيمة تقريباً في نهاية المطاف.

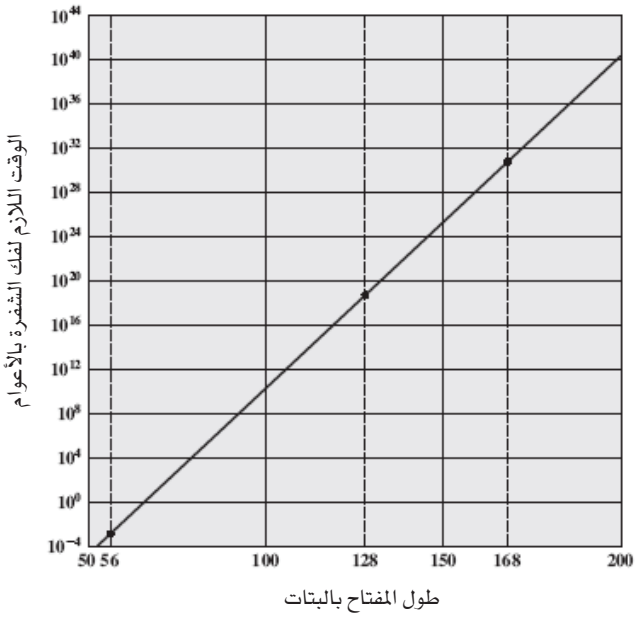
---

[3] على الأقل، لم يعلن أحد عن مثل هذا الاكتشاف بعد.

غير أنه من المهم الإشارة إلى أن عملية البحث عن المفتاح لا تقتصر على مجرد تجربة كل المفاتيح الممكنة الواحد تلو الآخر. فمحلل الشفرة ينبغي أن تكون لديه القدرة على التعرف على الرسالة الناتجة من إزالة التشفير كنص عادي، اللهم إلا إذا كانت الرسالة الأصلية معروفة فعلاً. إذا كانت الرسالة مجرد نص عادي باللغة الإنجليزية، فإنها ستظهر فجأة وبوضوح عند استخدام المفتاح الصحيح، ولكن ينبغي أتمتة عملية التعرف على النص الإنجليزي. إذا كانت الرسالة النصّية قد تم ضغطها قبل التشفير فإن التعرف عليها سيكون أكثر صعوبة. وإذا كانت الرسالة من نوع عام من البيانات كالبيانات العددية، وكانت مضغوطة، فإن أتمتة عملية التعرف ستصبح أكثر صعوبة. وعليه فإننا نحتاج إلى دعم الأسلوب الاستقصائي في البحث عن المفتاح بقدر من المعلومات عن النص العادي المتوقع استرجاعه وبعض وسائل التمييز التلقائي بين الرسالة الأصلية المتوقعة والرسائل المشوَّشة الناتجة عن استخدام مفتاح غير صحيح. إن الطريقة التي اتبعتها مؤسسة EFF لفك الشفرة تناولت هذه الأمور واستخدمت بعض أساليب الأتمتة التي يمكن أن تكون فعالة في كثير من المواقف.

وثمة نقطة أخيرة: إذا كان الشكل الوحيد من أشكال الهجوم التي يمكن استخدامها ضد خوارزمية التشفير هو الهجوم الاستقصائي، فإن السبيل لمواجهة مثل تلك الهجمات يكون واضحاً: استخدم مفاتيح أطول. لتحصيل تصور تقريبي عن طول المفتاح المطلوب، لنستخدم آلة EFF لفك الشفرة كأساس لعمل التقديرات اللازمة. جدير بالذكر أن تلك الآلة كانت نموذجاً أولياً ويمكننا افتراض أن الأجهزة التي تستخدم تقنيات اليوم ستكون أسرع وأكثر فاعلية من حيث التكلفة. لو افترضنا أن جهاز فك الشفرة يمكنه إزالة التشفير بمعدل مليون مرة كل ميكروثانية، وهو نفس المعدل المذكور في العمود الأخير من الجدول 2-2، فإن فك شفرة DES سيستغرق نحو 10 ساعات. ويمثل ذلك تسريعاً بحوالي 7 مرات مقارنةً بنتائج آلة EFF. باستخدام هذه النسبة، ويبيّن الشكل 2-3 الوقت اللازم لفك رسالة مشفَّرة بخوارزمية من طراز DES كدالة في طول المفتاح. فعلى سبيل المثال، باستخدام مفتاح طوله 128 بتاً، وهو طول شائع الاستخدام بين خوارزميات التشفير المعاصرة، يستغرق الأمر أكثر من 10^{18} عاماً لفك الشفرة باستخدام آلة EFF. حتى

لو أمكن تسريع إزالة التشفير الذي تقوم به الآلة بنسبة تريليون واحد (10^{12})، فسيستغرق فك الشفرة أكثر من مليون عاماً. وعليه، فإن استخدام مفتاح طوله 128 بتاً يضمن عملياً عدم اختراق (فك شفرات) الخوارزمية بواسطة الأساليب الاستقصائية لاكتشاف المفتاح.



الشكل 2-3: الوقت اللازم لفك الشفرة
(بافتراض إزالة التشفير بمعدل 10^6 مرة كل ميكروثانية)

2-2-2 خوارزمية DES الثلاثية

تم اعتماد خوارزمية DES الثلاثية (3DES) كمعيار للاستخدام في التطبيقات المالية في عام 1985 وذلك ضمن معيار المعهد الوطني الأمريكي للمعايير (ANSI) رقم X9.17. وفي عام 1999 تم تضمين 3DES كجزء من معيار تشفير البيانات بموجب النشرة FIPS PUB 46-3. تستخدم 3DES ثلاثة مفاتيح وتتضمن تنفيذ

خوارزمية DES ثلاث مرات. تتبع الخوارزمية تسلسلاً من عمليات التشفير (إزالة التشفير) التشفير ((encrypt-decrypt-encrypt (EDE) (انظر الشكل 2-4 (a)):

$$C = E(K_3, D(K_2, E(K_1, P)))$$

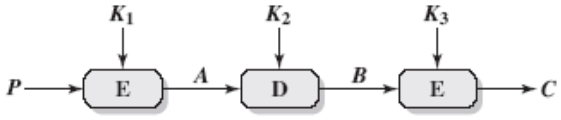
حيث:

C = الرسالة المشفَّرة

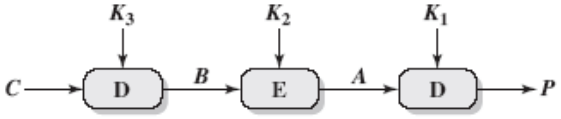
P = الرسالة الأصلية

$E(K,X)$ = تشفير الرسالة X باستخدام المفتاح K

$D(K,Y)$ = إزالة تشفير الرسالة Y باستخدام المفتاح K



(a) التشفير



(b) إزالة التشفير

الشكل 2-4: خوارزمية DES الثلاثية (3DES).

عملية إزالة التشفير ببساطة هي نفسها عملية التشفير مع عكس ترتيب استخدام المفاتيح (انظر الشكل 2-4 (b)):

$$P = D(K_1, E(K_2, E(K_3, C)))$$

ليست هناك أهمية تشفيرية معينة لعملية إزالة التشفير في المرحلة الثانية من خوارزمية DES3. الميزة الوحيدة لذلك هي تمكين مستخدمي 3DES من إزالة تشفير البيانات المشفَّرة بواسطة مستخدمي خوارزمية DES القديمة المفردة:

$$C = E(K_1, D(K_1, E(K_1, P))) = E[K,P]$$

باستخدام ثلاثة مفاتيح مختلفة، يكون لخوارزمية 3DES مفتاح بطول فعّال يبلغ 168 بتاً. أما معيار FIPS 46-3 فيسمح أيضاً باستخدام مفتاحين، وذلك بجعل $K_1 = K_2$، ومن ثم يكون الطول الفعّال للمفتاح 112 بتاً. يتضمن معيار FIPS 46-3 المبادئ التوجيهية الآتية فيما يتعلق باستخدام 3DES:

- 3DES هي خوارزمية التشفير المتماثل المعتمدة والمفضلة لدى FIPS.
- يسمح المعيار للأنظمة المتقادمة (legacy systems) باستخدام خوارزمية DES الأصلية، والتي تستخدم مفتاحاً واحداً طوله 56 بتاً، ولكن ينبغي أن تدعم كل الأنظمة الجديدة خوارزمية 3DES.
- تشجيع الهيئات الحكومية التي تستخدم أنظمة DES متقادمة على الانتقال إلى 3DES.
- من المتوقع أن تتعايش كلٌّ من 3DES ومعيار التشفير المتقدم (AES) كخوارزميات معتمدة من قِبَل FIPS، مما يسمح بالانتقال التدريجي إلى AES.

من الواضح صعوبة اختراق خوارزمية 3DES نظراً لأن الخوارزمية الأساسية وراءها هي خوارزمية DEA، ولهذا فإن 3DES يمكن أن تدّعي لنفسها نفس مقاومة تحليل الشفرة التي تتوفَّر لخوارزمية DEA. فضلاً عن ذلك، ونظراً لطول المفتاح الذي يبلغ 168 بتاً فإن الهجمات الاستقصائية لفك الشفرات تصبح مستحيلة من الناحية العملية.

في نهاية المطاف يُفترض أن تحل AES محل 3DES، ولكن هذه العملية سوف تستغرق عدة سنوات. يتوقع معهد NIST أن تبقى 3DES كخوارزمية معتمدة (لاستخدام الحكومة الأمريكية) في المستقبل المنظور.

3-2-2 خوارزمية معيار التشفير المتقدم (AES)

تمتاز 3DES بميزتين مهمتين تضمنان استخدامها على نطاق واسع خلال السنوات القليلة القادمة. أولاً، بمفتاح طوله 168 بتاً تتغلب الخوارزمية على نقاط الضعف التي عانت منها DEA في مواجهة الهجمات الاستقصائية. وثانياً، فإن خوارزمية التشفير الأساسية وراء 3DES هي نفس الخوارزمية المستخدمة في DEA والتي تعرضت للفحص والتمحيص أكثر من أي خوارزمية تشفير أخرى على مدى فترة طويلة من الزمن، ولم يثبت وجود أي هجوم تحليل شفرات فعال ضدها مبني على أساس الخوارزمية نفسها وليس على مبدأ الحل الاستقصائي. وبناءً على ذلك، فهناك درجة عالية من الثقة بأن 3DES تتمتع بمقاومة عاليةٍ جداً لتحليل الشفرة. وإذا كان الأمن هو العامل الوحيد الذي يُؤخذ بعين الاعتبار، فإن 3DES تصلح لاختيارها كخوارزمية التشفير المعيارية لعقود قادمة.

يكمن العيب الرئيس لخوارزمية 3DES في كونها بطيئةٍ نسبياً عند تنفيذها برمجياً، حيث كانت خوارزمية DEA الأصلية قد صُممت في منتصف السبعينيات بغرض تنفيذها في عتاد (hardware)، ولكن تنفيذها من خلال برمجيات لا يتمتع بكفاءة عالية. وعلى هذا فإن 3DES، والتي تستخدم ثلاثة أضعاف عدد الجولات التي تستخدمها DEA، ستكون من ثَمَّ أبطأ بنفس النسبة. ومن العيوب الثانوية أن كلاً من DEA و3DES تستخدم كتلاً بحجم 64 بتاً. لأسباب تتعلق بالكفاءة والأمن، يُعدُّ استخدام الكتل ذات الحجم الأكبر أمراً مرغوباً فيه.

