7

الفصل السابع

أمن شبكة الويب

محتويات الفصل:

"استخدم عقلك ...واستيقظ على الحقيقة" ـ من أغنية "استحوذت عليك تحت جلدي" ، لكول بورتر

النقاط الرئيسة

- يوفر بروتوكول طبقة المقابس الآمنة (SSL) خدمات الأمن للتطبيقات التي تستخدم بروتوكول TCP. أما النسخة القياسية الخاصة بالإنترنت فيطلق عليها "أمن طبقة النقل" (TLS). ونظراً لتقاربهما ، فسنشير إليهما على أنهما بروتوكول واحد SSL/TLS إلا في حالة وجود اختلاف.
- يوفر SSL/TLS السرية عن طريق استخدام التشفير المتماثل، ويوفر سلامة الرسائل باستخدام كود توثيق الرسالة (MAC).
- يتضمّن SSL/TLS آليات لتمكين مستخدمي TCP من تحديد طرق الأمن وخدماته التي سيستخدمونها.
- بروتوكول المُعامَلات الإلكترونية الآمنة (SET) هو معيار تشفير وأمن مفتوح صُمِّم لحماية استخدام بطاقات الائتمان على الإنترنت.

توجد مواقع ويب لكل الشركات التجارية تقريباً الآن وكذلك لأغلب الأجهزة الحكومية والعديد من الأفراد. يتزايد عدد الأفراد والشركات المتصلين بالإنترنت بسرعة، ولدى كلٍّ منهم متصفح ويب (Web browser) بواجهة مستخدم رسومية (GUI). مما جعل عدداً كبيراً من الشركات تتحمس لوضع الوسائل اللازمة على شبكة الويب لتسهيل التجارة الإلكترونية عبر الإنترنت. لكن تبقى الحقيقة الماثلة أن الإنترنت والويب يتعرضان بشكلٍ قوي لتشكيلة كبيرة من الهجمات.

ومع تنبه الشركات لهذه الحقيقة، ازداد الطلب على توفير خدمات ويب آمنة. وهذا الموضوع – أمن الويب – موضوع واسع، ويمكن أن يُشكِّل بسهولة كتاباً بكامله (للمزيد يمكنك الاطلاع على الكتب الموصَّى بها في نهاية هذا الفصل).

سنبدأ هذا الفصل بمناقشة المتطلبات العامة لأمن الويب، وبعد ذلك سنركز على معيارين ازدادت أهميتهما بشكلٍ كبير كجزء مهم من بنية نظم التجارة عبر الإنترنت: SSL/TLS، وSET.

1-7 متطلبات أمن الويب

إن شبكة المعلومات العالمية (والمعروفة بالويب) ليست في جوهرها إلا تطبيق "زبون/خادم" يعمل على شبكات الإنترنت والإنترانت (أي شبكات TCP/IP الداخلية). ولذا، فإن أدوات الأمن وأساليبه المختلفة التي ناقشناها حتى الآن في هذا الكتاب تصلح لتأمين الويب. لكن، كما أشار [GARF 97]، هناك مجموعة من التحديات الجديدة التي يقدمها الويب ولم تعالج بشكلٍ عام في سياق أمن الشبكة والحاسب؛ منها:

- يُعدُّ الويب ــ على خلاف بيئات النشر التقليدية أو حتى أنظمة النشر الإلكترونية الأخرى (كقنوات المعلومات النصية، وأنظمة الردّ الصوتي، ونظام إرسال الفاكسات واستقبالها عن طريق الحاسب) ــ عرضةً للكثير من الهجمات على خادمات الويب من خلال الإنترنت.
- يعمل الويب على نحو متزايد كواجهة مرئية للشركة ومنتجاتها، وأيضاً كمنصة لإجراء مُعامَلات تجارية. لكن يمكن أن تدمّر السمعة وتضيع الأموال إذا تعرضت خادمات الويب للتخريب.
- رغم سهولة استخدام متصفحات الويب وإعداد خادمات الويب وإدارتها وتطوير محتوى الويب، فإن برمجيات البنية التحتية معقدةٌ جداً. وقد يؤدى هذا التعقيد في البرمجيات إلى إخفاء كثير من النقاط المحتمل ضعفها أمنياً. إن تاريخ الويب القصير مليء بالأمثلة لأنظمة جديدة وأنظمة مُرقَّاة تم إعدادها بشكلٍ شرعي لكنها تتضمن نقاطاً عرضةً لكثير من الهجمات.
- يمكن أن يُستغَل خادم الويب كقاعدة لشن هجوم على جميع أجهزة الحاسب بالشركة أو المنظمة. عندما يتعرض خادم الويب للتخريب، يمكن

أن يصبح المهاجم قادراً على الدخول إلى أنظمة البيانات والأنظمة التي ليست جزءاً من الويب نفسه لكنها مرتبطة بالموقع المحلي للخادم.

- أكثر مستخدمي خدمات الويب أشخاصٌ عاديون وغير مدرَّبين على القضايا الأمنية، وهؤلاء لا يُدركون بالضرورة الأخطار الأمنية المُحدقة، كما أنهم لا يمتلكون الأدوات أو المعرفة لمواجهة تلك الأخطار بشكلٍ فعالٍ.

1-1-7 تهديدات أمن الويب

يتضمن الجدول 7-1 ملخصاً بالتهديدات الأمنية المختلفة التي يمكن التعرض لها عند استخدام الويب. يمكن تصنيف تلك التهديدات من حيث كونها هجمات سلبية أو هجمات نشطة. وتتضمّن الهجمات السلبية التنصت على حركة مرور البيانات على الشبكة بين المتصفح والخادم، والتمكّن من الوصول لمعلومات موجودة على موقع الويب يفترض أنها مقيّدة الاستخدام. أما الهجمات النشطة فتتضمّن تقمص شخصية مستخدم آخر، وتعديل الرسائل أثناء انتقالها بين الزبون والخادم، وتعديل المعلومات على موقع الويب.

من الطرق الأخرى لتصنيف تهديدات أمن الويب تصنيفها حسب موقع مصدر التهديد: هل هو خادم الويب، أم متصفح الويب، أم حركة مرور بيانات الشبكة بين المتصفح والخادم؟. تقع قضايا أمن الخادم والمتصفح ضمن أمن نظام الحاسب التي يناقشها بشكلٍ عام الباب الثالث من هذا الكتاب. أما قضايا أمن حركة مرور البيانات فتُصنَّف ضمن أمن الشبكة وسنناقشها في هذا الفصل.

2-1-7 طرق تأمين حركة مرور بيانات الويب

يمكن استخدام عدة طرق لتوفير أمن الويب. تتشابه الطرق المختلفة التي اقتُرحَت بدرجة كبيرة في الخدمات التي توفرها وإلى حدٍّ ما في الآليات التي

تستخدمها، لكنها تختلف من حيث نطاق التطبيق وموقعها ضمن رصَّة بروتوكولات TCP/IP.

يبيّن الشكل 7-1 هذا الاختلاف. يمكن توفير أمن الويب من خلال تأمين بروتوكول IP باستخدام IPSec (الشكل 7-1 (a)). وتمتاز هذه الطريقة بأنها تتم دون الحاجة إلى تدخلٍ من مستخدمي الأنظمة الطرفية على الشبكة أو تطبيقاتها، كما أنها توفر حلاً عاماً. يتضمّن IPSec أيضاً إمكانيةً للترشيح (filtering) كي تقتصر المعاناة من عبء المعالجة الإضافي على نوعية محددة فقط من حركة مرور البيانات.

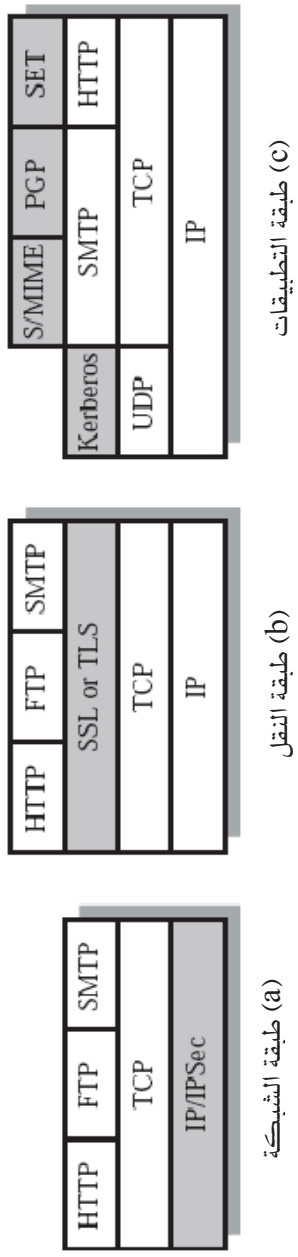
هناك حلٌّ آخرٌ عامٌ نسبياً وهو وضع إجراءات الأمن فوق TCP مباشرةً (انظر الشكل 7-1 (b)). وإن أفضل مثال لهذه الطريقة هو بروتوكول طبقة المقابس الآمنة (Secure Socket Layer (SSL)) ومعيار الإنترنت الذي تلاه والمعروف باسم أمن طبقة النقل (Transport Layer Security (TLS)). ويمكن تحقيق خدمات الأمن في هذا المستوى بأحد أسلوبين: توفير بروتوكول SSL (أو TLS) كجزء من رصَّة البروتوكولات التحتية وبالتالي لا تشعر التطبيقات بوجوده، البديل الآخر هو تضمين SSL في حزم برمجيات معينة.