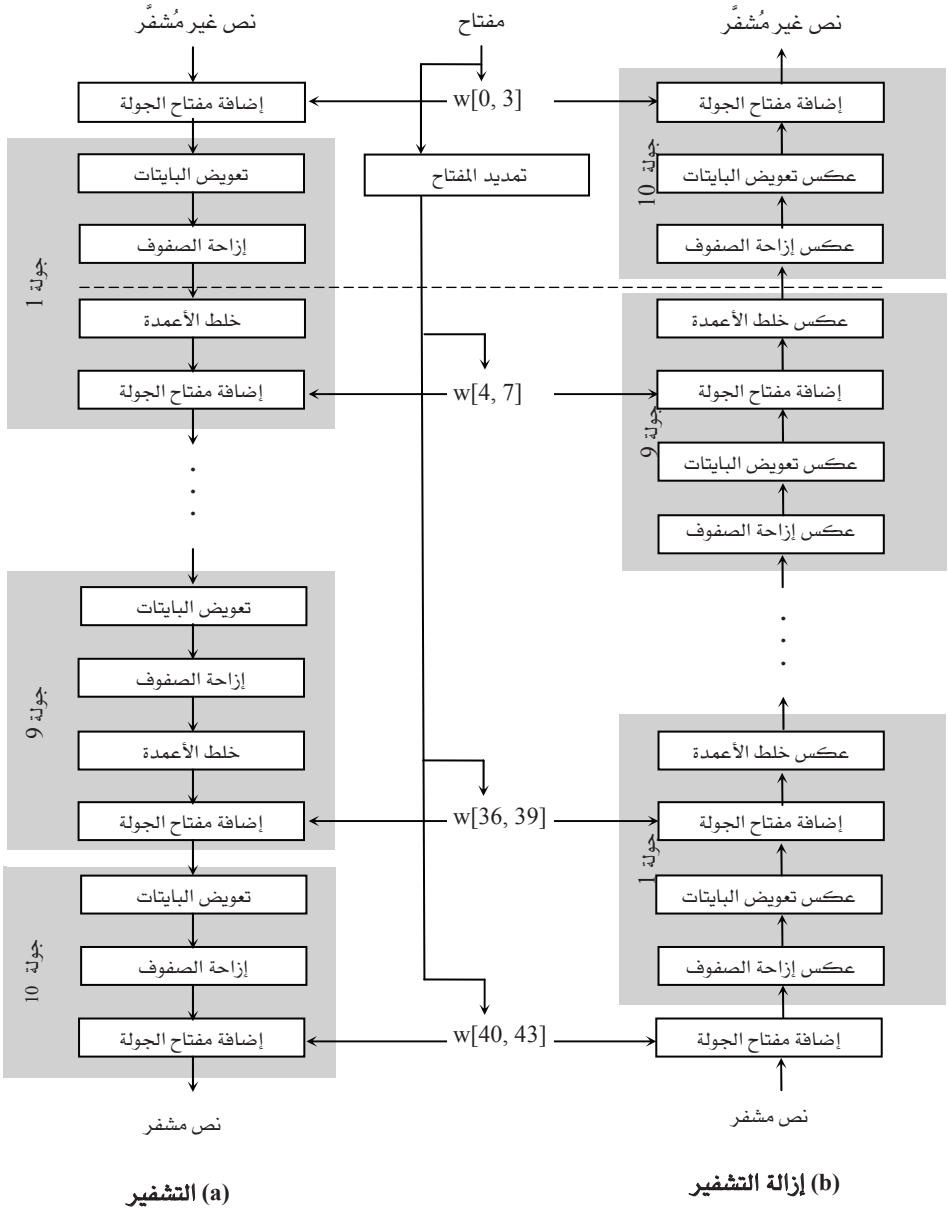
بسبب هذه العيوب، لا تُعدُّ 3DES مرشحاً معقولاً للاستخدام على المدى البعيد. وكبديل، أصدر معهد NIST في عام 1997 نداءً لتقديم عروض بمقترحات جديدة لمعيار التشفير المتقدم ((Advanced Encryption Standard (AES))، والذي ينبغي أن يوفر قوة أمنية تعادل أو تتفوق على قوة 3DES مع تحسين كفاءة الأداء بشكل كبير. وبالإضافة إلى تلك المتطلبات العامة، اشترط معهد NIST أن تكون AES خوارزمية تشفيرٍ متماثل، وأن تستخدم كتل بيانات بحجم 128 بتاً وتدعم مفاتيح بأطوال 128، و192، و256 بتاً. وتضمَّنت قائمة معايير التقييم لمقترحات الخوارزمية الجديدة: الأمن، والكفاءة الحسابية، واحتياجات الذاكرة، والملاءمة للتنفيذ في عتاد أو برمجيات، والمرونة.

في الجولة الأولى من عملية التقييم، تم قبول 15 خوارزمية مقترحة، وعبر الجولة الثانية أمكن تضييق مجال الاختيار إلى 5 خوارزميات. في نوفمبر 2001 انتهى معهد NIST من عملية التقييم ونشر وثيقة المعيار النهائي (FIPS PUB 197). وقد اختار معهد NIST خوارزمية Rijndael المقترحة لتكون خوارزمية AES الجديدة. والباحثان اللذان تقدما بهذا المقترح هما أخصائيا التشفير Joan Daemen وVincent Rijmen من بلجيكا.

❖ لمحة عن الخوارزمية:

تستخدم AES كتلة مقاسها 128 بتاً، ومفتاحاً طوله إمّا 128، أو 192، أو 256 بتاً. في وصفنا للخوارزمية هنا سنفترض مفتاحاً بطول 128 بتاً، وهو الطول المُتوقَّع أن يكون الأكثر شيوعاً.

يبيّن الشكل 2-5 هيكل خوارزمية AES. مُدخل الخوارزمية عبارة عن كتلة واحدة حجمها 128 بتاً. تُمثَّل تلك الكتلة على شكل مصفوفة مربّعة من البايتات في الوثيقة رقم FIPS PUB 197. ويتم نسخ الكتلة إلى صفيف الحالة (State array) حيث يجري تعديلها في كل مرحلة من مراحل التشفير أو إزالة التشفير. بعد المرحلة الأخيرة يتم نسخ صفيف الحالة إلى مصفوفة المُخرج (output matrix). بالمثل، يُمثّل المفتاح بطول 128 بتاً كمصفوفة مربعة من البايتات. ويتم بعد ذلك تمديد المفتاح إلى صفيف من كلمات جدولة المفتاح (key schedule words)؛ حيث تضم كل كلمة منها أربعة بايتات. يتألف الجدول الكلي للمفتاح بطول 128 بتاً من 44 كلمة. وتُرتّب البايتات ضمن المصفوفات حسب الأعمدة. لذا، وعلى سبيل المثال، تحتل البايتات الأربعة الأولى من كتلة الرسالة الأصلية بطول 128 بتاً العمود الأول من مصفوفة المُدخل (in matrix)، وتحتل البايتات الأربعة التالية العمود الثاني، وهكذا. وبالمثل، فإن البايتات الأربعة الأولى من المفتاح الممدد، والتي تشكل كلمة، تحتل العمود الأول من المصفوفة **w** (الشكل 2-5).



الشكل 2-5: التشفير وإزالة التشفير في خوارزمية AES.

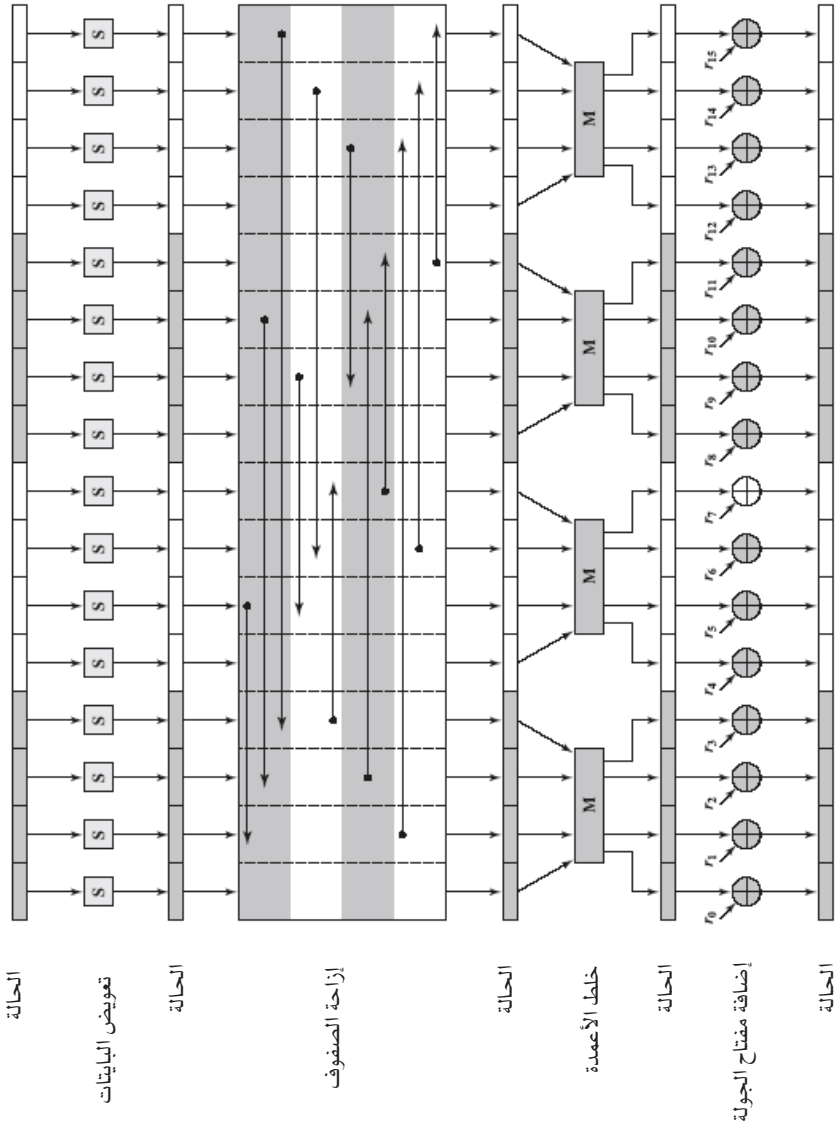
تُلقِي التعليقات الآتية بعض الضوء على خوارزمية AES:

1. من السمات الجديرة بالذكر لهذه الخوارزمية أنها لا تتبع هيكل فيستل. ويُذكر أنه في هيكل فيستل التقليدي تُستخدم نصف كتلة البيانات لتعديل النصف الآخر، ثم يتم بعد ذلك تبديل النصفين. فخوارزمية AES لا تستخدم هيكل فيستل، ولكنها تقوم بمعالجة كتلة البيانات بأكملها على التوازي أثناء كل جولة وذلك باستخدام عمليتي التعويض والتبديل.
2. يتم تمديد المفتاح الذي يقدم كمدخل للخوارزمية إلى صفيف يتألف من 44 كلمة كلٌّ منها (أي $\mathbf{w}[i]$) بطول 32 بتاً. تستخدم كل جولة من جولات المعالجة أربع كلمات مختلفة (128 بتاً) من هذا الصفيف كمفتاح جولة.
3. تستخدم الخوارزمية أربع مراحل مختلفة، واحدة للتبديل وثلاث للتعويض:
   - تعويض البايتات: حيث يُستخدَم جدول، يشار إليه بصندوق التعويض (S-box)[4] لتعويض بايتات كتلة البيانات بايتاً بايتاً.
   - إزاحة الصفوف: تبديل بسيط يتم على الصفوف صفاً صفاً.
   - خلط الأعمدة: تعويض يغيّر كل بايت في العمود كدالة في كل بايتات العمود.
   - إضافة مفتاح الجولة: عملية XOR بسيطة تُجرى على مستوى البتات بين البيانات الحالية للكتلة وجزءٍ من المفتاح الممدد.
4. هيكل بسيط للغاية. لكلٍّ من التشفير وإزالة التشفير، تبدأ الخوارزمية بمرحلة إضافة مفتاح الجولة، وتليها تسع جولات تتكون كل واحدة منها من الأربع مراحل المذكورة، وتعقبها جولة عاشرة تتألف من ثلاث مراحل. ويوضح الشكل 2-6 الهيكل الكامل لعملية التشفير.
5. يُستخدم المفتاح في مرحلة واحدة فقط من المراحل الأربع (مرحلة إضافة مفتاح الجولة). ولهذا السبب فإن الخوارزمية تبدأ وتنتهي بمرحلة إضافة مفتاح الجولة. أي مرحلة أخرى، سواءً طُبّقت في البداية أو في النهاية، قابلة للعكس دون الحاجة لمعرفة المفتاح ومن ثم فإنها لا تضيف أي أمن.

---

[4] يُستخدم المصطلح S-box (صندوق التعويض) بكثرة في وصف أنظمة التشفير المتماثل للإشارة إلى جدول يُستخدم مع آلية للتعويض مبنية على الرجوع إلى جدول لتحديد التعويض المطلوب.

6. يلاحظ أن مرحلة إضافة مفتاح الجولة في حد ذاتها لن تكون صعبة الاختراق. كما أن المراحل الثلاث الأخرى مجتمعة تقوم بخلط البيانات ولكنها بحد ذاتها لا توفّر الأمن لأنها لا تستخدم المفتاح. يمكن النظر للخوارزمية كعمليات متعاقبة من تشفير الكتلة بدالة XOR (إضافة مفتاح الجولة)، يليها خلط بايتات الكتلة (بالمراحل الثلاث الأخرى)، يليها تشفير بدالة XOR وهكذا دواليك. ويتسم هذا النظام بأنه فعّال وآمن للغاية.
7. يمكن عكس كل مرحلة من المراحل الأربع بسهولة. لعكس كلٍّ من مراحل تعويض البايتات، وإزاحة الصف، وخلط الأعمدة، تستخدم دالة عكسية في خوارزمية إزالة التشفير. أما بالنسبة لمرحلة إضافة مفتاح الجولة، فتتم الوظيفة العكسية بعملية XOR لمفتاح الجولة نفسه مع بيانات الكتلة، تطبيقاً للعلاقة: $A \oplus A \oplus B = B$.
8. كما هو الحال مع معظم أنظمة تشفير الكتلة، تستخدم خوارزمية إزالة التشفير المفتاح الممدد بترتيب عكسي. ومع ذلك، فخوارزمية إزالة التشفير ليست مطابقة لخوارزمية التشفير، وذلك نتيجة للطبيعة الخاصة لهيكلة AES.
9. بمجرد التأكد من أن المراحل الأربع جميعاً قابلة للعكس، يسهل التحقق من أن إزالة التشفير تستعيد بالفعل الرسالة الأصلية . يبيّن الشكل 2-5 أن خطوات التشفير وإزالة التشفير تسيران في اتجاهين متضادين رأسياً. عند كل نقطة أفقية (على سبيل المثال، على الخط الأفقي المنقط في الشكل) تكون لعمليتي التشفير وإزالة التشفير نفس الحالة (state).
10. تتألف الجولة الأخيرة من كلٍّ من عمليتي التشفير وإزالة التشفير من ثلاث مراحل فقط. مرة أخرى، يرجع ذلك إلى الطبيعة الخاصة لهيكلة AES، كما أنه مطلوب لجعل عملية التشفير قابلة للعكس.



الشكل 2-6: جولة تشفير ضمن خوارزمية AES.

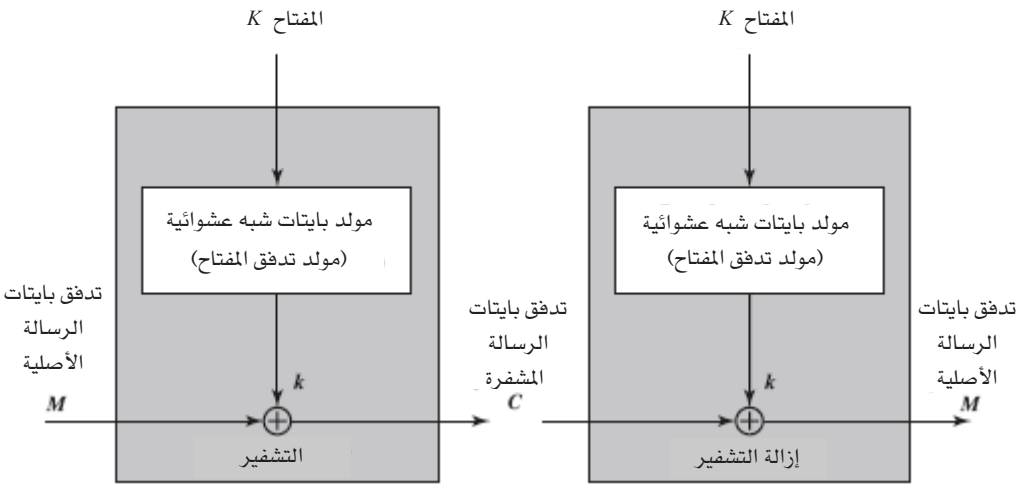
3-2 تشفير التدفق وخوارزمية RC4

يقوم نظام تشفير الكتلة بتشفير عناصر بيانات الكتلة المُدخلة في المرة الواحدة منتِجاً كتلة مُخرج مشفَّرة لكل كتلة مُدخل. أما نظام تشفير التدفق فيقوم بمعالجة عناصر البيانات الداخلة باستمرار عنصراً عنصراً في كل مرة، منتجاً عنصر مُخرج مُشَفَّراً لكل عنصر مُدخل بطريقةٍ مستمرة. ورغم أن تشفير الكتلة يُعدُّ أكثر شيوعاً بكثير، فإن هناك تطبيقات معينة يُعدُّ تشفير التدفق أكثر ملاءمةً لها، وسنورد بعض الأمثلة على ذلك لاحقاً في هذا الكتاب. في هذا الجزء سنلقي نظرةً على نظام RC4 والذي ربما يُعدُّ أشهر أنظمة تشفير التدفق المتماثلة. نبدأ بلمحة عامة عن هيكل تشفير التدفق ثم نتناول خوارزمية RC4 بشيءٍ من التفصيل.

2-3-1 هيكل تشفير التدفق

يقوم نظام تشفير التدفق في العادة بتشفير بايت واحد من الرسالة الأصلية في كل مرة، ومع ذلك فمن الممكن تصميم النظام بحيث يعمل على بت واحد في كل مرة أو على وحدات أكبر من بايت واحد في كل مرة. ويمثل الشكل 2-7 هيكل تشفير التدفق. وفي هذا الهيكل يتم إدخال مفتاح إلى مولد بتات شبه عشوائي (pseudorandom) يُنتِج سلسلة من الأرقام بطول 8 بتات لكلٍّ والتي تبدو عشوائية. التدفق شبه العشوائي هو الذي لا يمكن التنبؤ به بدون معرفة المفتاح الداخل والذي يبدو كما لو كان ذا طابع عشوائي. ويتم دمج مخرج المولّد، والذي يُعرف بتدفق المفتاح keystream، مع تدفق الرسالة الأصلية بمعدل بايت في كل مرة، باستخدام عملية "أو-الحصرية" (XOR) على البتات. على سبيل المثال، إذا كان البايت الخارج من المولد هو 01101100 والبايت المقابل من الرسالة الأصلية هو 11001100، فإن البايت المشفَّر المُخرج في هذه الحالة يكون:

بايت الرسالة الأصلية	11001100
بايت تدفق المفتاح	⊕ 01101100
البايت المشفَّر المُخرج	10100000



الشكل 2-7: هيكل تشفير التدفق.

تتطلب إزالة التشفير استخدام نفس تسلسل الأرقام شبه العشوائي المستخدم أثناء التشفير:

بايت الرسالة المشفَّرة	10100000
بايت تدفق المفتاح	⊕ 01101100
بايت الرسالة الأصلية المُخرج	11001100

يشبه نظام تشفير التدفق نظام حشوة المرة الواحدة (one-time pad). ويكمن الفرق في أن حشوة المرة الواحدة تستخدم سلسلة أرقاماً عشوائيةً بحق، بينما يَستخدم نظام تشفير التدفق أرقاماً شبه عشوائية. ويورد [KUMA97] اعتبارات التصميم الآتية فيما يتعلق بتشفير التدفق:

1. ينبغي أن تكون فترة التكرار لسلسلة التشفير طويلة، حيث يَستخدم مولد الأرقام شبه العشوائية دالةً تنتج سلسلة أرقام محددة تتكرر دورياً. وكلما ازداد طول فترة التكرار، كان من الصعب القيام بتحليل الشفرة.

2. ينبغي أن يقارب تدفق المفاتيح في خصائصه مولِّد أرقام عشوائية حقيقياً بأقرب قدرٍ ممكن. فعلى سبيل المثال، ينبغي أن يكون هناك عددٌ متساوٍ تقريباً من القيم 0 و1. وإذا نظرنا إلى تدفق المفاتيح كسلسلة من البايتات، فينبغي أن يتكرر ظهور كل قيم البايتات الـ 256 نفس عدد المرات تقريباً. وكلما كان تدفق المفاتيح أكثر عشوائيةً، كلما كانت الرسالة المشفَّرة أكثر عشوائية، مما يزيد من صعوبة عملية تحليل الشفرة.
3. نلاحظ من الشكل 2-7 أن الناتج من مولد الأرقام العشوائية يعتمد على قيمة المفتاح المستخدم. للحماية ضد الهجمات الاستقصائية، ينبغي أن يكون المفتاح طويلاً بدرجة كافية. وتنطبق هنا أيضاً نفس الاعتبارات التي تنطبق على أنظمة تشفير الكتلة. وعليه، فباستخدام التقنية المتوفِّرة حالياً يُنصح باستخدام مفتاح لا يقل طوله عن 128 بتاً.

باستخدام مولد أرقام عشوائية مصمم جيداً، يمكن الحصول على نظام تشفير تدفق بنفس درجة أمان نظام تشفير كتلة له نفس طول المفتاح تقريباً. وتكمن الميزة الرئيسة لأنظمة تشفير التدفق على أنظمة تشفير الكتلة في كونها أسرع وأنها تستخدم كوداً أصغر بكثير. فخوارزمية RC4 مثلاً التي نتناولها في هذا الجزء يمكن تنفيذها في بضعة أسطر من الكود. يَستخدم الجدول 2-3 بيانات من [RESC01] للمقارنة بين سرعة تنفيذ RC4 وثلاث خوارزميات أخرى مشهورة لتشفير الكتلة. غير أن تشفير الكتلة يمتاز بإمكانية إعادة استخدام المفاتيح، أما في حالة تشفير التدفق فإن تشفير رسالتين عاديتين باستخدام نفس المفتاح يؤدي غالباً إلى جعل عملية تحليل الشفرة أمراً بسيطاً للغاية [DAWS96]. عند إجراء عملية XOR على تدفقي نصّين مشفَّرين تكون النتيجة هي نفس ناتج إجراء XOR على تدفقي النصّين الأصليين قبل التشفير. فإذا كان النص غير المشفَّر سلسلة من الحروف، أو أرقام بطاقات الائتمان، أو غيرها من سلسلة البايتات التي تحكمها قواعد معروفة، فقد تنجح عمليات التحليل في فك الشفرة.

الجدول 2-3: مقارنة سرعات تنفيذ التشفير المتماثل على معالج Pentium II.

نظام التشفير	طول المفتاح	السرعة (ميجابت/ثانية)
DES	56	9
3DES	168	3
RC2	متغير	0.9
RC4	متغير	45

للتطبيقات التي تتطلب تشفير/إزالة تشفير بيانات متدفقة، كتلك التي تمر عبر قناة اتصال أو وصلة متصفح/إنترنت، قد يكون تشفير التدفق هو البديل الأفضل. أما في التطبيقات التي تتعامل مع كتل من البيانات؛ كنقل الملفات والبريد الإلكتروني وقواعد البيانات، فقد يكون من الأنسب استخدام تشفير الكتلة. ومع ذلك، فمن الممكن استخدام كلا الأسلوبين للتشفير في أي تطبيق تقريباً.

2-3-2 خوارزمية RC4

خوارزمية RC4 هي خوارزمية تشفير تدفق صمّمها رون ريفيست من شركة RSA Security في عام 1987. تستخدم الخوارزمية مفتاحاً بطول متغير، وتجري عملياتها على البايتات، وهي مبنية على أساس التبديل العشوائي. تبيّن التحاليل أن فترة التكرار للخوازمية طويلة للغاية، حيث تزيد على 10^{100} [ROBS95a]. تتضمَّن الخوارزمية إجراء ما بين 8 و16 عملية آلة (machine operations) لإنتاج كل بايت في المخرج، ويُتوقع أن تعمل RC4 بسرعات عالية عند تنفيذها برمجياً. تُستخدم RC4 في معايير SSL/TLS (Secure Sockets Layer/Transport Layer Security) (طبقة المقابس الآمنة/ أمن طبقة النقل) والتي تم تحديدها للاتصال بين المتصفحات والخوادم على شبكة الإنترنت. كما أنها تستخدم في بروتوكول WEP (الخصوصية المكافئة للشبكات السلكية Wired Equivalent Privacy) والبروتوكول الأحدث WPA (الوصول اللاسلكي المحمي Wireless Protected Access) واللذان يشكّلان جزءاً من معيار IEEE 802.11 للشبكات اللاسلكية المحلية. احتفظت شركة RSA

Security بخوارزمية RC4 كسرٌ تجاري خاص بها. غير أنه في سبتمبر 1994 قام مجهولون بوضع الخوارزمية على قائمة Cypherpunks الخاصة بممرّري البريد المجهولين على شبكة الإنترنت.

تمتاز خوارزمية RC4 بالبساطة الشديدة وبسهولة شرحها. ويُستخدم مفتاح بطول متغير من 1 إلى 256 بايتاً (8 إلى 2048 بتاً) لتهيئة متجه الحالة S (state vector) بطول 256 بايتاً، والذي يتألف من العناصر S[0], S[1], . . ., S[255]. في جميع الأوقات، ويحتوي المتجه S على تبديلة من كل الأرقام بطول 8 بتات (من 0 إلى 255). للقيام بعملية التشفير وإزالة التشفير يتم توليد بايت k (انظر الشكل 2-7) من المتجه S باختيار أحد العناصر الـ 256 بشكل نظامي. وفي كل مرة يتم توليد قيمة جديدة للبايت k يتم أيضاً تبديل عناصر المتجه S مرة أخرى.

❖ تهيئة متجه الحالة S:

لبدء تشغيل الخوارزمية يتم تحديد قيم عناصر متجه الحالة S لتأخذ القيم من 0 إلى 255 بترتيب تصاعدي؛ أي S[0] = 0, S[1] = 1, … , S[255] = 255. كما يتم توليد متجه مؤقت جديد T. إذا كان طول المفتاح K هو 256 بايتاً يتم نقل K بأكمله إلى T، وإلا فإنه لمفتاح طوله keylen بايتاً يتم نسخ أول keylen عنصراً من K إلى T ثم يكرر K عدة مرات حسب الحاجة لملء T. ويمكن تلخيص هذه العمليات الأولية على النحو الآتي:

```
/* التهيئة */
for i = 0 to 255 do
    S[i] = i;
    T[i] = K[i mod keylen];
```

بعد ذلك نستخدم T لإنتاج التبديلة الأولية لـ S. ويتم ذلك بالبدء من S[0] والاتجاه إلى S[255]، ولكل قيمة S[i] يتم تبديل S[i] ببايت أخرى من S تبعاً لنظام تحدده T[i]:

```
/* S التبديلة الأولية لـ */
j = 0;
for i = 0 to 255 do
      j = (j + S[i] + T[i]) mod 256;
      Swap (S[i], S[j]);
```

ونظراً إلى أن العملية الوحيدة المستخدمة هي عملية تبادل، فإن الأثر الوحيد هو تبديل S. في نهاية العملية سيبقى S يحتوي على جميع الأرقام من 0 إلى 255.

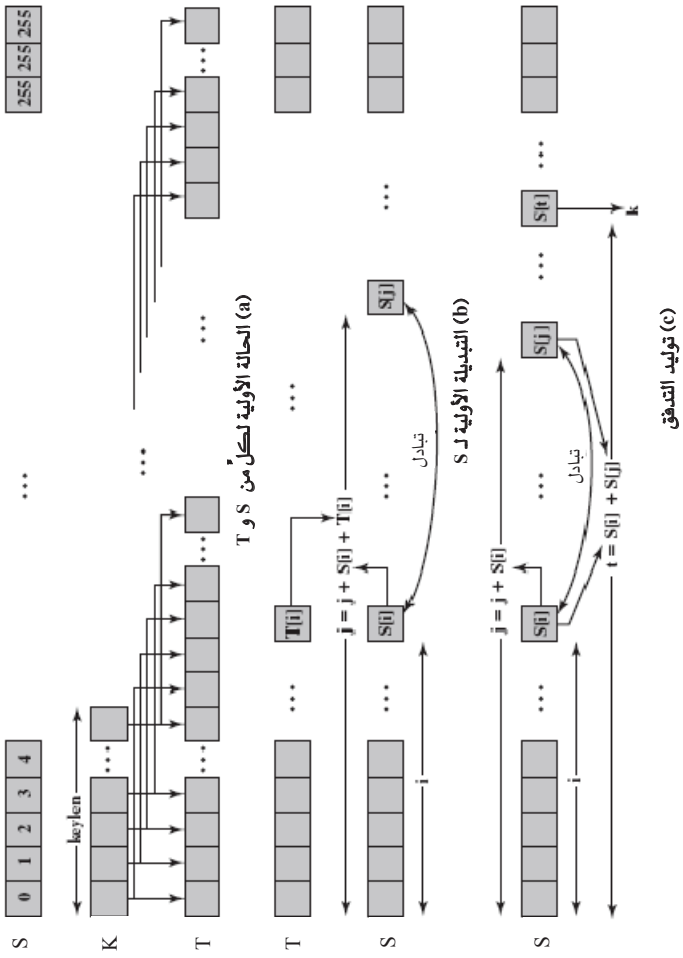
❖ توليد التدفق:

بعد الانتهاء من تهيئة المتجه S لا تكون هناك حاجة لاستخدام المفتاح المُدخل بعد ذلك. ويتضمَّن توليد التدفق التدوير عبر كل العناصر S[i]، ولكل عنصر S[i] يتم تبديل S[i] ببايت آخر من S تبعاً لنظام تحدده الترتيبة الحالية لـ S. بعد الوصول إلى S[255] تستمر العملية بدءاً بـ S[0] من جديد:

```
/* توليد التدفق */
i, j = 0;
while (true)
   i = (i + 1) mod 256;
   j = (j + S[i]) mod 256;
   Swap (S[i], S[j]);
   t = (S[i] + S[j]) mod 256;
   k = S[t];
```

في عملية التشفير، تُجرى عملية XOR بين قيمة k والبايت التالي من الرسالة الأصلية. لإزالة التشفير تُجرى عملية XOR بين قيمة k والبايت التالي من الرسالة المشفَّرة.

ويوضح الشكل 2-8 منطق خوارزمية RC4.



الشكل 2-8: منطق خوارزمية RC4.

❖ **قوة خوارزمية RC4:**

نُشرت عدة أبحاث حول تحليل أساليب مهاجمة RC4، منها على سبيل المثال [KNUD98]، [MIST98]، [FLUH00]، [MANT01]، [PUDO02]، [PAUL03]، [PAUL04]. لم تثبت فعالية أيٍّ من أساليب الهجوم تلك ضد خوارزمية RC4 بمفتاح طوله 128 بتاً. يتناول [FLUH01] مشكلة أكثر خطورة، حيث يوضح المؤلفون أن بروتوكول WEP، والمفترض أنه يوفر السرية على شبكات 802.11 اللاسلكية المحلية، يكون عرضة لهجوم معين. وحقيقة الأمر أن المشكلة ليست في خوارزمية RC4 في حد ذاتها ولكن في الطريقة التي يتم بها توليد المفاتيح المستخدمة كمُدخل إلى RC4. ولا يبدو أن هذه المشكلة خطيرة فيما يتعلق بالتطبيقات الأخرى التي تستخدم RC4، كما أنه يمكن علاجها في بروتوكول WEP بتغيير الطريقة المستخدَمة لتوليد المفاتيح. وتوضح هذه المشكلة الصعوبات التي تكتنف تصميم نظام آمن يتضمَّن كلاً من وظائف التشفير والبروتوكولات التي تستخدمها.