على سبيل المثال، يأتي متصفح نيتسكيب ومتصفح مستكشف مايكروسوفت مزوداً ببروتوكول SSL، وكذلك تتضمَّن أكثر خادمات الويب ذلك البروتوكول. لكن خدمات الأمن الخاصة بتطبيق معين تكون متضمَّنة مع ذلك التطبيق نفسه. ويبيّن الشكل 7-1 (c) أمثلة لتلك البنية المعمارية. وتكمن ميزة هذه الطريقة في القدرة على تفصيل الخدمة لتلبي احتياجات معيّنة للتطبيق. ومن الأمثلة المهمة لتلك الطريقة - ضمن سياق أمن الويب - بروتوكول المُعامَلات الإلكترونية الآمنة (SET).

سنكرس بقية هذا الفصل لمناقشة SSL/TLS وSET.

الجدول 7-1: مقارنة التهديدات الأمنية على الويب [RUBI97].

	التهديدات	العواقب	الأساليب المضادة
سلامة البيانات	- تعديل بيانات المستخدم - متصفح حصان الطروادة - تعديل محتويات الذاكرة - تعديل حركة مرور الرسائل العابرة	- فقد معلومات - اختراق الجهاز - التعرض لجميع التهديدات الأخرى	المجموع التدقيقي المُشفَّر
السرية	- التجسس على الشبكة - سرقة معلومات من الخادم - سرقة بيانات من الزبون - تسرب معلومات إعدادات الشبكة - تسرب معلومات عن الزبائن الذين يتصلون بالخادم	- فقد معلومات - فقد الخصوصية	التشفير، خادمات وكيل الويب
حجب الخدمة	- إنهاء عمليات المستخدم قبل الأوان - غمر الجهاز بالطلبات المزيَّفة - ملء الذاكرة أو القرص الصلب (إهدارهما فيما لا طائل منه) - عزل الجهاز بمهاجمة DNS	- عرقلة الخدمة - مضايقة المستخدمين - منع المستخدم من أداء عمله (ومن ثَمَّ تؤثر على الإنتاجية)	من الصعب منعها
التوثيق	- تقمص شخصيات أخرى - تزييف البيانات	- إساءة تمثيل المستخدم - اعتقاد أن المعلومات المزيفة صحيحة	أساليب التشفير



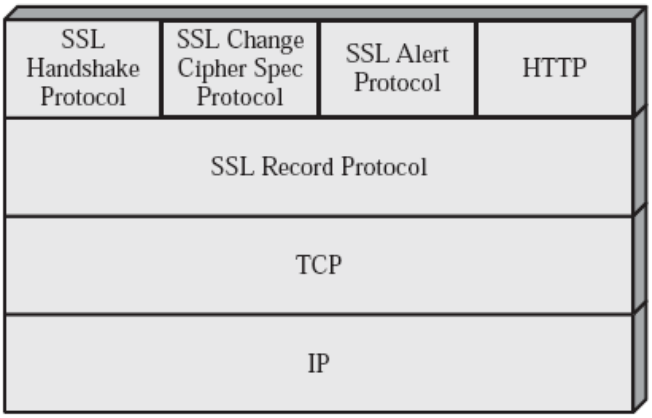
الشكل 7-1: مواقع وسائل الأمن ضمن رصَّة بروتوكولات TCP/IP.

2-7 طبقة المقابس الآمنة وأمن طبقة النقل

قامت شركة نيتسكيب بتطوير بروتوكول SSL. صُمِّمت النسخة 3 من هذا البروتوكول بعد مراجعة من مجتمع الإنترنت وملاحظات الصناعة ونُشِرت كمسوّدة إنترنت أولية. بعد ذلك، وعندما تم التوصل إلى إجماع لإصدار معيار موحّد للإنترنت، شُكِّلت مجموعة عمل خاصة ضمن فريق عمل هندسة الإنترنت (IETF) لتطوير معيار موحّد. أُطلق على هذه النسخة TLS، ويمكن اعتبارها في جوهرها النسخة 3.1 من بروتوكول SSL (SSLv3.1) وهي قريبة جداً ومتوافقة مع SSLv3. سنكرِّس معظم هذا الجزء لمناقشة SSLv3. وفي نهاية الجزء سنوضح الاختلافات الرئيسة بين SSLv3 وTLS.

1-2-7 بنية SSL

صُمِّم SSL بحيث يستخدم TCP لتقديم خدمة آمنة موثوقة من طرف إلى طرف. جديرٌ بالذكر أن SSL ليس بروتوكولاً واحداً وإنما يتألف في الواقع من طبقتين من البروتوكولات كما هو مبين في الشكل 7-2. يوفر بروتوكول سجل SSL (SSL Record Protocol) خدمات الأمن الأساسية لبروتوكولات الطبقات الأعلى المختلفة. وبشكلٍ خاص، يعمل بروتوكول HTTP ــ الذي يوفر خدمة النقل بين زبون وخادم الويب ــ فوق بروتوكول SSL. توجد ثلاثة بروتوكولات للطبقات العليا مُعرَّفة كجزءٍ من SSL: بروتوكول المصافحة (Handshake Protocol)، وبروتوكول تغيير مواصفات التشفير (Change Cipher Spec Protocol)، وبروتوكول التنبيه (Alert Protocol). وتُستخدَم تلك البروتوكولات الثلاث في إدارة تبادلات SSL (SSL exchanges)، وسنناقشها لاحقاً في هذا الجزء.



الشكل 7-2: رصّة بروتوكولات SSL.

من مفاهيم SSL المهمة: مفهوم توصيلة SSL ومفهوم جلسة SSL؛ وهما مُعرَّفان في مواصفات البروتوكول كالتّالي:

- التوصيلة (Connection): هي وسيلة نقل (في مصطلحات نموذج OSI) توفر نوعاً مناسباً من الخدمة. تمثل توصيلات SSL علاقات نظير لنظير، وهذه التوصيلات مؤقتة. حيث ترتبط كل توصيلة بجلسة واحدة.
- الجلسة (Session): تمثل جلسة SSL ارتباطاً بين زبون وخادم. ويتم إنشاء الجلسات عن طريق بروتوكول المصافحة. وتُعرِّف الجلسات مجموعة مُعامِلات الأمن الخاصة بالتشفير التي يمكن أن تشترك فيها عدة توصيلات. تُستخدم الجلسات لتفادي التفاوضات المكلِّفة حول مُعامِلات أمن جديدة لكل توصيلة.

قد يوجد بين كل زوج من التطبيقات (مثلاً HTTP على الزبون والخادم) عدة توصيلات آمنة. نظرياً، قد يكون هناك أيضاً عدة جلسات بين تلك الأطراف في نفس الوقت، ولكن هذه الخاصية لا تُستخدم عملياً. ويرتبط بكل جلسة في الواقع عددٌ من أوضاع (حالات) التشغيل. وبمجرد إنشاء الجلسة، تكون في حالة تشغيل

حالية تسمح بالقراءة والكتابة (أي الإرسال والاستقبال). كما أنه أثناء عملية المصافحة، يتم إنشاء حالات معلَّقة للقراءة والكتابة. وعند الانتهاء من بروتوكول المصافحة بنجاح، تصبح الحالات المعلَّقة حالات حالية. وتُعرَّف حالة الجلسة بالمُعامِلات الآتية (التعريفات مقتبسة من مواصفات SSL):

- مُعرِّف الجلسة: سلسلة من البايتات الاعتباطية يختارها الخادم لتعريف حالة جلسة نشطة أو قابلة للاستئناف.
- شهادة النظير: وهي شهادة X509.v3. ويمكن أن يأخذ القيمة null (لا يوجد).
- أسلوب ضغط البيانات: يحدد الخوارزمية المستخدمة لضغط البيانات قبل التشفير.
- مواصفات التشفير: يحدد خوارزمية تشفير كتل البيانات (مثلاً AES أو لا يوجد) وخوارزمية التحوير (مثلاً MD5 أو SHA-1) المستخدمة لحساب كود توثيق الرسالة (الماك). كما أنه يُعرِّف أيضاً خواص التشفير مثل طول كود التحوير.
- السرّ الرئيس: وهو سرٌ مشترك بين الزبون والخادم يتكون من 48 بايتاً.
- عَلَم يشير إلى مدى إمكانية استخدام الجلسة لبدء توصيلات جديدة.