4-2 أنماط التشغيل في تشفير الكتل

في تشفير الكتل تتم معالجة كتلة بياناتٍ واحدةٍ في كل مرة، ويبلغ طول الكتلة 64 بتاً في كلٍّ من DES و3DES. للتعامل مع رسائل أصلية أطول من ذلك، يتعين تقسيم الرسالة الأصلية إلى كتل يضم كلٌّ منها 64 بتاً (وحشو الكتلة الأخيرة عند اللزوم لتصبح 64 بتاً). وأبسط طريقة لعمل ذلك هي استخدام ما يعرف بنمط دفتر الشفرة الإلكتروني (electronic codebook (ECB))، حيث تعالَج كتلة من الرسالة الأصلية طولها 64 بتاً في كل مرة ويتم تشفير كل كتلة باستخدام نفس المفتاح. يُستخدم مصطلح codebook (دفتر الشفرة) هنا لأنه – لنفس المفتاح المُستخدَم – تُنتِج كل كتلة (64 بتاً) من الرسالة الأصلية رسالةً مشفَّرةً فريدة. وعليه، يمكن للمرء تخيل دفتر شفرة هائل يضم عنصر رسالة مشفَّرة (64 بتاً) لكل نمط ممكن من قيم الـ 64 بتاً التي يمكن أن تمثل كتلة من كتل الرسالة الأصلية.

في نمط ECB للتشغيل، إذا تكرر ظهور نفس الكتلة التي تضم 64 بتاً أكثر من مرة في الرسالة الأصلية، فإنها تنتج دائماً نفس الرسالة المشفَّرة. ولهذا السبب، فإن نمط ECB قد لا يوفر وسيلة تشفير آمنة للرسائل الطويلة. إذا كانت الرسالة منتظمة بدرجة كبيرة، فقد يتمكن محلل الشفرة من استغلال هذا الانتظام. على سبيل المثال، إذا كان معروفاً أن الرسالة تبدأ دائماً ببعض الحقول المحدّدة مسبقا، فقد يُتاح لمحلل الشفرة عددٌ من الأزواج المعروفة من الرسالة الأصلية والرسالة المشفَّرة المناظرة للعمل عليها. إذا تضمَّنت الرسالة عناصر متكررة بفترة تكرار تبلغ مضاعفات 64 بتاً، فسيكون بوسع محلل الشفرات تحديد تلك العناصر، مما قد يساعد في تحليل الشفرة أو يوفر فرصة لاستبدال أو إعادة ترتيب كتل البيانات في الرسالة. للتغلب على أوجه القصور الأمني لنمط ECB، يُفضل استخدام أسلوب يضمن أنه إذا ما تكررت نفس الكتلة من الرسالة الأصلية، فإنها تنتج كتلاً مختلفة من الرسالة المشفَّرة. وسنتناول في هذا الجزء أسلوبين يحققان هذا الشرط تم تعريفهما في الوثيقة FIPS PUB 81.

1-4-2 نمط سَلسَلة كتل الشفرة

في نمط سَلسَلة كتل الشفرة (cipher block chaining (CBC)) يدخل إلى خوارزمية التشفير ناتج إجراء عملية XOR على الكتلة الحالية من الرسالة الأصلية والكتلة السابقة من الرسالة المشفَّرة، ويُستخدم المفتاح نفسه لكل كتلة (انظر الشكل 2-9). ويؤدي ذلك في واقع الأمر إلى ربط معالجة الكتل المتعاقبة للرسالة الأصلية. فكتلة البيانات الداخلة لخوارزمية التشفير ليس لها علاقة ثابتة بالكتلة المناظرة من الرسالة الأصلية، ومن ثم فلن يكشف التشفير بهذا النمط عن تكرار أنماط الـ 64 بتاً في الرسالة الأصلية.

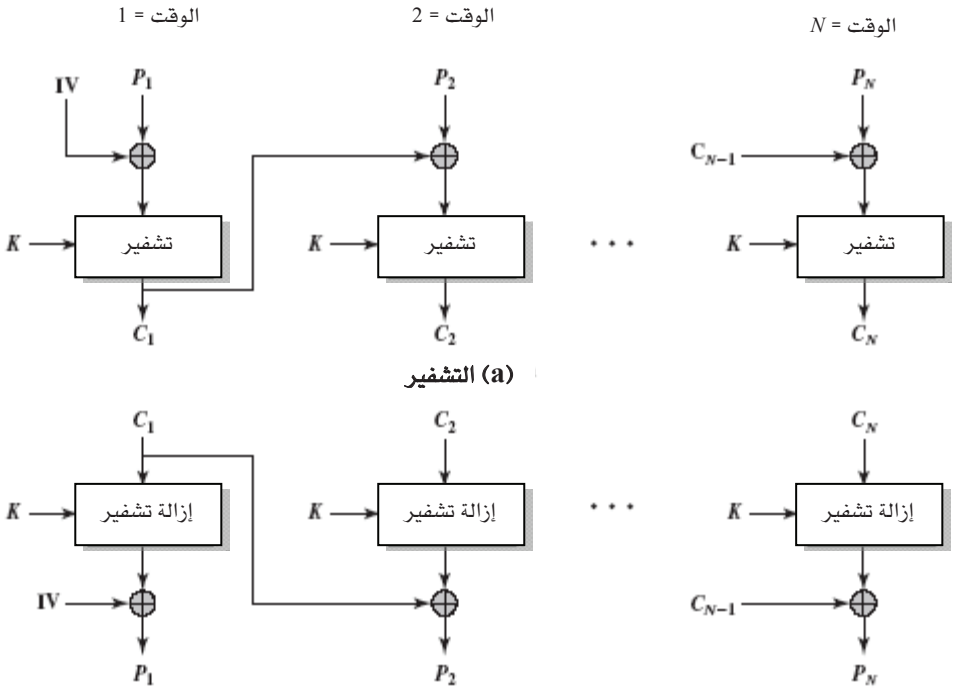
في عملية إزالة التشفير تُمرَّر كل كتل الشفرة عبر خوارزمية إزالة التشفير. تُجرى عملية XOR على النتيجة مع الكتلة السابقة من الرسالة المشفَّرة لإنتاج كتلة الرسالة الأصلية. للتحقق من أن ذلك يعطي النتيجة الصحيحة، يمكننا كتابة:

$$C_j = E(K, [C_{j-1} \oplus P_j])$$

حيث E[K,X] هو تشفير الرسالة الأصلية X باستخدام مفتاح K، والرمز ⊕ يمثل عملية "أو-الحصرية")، ومن ثم:

$$D(K, C_j) = D(K, E(K, [C_{j-1} \oplus P_j]))$$
$$D(K, C_j) = C_{j-1} \oplus P_j$$
$$C_{j-1} \oplus D(K, C_j) = C_{j-1} \oplus C_{j-1} \oplus P_j = P_j$$

مما يثبت صحة الشكل 2-9.



الشكل 2-9: نمط سلسلة كتل الشفرة (CBC).

لإنتاج أول كتلة من الرسالة المشفَّرة تُجرى عملية XOR بين متجه أوّلي (Initialization Vector (IV)) والكتلة الأولى من الرسالة الأصلية. في حالة إزالة التشفير، تُجرى عملية XOR بين المتجه الأوّلي IV وناتج خوارزمية إزالة التشفير للحصول على أول كتلة من الرسالة المشفَّرة.

يجب أن يكون المتجه IV معروفاً لكلٍّ من المرسل والمستلم. وينبغي حماية كلٍّ من IV والمفتاح لتحقيق أقصى قدر من الأمن. ويمكن أن يتم ذلك بإرسال IV باستخدام نمط ECB للتشفير. من أسباب ضرورة حماية IV الآتي: إذا استطاع الخَصْم خداع المستلم بحيث يَستخدم قيمة مختلفة لـ IV، فسيكون بوسع الخَصْم قلب (عكس) بتات مختارة في أول كتلة من الرسالة الأصلية التي يتلقاها المستلم، لإثبات ذلك، خذ بعين الاعتبار ما يأتي:

$$C_1 = \mathrm{E}(K, [\mathrm{IV} \oplus P_1])$$
$$P_1 = \mathrm{IV} \oplus \mathrm{D}(K, C_1)$$

باستخدام $X[j]$ للدلالة على البت j ضمن الـ 64 بتاً التي تضمها الكمية X يمكننا كتابة

$$P_1[i] = \mathrm{IV}[i] + \mathrm{D}(K, C_1)[i]$$

وباستخدام خصائص العملية XOR، يمكننا القول بأن:

$$P_1[i]' = \mathrm{IV}[i]' + \mathrm{D}(K, C_1)[i]$$

حيث تشير "′" لعملية قلب البت (bit complementation). يعني ذلك أنه إذا تمكن الخَصْم من قلب بتات بعينها في المتجه الأوّلي IV فسيتمكن بذلك من قلب (أي تغيير) البتات المناظرة في كتلة الرسالة الأصلية التي يسترجعها المستلم.

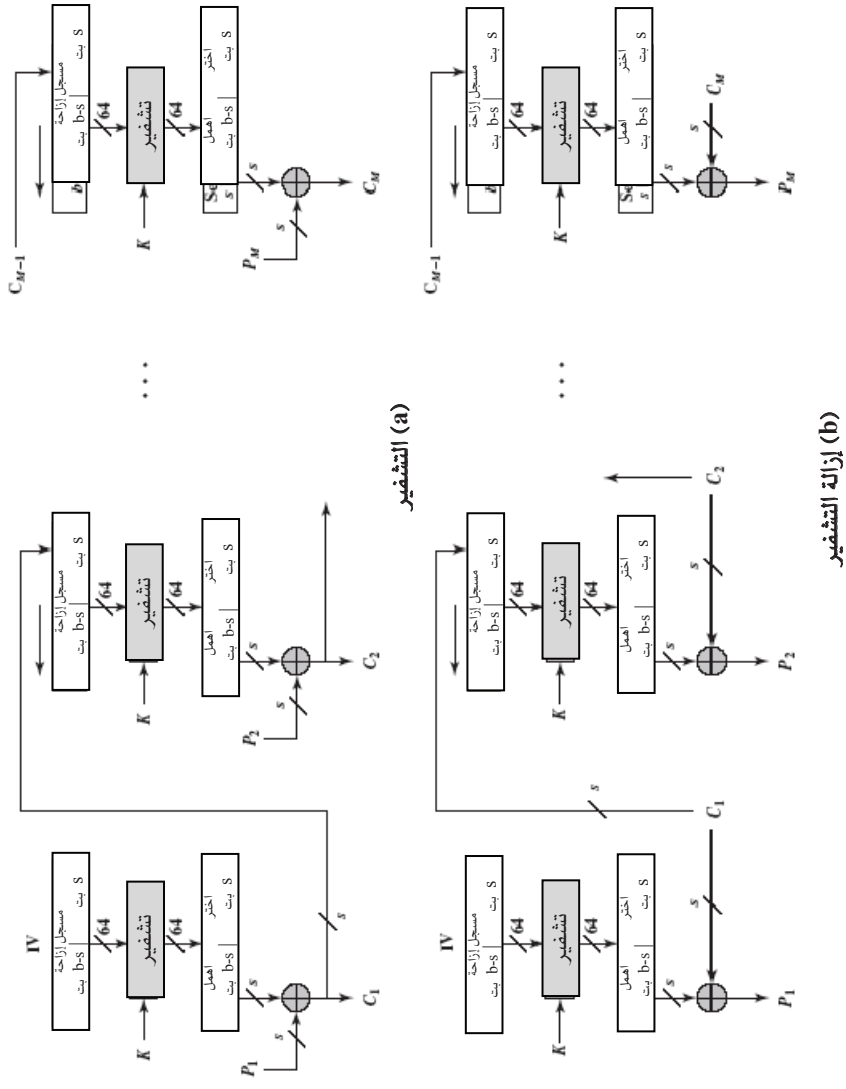
يُستخدم نمط CBC على نطاق واسع في تطبيقات الأمن، كما سنرى في الباب الثاني من الكتاب.

2-4-2 نمط التشفير بالتغذية المرتجعة

من الممكن تحويل أي خوارزمية تشفير كتلة إلى خوارزمية تشفير تدفق باستخدام نمط التشفير بالتغذية المرتجعة (Cipher Feedback (CFB) mode). باستخدام تشفير التدفق لا نحتاج لحشو الرسالة الأصلية لتصبح عدداً كاملاً من الكتل، كما أنه يمكن استخدامها للعمل في الوقت الحقيقي. وعليه، فعند إرسال سلسلة من الحروف، سيكون بالإمكان تشفير كل حرف وإرساله في الحال باستخدام نظام تشفير تدفق يعمل على الحروف.

من المزايا المرغوبة لنظام تشفير التدفق أن تكون الرسالة المشفَّرة بنفس طول الرسالة الأصلية. وعليه، فعند إرسال حروف يتألف كلٌّ منها من 8 بتات ينبغي تشفير كل حرف باستخدام 8 بتات. إذا استخدم النظام أكثر من 8 بتات فإن ذلك يمثل إهداراً لسعة الإرسال.

يمثل الشكل 2-10 نظام التشفير بنمط CFB. يُفترض في الشكل أن وحدة الإرسال هي s بت، ومن القيم الشائعة $s = 8$. كما في نمط CBC، تتم سَلسَلة وحدات بيانات الرسالة الأصلية معاً بحيث تعتمد الرسالة المشفَّرة لأي وحدة من الرسالة الأصلية على كل الوحدات السابقة للرسالة الأصلية.



الشكل 2-10: نمط التغذية المرتجعة للشفرة (CBF) بـ *s* بت.

خذ بعين الاعتبار أولاً عملية التشفير. يتألف المُدخل إلى خوارزمية التشفير من سِجِل إزاحة (shift register) طوله 64 بتاً يُملأ في البداية بمتجه أوّلي (IV). تُجرى عملية XOR بين الـ s بت في أقصى اليسار (الأعلى وزناً) من مُخرج دالة التشفير والوحدة الأولى من الرسالة الأصلية P_1 لإنتاج الوحدة الأولى من الرسالة المشفَّرة C_1 والتي يتم إرسالها. بالإضافة إلى ذلك، تتم إزاحة محتويات سِجِل الإزاحة s خانة إلى اليسار وتوضع C_1 في الـ s بت بأقصى اليمين (الأدنى وزناً) من سِجِل الإزاحة. وتستمر هذه العملية حتى يتم تشفير جميع وحدات الرسالة الأصلية .

للقيام بإزالة التشفير، تُتبع نفس الطريقة، غير أنه يتم إجراء عملية XOR بين الوحدة التي يتم استلامها من الرسالة المشفَّرة ومُخرج دالة التشفير لإنتاج وحدة من الرسالة الأصلية. لاحظ أن المُستخدَم هنا هو دالة التشفير وليس دالة إزالة التشفير. ويمكن تفسير ذلك بسهولة. افترض أن $S_s(X)$ ترمز للـ s بت الأعلى وزناً في X، ومن ثم فإن

$$C_1 = P_1 \oplus S_s[E(K, IV)]$$

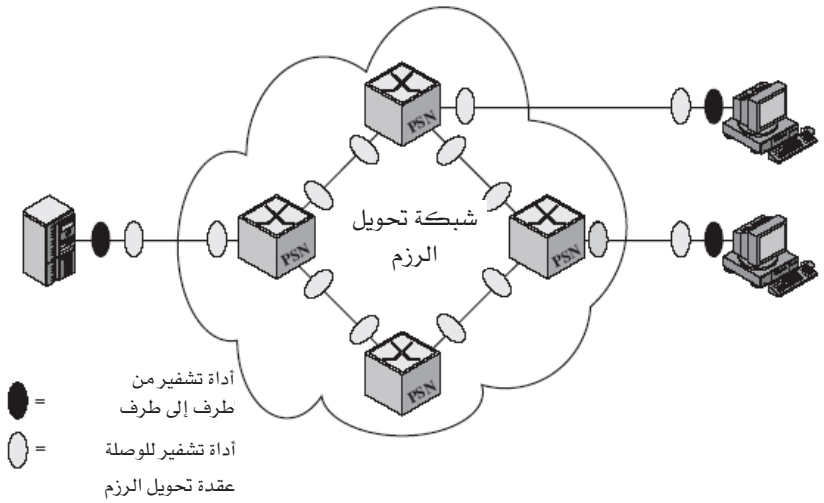
ولذلك فإن

$$P_1 = C_1 \oplus S_s[E(K, IV)]$$

وينطبق نفس المنطق على الخطوات التي تلي ذلك في تلك العملية.

5-2 مواضع أجهزة التشفير

يُعدُّ التشفير أقوى الأساليب وأكثرها شيوعاً لمواجهة الأخطار التي تهدد أمن الشبكات. وعند استخدام التشفير علينا أن نقرر ما الذي نشفره؟ وأين سنضع أجهزة التشفير؟ هناك بديلان أساسيان: تشفير الوصلة، والتشفير من طرف إلى طرف. يوضح الشكل 2-11 استخدام هذين الأسلوبين عبر شبكة تحويل الرزم.



الشكل 2-11: التشفير عبر شبكة تحويل الرزم.

في تشفير الوصلة تزوّد كل الوصلات المعرّضة للهجوم بأداة تشفير على كلٍّ من طرفي الوصلة، وبذلك يتم تأمين كل حركة مرور للبيانات على جميع وصلات الاتصال. ورغم أنه في شبكة كبيرة يتطلب ذلك الكثير من أجهزة التشفير، فإنه يوفر مستوى عالٍ من الأمن. ولكن من عيوب هذا الأسلوب أنه يتعين إزالة التشفير في كل مرة تدخل فيها الرسالة إلى محول رزم، وذلك كي يتسنى للمحول قراءة العنوان (رقم الدائرة الافتراضية) الموجود في ترويسة الرزمة حتى يتمكن من توجيهها باتجاه وجهتها النهائية. وهكذا تكون الرسالة عرضةً للهجوم عند كل محول رزم. واذا كانت الشبكة شبكة عامة لتحويل الرزم فلن يكون للمستخدم أي سيطرة على الأمن في عقد الشبكة.

في التشفير من طرف إلى طرف تتم عملية التشفير وإزالة التشفير في الأنظمة الطرفية. ويقوم المصدر أو الطرف المرسِل (مضيف أو محطة) بتشفير البيانات. ويتم إرسال البيانات بصيغتها المشفَّرة، ومن ثم تنتقل عبر الشبكة دون تغيير إلى وجهتها النهائية (محطة أو مضيف). يشترك طرف الوجهة النهائية مع طرف المصدر في مفتاح سري، ومن ثم يتسنى له إزالة تشفير البيانات. وقد يبدو أن هذا الأسلوب يؤمّن

انتقال البيانات ضد هجمات وصلات الشبكة ومحولاتها. ومع ذلك، فلا تزال هناك نقطة ضعف.

خذ الوضع التالي بعين الاعتبار. يرتبط مضيف بشبكة X.25 لتحويل الرزم، وينشئ دائرة إفتراضية إلى مضيف آخر، ويتأهب لنقل البيانات إلى ذلك المضيف باستخدام التشفير من طرف إلى طرف. وتُرسَل البيانات عبر تلك الشبكة على شكل رزم يتألف كلٌّ منها من ترويسة في المقدمة تتبعها بعضٌ من بيانات المستخدم. أي جزء من كل رزمة سيقوم طرف المصدر بتشفيره؟ وإذا افترضنا أن المصدر يقوم بتشفير الرزمة بكاملها، بما في ذلك الترويسة فلن تفلح هذه الطريقة، لأنه كما ذكرنا فإن طرف الوجهة فقط هو الذي يمكنه إزالة التشفير. لذا فستتلقى عقدة تحويل الرزم رزمة مشفَّرة وسيتعذر عليها قراءة الترويسة، ومن ثَمَّ لن تكون قادرة على توجيه الرزمة. ويترتب على ذلك أن طرف المصدر يمكنه تشفير الجزء من الرزمة الخاص ببيانات المستخدِم فقط، ويجب أن يترك الترويسة غير مشفَّرة بحيث يمكن قراءتها بواسطة الشبكة.

وهكذا، فباستخدام التشفير من طرف إلى طرف يتم تأمين بيانات المُستخدِم، أما نمط حركة البيانات فيبقى غير مؤمّن، حيث تنقل ترويسات الرزم واضحة بدون تشفير. لتحقيق قدر أكبر من الأمن، نحتاج إلى كلٍّ من تشفير الوصلات والتشفير من طرف إلى طرف، كما هو مبين في الشكل 2-11.

باختصار، عند استخدام كلٍّ من أسلوبي التشفير معاً، يقوم المضيف بتشفير جزء الرزمة الخاص ببيانات المستخدِم باستخدام مفتاح للتشفير من طرف إلى طرف. ويتم بعد ذلك تشفير الرزمة بكاملها باستخدام مفتاح لتشفير الوصلة. بينما تعبر الرزمة الشبكة انطلاقاً من المصدر إلى الوجهة يقوم كل محول رزم بإزالة تشفير الرزمة باستخدام مفتاح تشفير الوصلة لقراءة الترويسة ثم يعيد تشفير الرزمة بكاملها لإرسالها من جديد على الوصلة التالية. وبهذه الطريقة تُعدُّ الرزمة بكاملها مؤمّنة باستثناء الوقت الذي تكون فيه الرزمة بالفعل في ذاكرة محول الرزم، حيث تكون ترويسة الرزمة عندئذٍ غير مشفَّرة.

6-2 توزيع المفاتيح

لكي يُستخدم التشفير المتماثل بنجاح، ينبغي أن يشترك طرفا عملية تبادل البيانات في نفس المفتاح، وأن يتم توفير الحماية لذلك المفتاح من وصول الآخرين إليه. علاوة على ذلك، عادةً ما يُنصح بتغيير المفتاح من حين لآخر للحد من كمية البيانات التي قد تتعرض للانكشاف إذا استطاع المهاجمون معرفة المفتاح. لذا، فإن قوة أي نظام للتشفير ترتكز على أسلوب توزيع المفاتيح، وهو المصطلح الذي يشير إلى وسيلة توصيل المفتاح للطرفين اللذين يرغبان في تبادل البيانات، بدون السماح لغيرهما بالاطلاع على المفتاح. ويمكن تحقيق توزيع المفاتيح للطرفين A وB بعدد من الطرق منها:

1. يمكن أن يختار الطرف A مفتاحاً ويتم توصيل ذلك المفتاح مادياً إلى الطرف B.
2. يمكن أن يقوم طرف ثالث باختيار المفتاح وتوصيله مادياُ إلى كلٍّ من A وB.
3. إذا كان قد سبق لأيٍّ من A وB استخدام مفتاح جديد مؤخراَ، يمكن لأيٍّ من الطرفين إرسال المفتاح الجديد للطرف الآخر، مشفَّراً باستخدام المفتاح القديم.
4. إذا توافر لكلٍّ من A وB توصيلة مشفَّرة مع طرف ثالث C، يمكن للطرف C توصيل مفتاح على الوصلتين المشفَّرتين إلى كلٍّ من A وB.