تُعرَّف حالة التوصيلة بالمُعامِلات الآتية:

- رقم عشوائي لكلٍّ من الخادم والزبون: وهو سلسلة بايتات يختارها كلٌّ من الخادم والزبون لكل توصيلة.
- سرّ كود توثيق الرسالة للكتابة لدى الخادم: المفتاح السرّي المستخدم في عمليات الماك على البيانات التي يرسلها الخادم.
- سرّ كود توثيق الرسالة للكتابة لدى الزبون: المفتاح السرّي المستخدم في عمليات الماك على البيانات التي يرسلها الزبون.

- مفتاح الخادم للكتابة: مفتاح التشفير التقليدي للبيانات التي يشفّرها الخادم ويقوم الزبون بإزالة تشفيرها.
- مفتاح الزبون للكتابة: مفتاح التشفير التقليدي للبيانات التي يشفّرها الزبون ويقوم الخادم بإزالة تشفيرها.
- متجهات التهيئة (IV): عند استخدام تشفير كتلة في نمط CBC، يتم الاحتفاظ بمتجه تهيئة (IV) لكل مفتاح. يأخذ هذا الحقل قيمته الأولية من قِبَل بروتوكول المصافحة، ومن ثم يتم الاحتفاظ بكتلة الشفرة النهائية لكل سجل لاستخدامها كمتجه تهيئة للسجل الذي يليه.
- أرقام التسلسل: يحتفظ كل طرف بأرقام تسلسل مستقلة للرسائل المرسلة والمستلمة لكل توصيلة. عندما يرسل طرف أو يتلقى رسالة تغيير مواصفات التشفير (change cipher spec)، يقوم بتصفير رقم التسلسل المعني. يجب ألا تتجاوز قيمة رقم التسلسل $2^{64}-1$.

7-2-2 بروتوكول سجل SSL

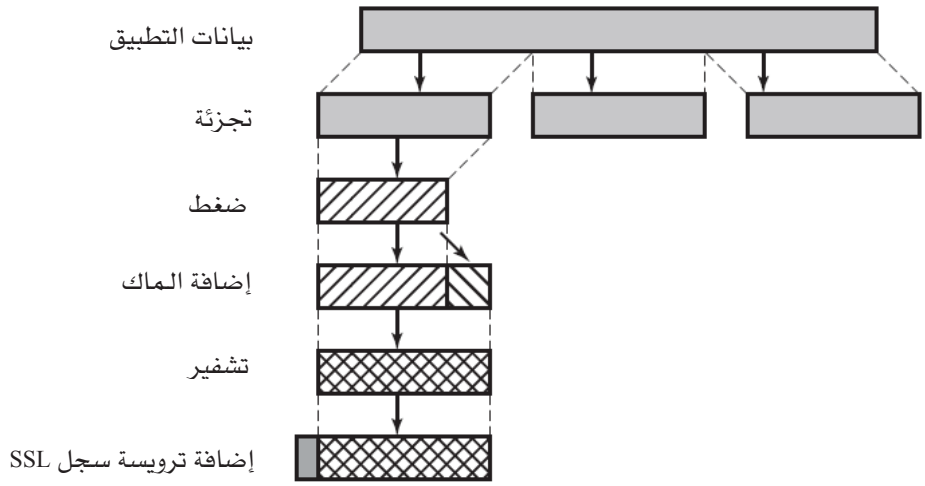
يوفر بروتوكول سجل SSL خدمتين لتوصيلات SSL:

- السرّية: يعرّف بروتوكول المصافحة مفتاحاً سرياً مشتركاً يُستخدَم للتشفير التقليدي للحمل الآجر لبروتوكول SSL.
- سلامة الرسالة: يُعرِّف بروتوكول المصافحة أيضاً مفتاحاً سرّياً مشتركاً يُستخدَم لتكوين كود توثيق الرسالة (الماك).