يتطلب الخياران 1 و2 تسليم المفتاح يدوياً، الأمر الذي يُعدُّ مطلباً معقولاً في حالة تشفير الوصلة حيث ستقوم أداة التشفير على الوصلة فقط بتبادل البيانات مع الشريك على الطرف الآخر من الوصلة. أما فيما يتعلق بالتشفير من طرف إلى طرف، فإن التسليم اليدوي للمفاتيح يُعدُّ مزعجاً. في نظام موزّع، قد يحتاج أي مضيف أو طرف لتبادل بيانات مع العديد من الأنظمة المضيفة والأطراف الأخري. وعليه، فإن كل جهاز سيحتاج إلى عدد من المفاتيح يتم توفيرها بشكل ديناميكي وتكون هذه المشكلة أصعب بشكلٍ خاص في حالة نظام موزَّع يغطى منطقة واسعة.

الخيار 3 ممكن مع كلٍّ من تشفير الوصلة والتشفير من طرف إلى طرف، ولكن إذا حدث ونجح مهاجم في الوصول إلى أحد المفاتيح فسوف تنكشف كل المفاتيح التي تليه. حتى لو تم تغيير مفاتيح تشفير الوصلة بصورة متكررة، فينبغي أن يتم ذلك يدوياً. لتوفير المفاتيح للتشفير من طرف إلى طرف، يفضل استخدام الخيار 4.

يوضح الشكل 2-12 طريقة لتحقيق متطلبات الخيار 4 للتشفير من طرف إلى طرف. في هذا الشكل، تم إهمال تشفير الوصلة، والذي يمكن إضافته أو عدم إضافته حسب الحاجة. في هذه الترتيبة يمكن تحديد نوعين من المفاتيح:

- مفتاح الجلسة: عندما يرغب نظامان طرفيان (مثلاً: من المضيفات أو المحطات الطرفية) في الاتصال فيما بينهما، فإنهما يقيمان توصيلة منطقية (مثلاً: دائرة افتراضية). يتم تشفير كل بيانات المستخدِم بمفتاح جلسة يُعطَى مرة واحدة طوال فترة وجود تلك التوصيلة المنطقية، وفي ختام الجلسة أو التوصيلة يتم التخلص من المفتاح.
- المفتاح الدائم: وهو مفتاح للاستخدام بين الكيانات بغرض توزيع مفاتيح الجلسات.

تتألف الترتيبة من العناصر الآتية:

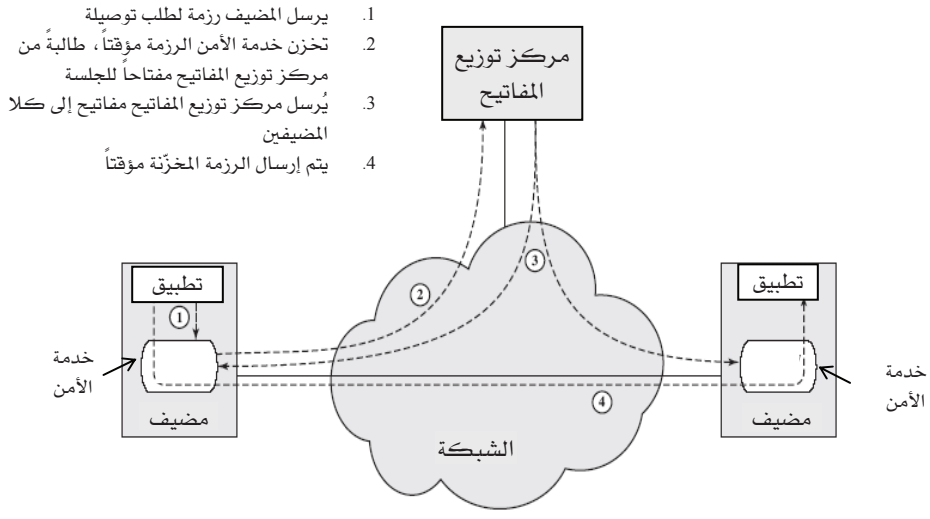
- مركز توزيع المفاتيح: يقوم مركز توزيع المفاتيح (Key Distribution Center (KDC)) بتحديد أي النظم يُسمح لها بالاتصال مع بعضها بعضاً. عند التصريح لنظامين بإنشاء توصيلة، يوفر مركز توزيع المفاتيح مفتاح جلسة لتلك التوصيلة للاستخدام لمرة واحدة فقط.
- وحدة خدمة الأمن (Security Service Module (SSM)): تقوم هذه الوحدة، والتي قد تتكون من وظائف في بروتوكول طبقة واحدة، بعملية التشفير من طرف إلى طرف وتحصل على مفاتيح الجلسات نيابة عن المستخدِمين.

يوضح الشكل 2-12 الخطوات اللازمة لإنشاء توصيلة. عندما يرغب مضيف في إنشاء توصيلة مع مضيف آخر، فإنه يقوم بإرسال رزمة طلب توصيلة (الخطوة 1).

تقوم وحدة SSM بتخزين تلك الرزمة وتتقدم إلى مركز KDC بطلب إنشاء توصيلة (الخطوة 2). يتم تشفير الاتصال بين SSM وKDC باستخدام مفتاح رئيس يشترك فيه فقط وحدة SSM تلك ومركز KDC. إذا وافق مركز KDC على طلب التوصيلة، فإنه يولد مفتاح جلسة ويوصّله إلى كلٍّ من وحدتي SSM المعنيتين، وذلك باستخدام مفتاح دائم مخصص لكل وحدة (الخطوة 3). تقوم الآن وحدة SSM التي طلبت التوصيلة بسحب رزمة طلب التوصيلة، ومن ثم يتم إنشاء توصيلة بين النظامين الطرفيين (الخطوة 4). يتم تشفير كل البيانات المتبادلة بين النظامين الطرفيين بواسطة وحدة SSM الخاصة بكلٍّ منهما وذلك باستخدام مفتاح الجلسة لمرة واحدة.

توفّر الآلية الأتوماتيكية لتوزيع المفاتيح مزايا المرونة والديناميكية اللازمة للسماح لعدد من المُستخدِمين من محطات طرفية بالوصول إلى عدد من المضيفين، وكذلك للمضيفين بتبادل البيانات بين بعضهم بعضاً.

هناك طريقة أخرى لتوزيع المفاتيح تستخدم التشفير بالمفاتيح العامة، وسوف نرجئ مناقشتها إلى الفصل الثالث.



الشكل 2-12: التوزيع الأتوماتيكي للمفاتيح لبروتوكول مبنيٍّ على استخدام توصيلة.

7-2 توصيات للمطالعة

يوفّر [STAL06a] تغطيةً أكثر عمقاً للمواضيع المدرجة في هذا الفصل. يُعدّ [SCHN96] مرجعاً أساسياً لتغطية خوارزميات التشفير، حيث يتضمَّن وصفاً لكل خوارزمية أو بروتوكول تشفير نُشرت حتى وقت كتابة ذلك الكتاب تقريباً. يوجد مسح تفصيلي آخر جدير بالاهتمام في [MENE97]. يوفر [STIN06] معالجة أكثر عمقاً مع مناقشة رياضية محكمة.

[FEIS73] Feistel, H. "Cryptography and Computer Privacy." *Scientific American*, May 1973.

[MENE97] Menezes, A.; van Oorschot, P.; and Vanstone, S. *Handbook of Applied Cryptography*. Boca Raton, FL: CRC Press, 1997.

[SCHN96] Schneier, B. *Applied Cryptography*. New York: Wiley, 1996.

[STAL06a] Stallings, W. *Cryptography and Network Security: Principles and Practice*, Fourth Edition. Upper Saddle River, NJ: Prentice Hall, 2006.

[STIN06] Stinson, D. *Cryptography: Theory and Practice*. Boca Raton, FL: CRC Press, 2006.

8-2 مصادر للمعلومات على الويب

- صفحة AES على الويب: تتضمَّن صفحة AES على موقع NIST وثيقة المعيار بالإضافة إلى عدد من الوثائق الأخرى ذات الصلة.
- صالون AES على الويب: يتضمَّن قوائم شاملة للمصادر تحتوي على الوثائق والأبحاث عن AES مع إمكانية للحصول على نسخ إلكترونية منها.
- أنماط تشغيل أنظمة تشفير الكتلة: صفحة على موقع NIST تتضمَّن معلومات وافية عن أنماط التشغيل المعتمدة من قِبَل NIST.

9-2 مصطلحات رئيسة

Advanced Encryption Standard (AES)	معيار التشفير المتقدم (AES)
block cipher	تشفير الكتلة
brute-force attack	الهجوم الاستقصائي
Cipher Block Chaining (CBC) mode	نمط سلسلة كتل الشفرة (CBC)
Cipher FeedBack (CFB) mode	نمط التغذية المرتجعة للشفرة (CFB)
ciphertext	رسالة مشفَّرة
cryptanalysis	تحليل الشفرة
Data Encryption Standard (DES)	معيار تشفير البيانات (DES)
decryption	إزالة التشفير
Electronic CodeBook (ECB) mode	نمط دفتر الشفرة الإلكتروني (ECB)
encryption	التشفير
end-to-end encryption	التشفير من طرف إلى طرف
Feistel cipher	هيكل فيستل للتشفير
key distribution	توزيع المفاتيح
link encryption	تشفير الوصلة
plaintext	الرسالة الأصلية
session key	مفتاح الجلسة
stream cipher	تشفير التدفق
subkey	المفتاح الفرعي
symmetric encryption	التشفير المتماثل
Triple DES (3DES)	خوارزمية DES الثلاثية (3DES)

10-2 أسئلة للمراجعة ومسائل

1-10-2 أسئلة للمراجعة

2-1 ما المكونات الأساسية لنظام التشفير المتماثل؟

2-2 ما الوظيفتان الأساسيتان المستخدَمتان في خوارزميات التشفير؟

2-3 كم عدد المفاتيح اللازمة لكي يتمكن شخصان من الاتصال عبر نظام للتشفير المتماثل؟

2-4 ما الفرق بين تشفير الكتلة وتشفير التدفق؟

2-5 ما الأسلوبان العامان للهجوم على شفرة بهدف فكها؟

2-6 لماذا تستخدم بعض أنماط تشفير الكتل التشفير فقط، بينما يَستخدم البعض الآخر كلاً من التشفير وإزالة التشفير؟

2-7 عرِّف المقصود بالتشفير الثلاثي.