يبيّن الشكل 7-3 طريقة عمل بروتوكول سجل SSL. يأخذ البروتوكول رسالة التطبيق المراد إرسالها ويجزئ البيانات إلى كتلٍ يسهل التعامل معها، وقد يضغط البيانات (وهي عملية اختيارية)، ثم يطبّق الماك. يقوم بعد ذلك بتشفير الناتج، وإضافة ترويسة، ثم إرسال وحدة البيانات الناتجة في قطعة TCP. يتم إزالة تشفير

البيانات المُستلَمة وتوثيقها، وإزالة الضغط (إن وجد)، ثم يعاد تجميع الأجزاء وتسليمها للمستخدمين في الطبقات العليا.

إن أول خطوة هي **التجزئة**. تُجزأ كل رسالة واردة من الطبقة العليا إلى كتل مؤلَّفة من 2^{14} بايتاً (أي 16384 بايتاً) أو أقل. بعد ذلك تُضغط الكتل الناتجة عند الرغبة في ذلك؛ ويجب أن يكون الضغط بدون فقد (lossless) وألا يزيد طول المحتوى عن 1024 بايتاً. واضح أن الهدف من الضغط هو تقليص حجم البيانات. ومع ذلك، عندما تكون الكتل قصيرة جداً فقد تؤدي خوارزمية الضغط إلى زيادة الطول عن الطول الأصلي؛ بسبب مقتضيات الصياغة.



الشكل 7-3: طريقة عمل بروتوكول سجل SSL.

في بروتوكول SSLv3 (وأيضاً في النسخة الحالية من TLS)، لم يتم تحديد خوارزمية ضغط ومن ثمَّ تكون القيمة الاعتيادية لخوارزمية الضغط "لا يوجد" (null).

الخطوة التالية هي حساب كود توثيق الرسالة (MAC) للرسالة من البيانات المضغوطة. لهذا الغرض يُستخدَم مفتاح سرّي مشترك؛ وتتم الحسابات كما يأتي:

```
Hash(MAC_write_secret || pad_2 ||
        Hash(MAC_write_secret || pad_1 || seq_num ||
        SSLCompressed.type ||
        SSLCompressed.length || SSLCompressed.fragment))|
```

حيث:

|| = عملية الوصل (الإلحاق).

MAC_write_secret = مفتاح سري مشترك.

Hash = خوارزمية التحوير MD5 أو SHA-1.

pad_1 = القيمة 0x36 (أي 0110 0011) مكررة 48 مرة (أي بطول كلي 384 بتاً) في حالة MD5 ومكررة 40 مرة (أي بطول كلي 320 بتاً) في حالة SHA-1.

pad_2 = القيمة 0x5C (أي 1100 0101) مكررة 48 مرة في حالة MD5 ومكررة 40 مرة في حالة SHA-1.

seq_num = الرقم التسلسلي لتلك الرسالة.

SSLCompressed.type = بروتوكول الطبقة العليا المستخدم لمعالجة هذا الجزء (segment).

SSLCompressed.length = طول الجزء بعد الضغط.

SSLCompressed.fragment = الجزء المضغوط (في حالة عدم استخدام الضغط يوضع جزء النص الأصلي).

لاحظ أن هذه العملية تشبه بشكلٍ كبير جداً خوارزمية HMAC التي عرَّفناها في الفصل الثالث. الاختلاف هو أنه يتم وصل pad_1 مع pad_2 في SSLv3، لكن في HMAC يتم حساب العملية المنطقية XOR عليهما. وتعتمد خوارزمية ماك لبروتوكول SSLv3 على مسوّدة الإنترنت الأصلية لـ HMAC التي تستخدم عملية الوصل. أما النسخة النهائية لـ HMAC، والمُعرَّفة في RFC 2104 فتستخدم XOR. بعد ذلك تُشفَّر الرسالة المضغوطة مضافاً إليها الماك بالتشفير المتماثل. ولا يزيد التشفير طول المحتوى بأكثر من 1024 بايتاً، ومن ثمَّ لا يزيد الطول الكلي عن 2048+2^{14}.

يمكن استخدام أيٍّ من خوارزميات التشفير الآتية:

تشفير التدفق		تشفير الكتل	
الخوارزمية	طول المفتاح	الخوارزمية	طول المفتاح
RC4-40	40	AES	256 ،128
RC4-128	128	IDEA	128
		RC2-40	40
		DES-40	40
		DES	56
		3DES	168
		Fortezza	80

يمكن استخدام طريقة فورتيزا (Fortezza) في نظامٍ لتشفير البطاقات الذكية. في تشفير التدفق، تُشفَّر الرسالة المضغوطة مضافاً إليها الماك. لاحظ أن الماك يُحسَب قبل القيام بالتشفير وأن الماك يُشفَّر سويةً مع النص الأصلي أو النص الأصلي بعد الضغط.