2-8 لماذا يشتمل الجزء الأوسط من خوارزمية 3DES على إزالة التشفير وليس إعادة التشفير؟

2-9 ما الفرق بين تشفير الوصلة والتشفير من طرف إلى طرف؟

2-10 اذكر الطرق التي يمكن استخدامها لتوزيع المفاتيح السرية على طرفين للاتصال المشفَّر باستخدامها.

2-11 ما الفرق بين مفتاح الجلسة والمفتاح الرئيس؟

2-12 عرِّف المقصود بمركز توزيع المفاتيح.

2-10-2 مسائل

2-1 وضّح أن عملية إزالة تشفير فيستل هي عكس عملية تشفير فيستل.

2-2 ما قيمة مفتاح RC4 الذي يترك قيمة S بدون تغيير خلال عملية التهيئة الأولية؟ بمعنى أنه بعد التبديلة الأولية لـ S تكون عناصر S تساوي القيم من 0 إلى 255 بترتيب تصاعدي.

2-3 لخوارزمية RC4 حالة داخلية سرية هي تبديلة من كل القيم المحتملة للمتجه S والمؤشرين i وj.

a. كم عدد البتات اللازمة لتخزين تلك الحالة الداخلية باستخدام طريقة تخزين مباشرة؟

b. لنفترض أننا سننظر إلى الأمر من وجهة نظر كمية المعلومات التي تمثلها الحالة. عندئذٍ سنحتاج إلى تحديد عدد الحالات المختلفة الموجودة، ثم نحسب لوغاريثم ذلك العدد للأساس 2 لتعيين عدد بتات المعلومات المطلوبة لتمثيل ذلك. باستخدام هذه الطريقة، كم بتاً سنحتاجها لتمثيل الحالة؟

2-4 في نمط دفتر الشفرة الإلكتروني (ECB)، إذا حدث خطأ في كتلة من كتل الرسالة المشفَّرة أثناء إرسالها، فستتضرر نتيجةً لذلك الكتلة المناظرة في الرسالة الأصلية فقط. أما في نمط سَلسَلة كتل الشفرة (CBC)، فسينتقل ذلك الخطأ. فمثلاً وجود خطأ في كتلة C_1 المرسَلة (في الشكل 2-9) سيُفسد أيضاً كلاً من الكتلتين P_2 وP_1.

a. هل تتضرر أي كتل أبعد من P_2؟

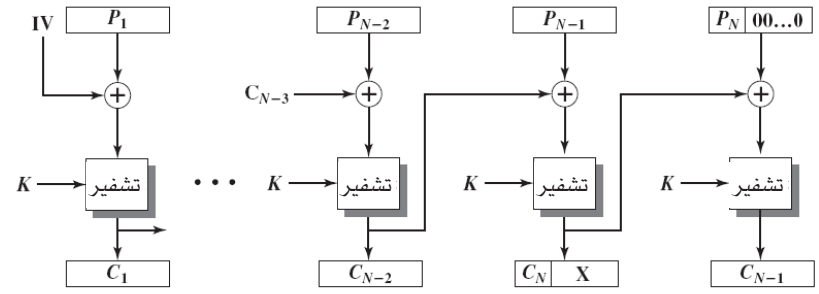
b. لنفترض وجود خطأ في أحد البتات بالنسخة المصدرية للكتلة P_1. كم عدد كتل الرسالة المشفَّرة التي سينتقل إليها ذلك الخطأ؟ ما تأثير ذلك عند المستلِم؟

2-5 حشو CBC هو نمط لتشغيل نظام تشفير الكتلة يُستخدم بالفعل في خوارزمية RC5، ولكن يمكن استخدامه أيضاً في أي نظام لتشفير الكتلة. ويمكن لحشو CBC التعامل مع رسالة أصلية بأي طول. تكون الرسالة المشفَّرة أطول من الرسالة الأصلية بكتلة واحدة كحد أقصى. يُستخدم الحشو للتأكد من أن طول الرسالة الأصلية من مضاعفات مقاس الكتلة. يُفترض أن طول الرسالة الأصلية غير المشفَّرة هو عدد صحيح من البايتات. يتم حشو تلك الرسالة الأصلية في نهايتها بعدد من البايتات يتراوح من 1 إلى bb بايت، حيث bb تساوي حجم الكتلة بالبايتات. تأخذ كل البايتات المحشوة نفس القيمة، وهي عدد البايتات التي تم حشوها. على سبيل المثال، إذا تم حشو 8 بايتات فستتضمَّن كل بايت حشو نمط البتات 00001000. لماذا لا يُسمح باستخدام حشوة صفرية؟ أي إذا كان طول الرسالة الأصلية من مضاعفات حجم الكتلة، فلماذا لا نمتنع عن الحشو كليةً في هذه الحالة؟

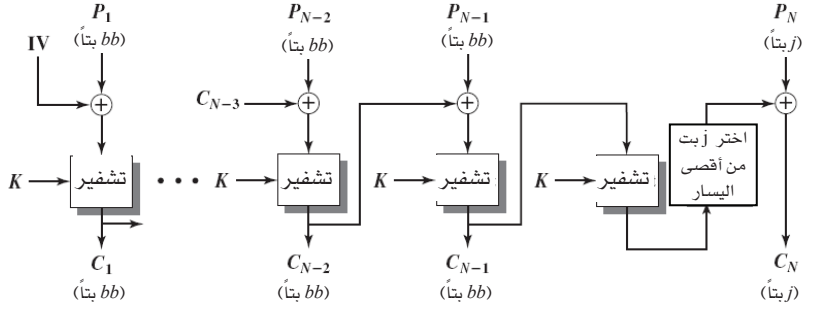
2-6 قد لا يكون أسلوب الحشو مناسباً في كل الحالات. على سبيل المثال، قد نرغب في تخزين الرسالة المشفَّرة في نفس الذاكرة المؤقتة التي كانت تحوي الرسالة الأصلية. وفي هذه الحالة، لا بد من أن تكون الرسالة المشفَّرة بنفس طول الرسالة الأصلية. هناك نمط تشغيل لتحقيق ذلك الغرض باختزال الرسالة المشفَّرة يُعرَف بـ ciphertext stealing (CTS). ويوضح الشكل 2-13 (a) طريقة لتنفيذ ذلك الأسلوب.

a. اشرح كيفية عمل تلك الطريقة.

b. صف كيف يمكن إزالة تشفير كلٍّ من الكتلتين C_n و C_{n-1}.



(a) نمط اختزال الرسالة المشفرة



(b) طريقة بديلة

الشكل 2-13 أنماط استخدام تشفير الكتلة مع رسالة أصلية بطول ليس من مضاعفات حجم الكتلة.

7-2 يبيّن الشكل 2-13(b) طريقة بديلة لأسلوب CTS لإنتاج رسالة مشفَّرة مساوية في الطول للرسالة الأصلية عندما لا يكون طول الأخيرة من مضاعفات حجم الكتلة.

a. اشرح الخوارزمية.

b. اشرح لماذا تفضّل طريقة CTS على هذه الطريقة المبينة في الشكل 2-13(b).

8-2 إذا حدث خطأ في أحد البتات أثناء نقل حرف من رسالة مشفَّرة باستخدام نمط التشغيل CFB بـ 8 بتات. إلى أي مدى سينتقل الخطأ؟

9-2 تعاني نظم توزيع المفاتيح التي تستخدم مركزاً للتحكم في الوصول و/أو مركزاً لتوزيع المفاتيح من وجود نقاط مركزية تكون عرضة للهجوم. ناقش الآثار الأمنية المترتبة على هذه المركزية في تلك النظم.

10-2 لنفترض أن شخصاً ما يقترح الطريقة التالية للتأكد من أن لدى كلٍّ واحد منكما نفس المفتاح السري. تقوم أنت بتوليد سلسلة بتات عشوائية بطول المفتاح، ثم تُجري

عملية XOR بينها وبين المفتاح، وترسل النتيجة عبر قناة الاتصال. يقوم شريكك بإجراء عملية XOR بين الكتلة التي يستلمها والمفتاح (والذي يجب أن يكون هو نفس مفتاحك) ويرسل الناتج إليك. تقوم أنت بالتحقق، فإذا وجدت أنك استلمت منه نفس سلسلة البتات العشوائية التي استخدمتها في البداية فستكون قد أثبتّ أن شريكك لديه نفس المفتاح السري، مع أنه لم يقم أيٌّ منكما بإرسال المفتاح نفسه على القناة. هل هناك عيب في هذه الخطة؟

الفصل الثالث

3 التشفير بالمفاتيح العامة وتوثيق الرسائل

محتويات الفصل:

لكل مصري اسمان: اسم حقيقي واسم شهرة، أو اسم خاص واسم عام على التوالي. وعادة ما يُعلَن اسم الشهرة أو الاسم العام للملأ، في حين يُفصَح عن الاسم الحقيقي أو الاسم الخاص بحذر.

ـ من كتاب "الغصن الذهبي" للسيد جيمس جورج فريزر.

تقتضي الحكمة الحذر عند التعامل مع الغرباء للوقاية من التأثير الوخيم الذي قد ينجم عن ممارساتهم. ولذا قبل أن يُسمَح لهم بالدخول إلى حي، أو على الأقل قبل أن يُسمَح لهم بالاختلاط بحرية مع سكان الحي، يقوم السكان الأصليون في كثير من الأحيان بطقوس معينة لنزع القوى السحرية من الغرباء أو ـ إذا جاز التعبير ـ لتنقية الجو الملوث الذي يُفترَض أن يكونوا محاطين به.

ـ من كتاب "الغصن الذهبي" للسيد جيمس جورج فريزر.

بالإضافة إلى سرية الرسائل، يُعدُّ توثيق الرسائل (التحقق منها) من الوظائف المهمة لأمن الشبكة. ويبحث هذا الفصل ثلاثة جوانب لتوثيق الرسائل. أولاً: سننظر في استخدام أكواد ودوال التحوير لتوثيق الرسائل، بعدها سنلقي نظرة على المبادئ الأساسية للتشفير بالمفاتيح العامة وعلى خوارزميتين محددتين للتشفير بالمفاتيح العامة. وتُستخدَم تلك الخوارزميات في تبادل مفاتيح التشفير التقليدية. ثم سننظر في استخدام التشفير بالمفاتيح العامة لتوليد توقيعات رقمية للرسائل توفر طريقة محسَّنة لتوثيقها. وأخيراً سنعاود النظر في مسألة إدارة مفاتيح التشفير.

1-3 طرق توثيق الرسائل

يحمي التشفير ضد الهجوم السلبي (التنصت). لكن هناك مطلب آخر وهو الحماية من الهجوم النشط (تزوير البيانات والعمليات). وتُعرَف الحماية من مثل هذه الهجمات بتوثيق الرسائل أو التحقق منها. ويقال أن رسالةً أو ملفاً أو وثيقةً من الوثائق أو غير ذلك من تجمعات البيانات موثَّقة عندما تكون حقيقية ومرسلة من المصدر الذي يدَّعي إرسالها. ويُعرَّف توثيق الرسالة بأنه إجراء يتيح