في تشفير الكتلة، قد تُضاف بعد الماك وقبل التشفير حشوة (padding). تتكون الحشوة من سلسلة من البايتات يليها بايت يبيّن طول الحشوة. الطول الكلي للحشوة هو أقصر طول يجعل الطول الكلي للبيانات المراد تشفيرها (النص الأصلي + الماك + الحشوة) من مضاعفات طول كتلة الشفرة. وكمثال على ذلك،

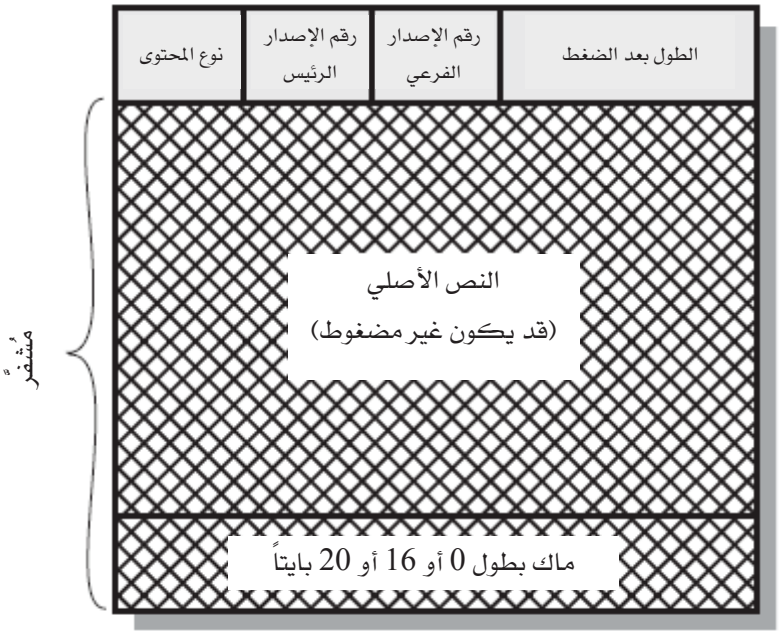
خذ في الاعتبار نصاً أصلياً (أو نصاً مضغوطاً إذا كان الضغط مستخدماً) طوله 58 بايتاً، وطول الماك 20 بايتاً (باستخدام SHA-1)، ويتم تشفيره باستخدام كتلة بطول 8 بايتات (مثلاً باستخدام DES). مع بايت طول الحشوة (padding.length byte) يكون المجموع 79 بايتاً. ولجعل المجموع الكلي من مضاعفات العدد 8، تضاف حشوة من بايت واحد.

تتضمن خطوة المعالجة النهائية في بروتوكول سجل SSL إضافة ترويسة تشمل الحقول الآتية:

- نوع المحتوى (8 بتات): بروتوكول الطبقة العليا المُستخدَم لمعالجة جزء الرزمة (fragment) المتضمَّن.
- الإصدار الرئيس (8 بتات): يحدد إصدار SSL الرئيس المُستخدَم. في حالة SSLv3 تكون القيمة 3.
- الإصدار الثانوي (8 بتات). يحدد إصدار SSL الثانوي المُستخدَم. في حالة SSLv3 تكون القيمة 0.
- الطول بعد الضغط (16 بتاً): طول جزء النص الأصلي بالبايتات (أو طول الجزء المضغوط إذا كان الضغط مستخدماً). القيمة القصوى هي $2^{14}+2048$.

أنواع المحتوى المُعرَّفة هي: change_cipher_spec (تغيير مواصفات الشفرة)، وalert (تنبيه)، وhandshake (مصافحة)، وapplication_data (بيانات تطبيق). الأنواع الثلاثة الأولى هى بروتوكولات خاصة بـ SSL، وسنناقشها فيما بعد. لاحظ عدم وجود تمييز بين التطبيقات المختلفة التي قد تستخدم SSL (كـ HTTP مثلاً)؛ فمحتوى البيانات من مثل هذه التطبيقات يبقى غير مميّز لـ SSL.

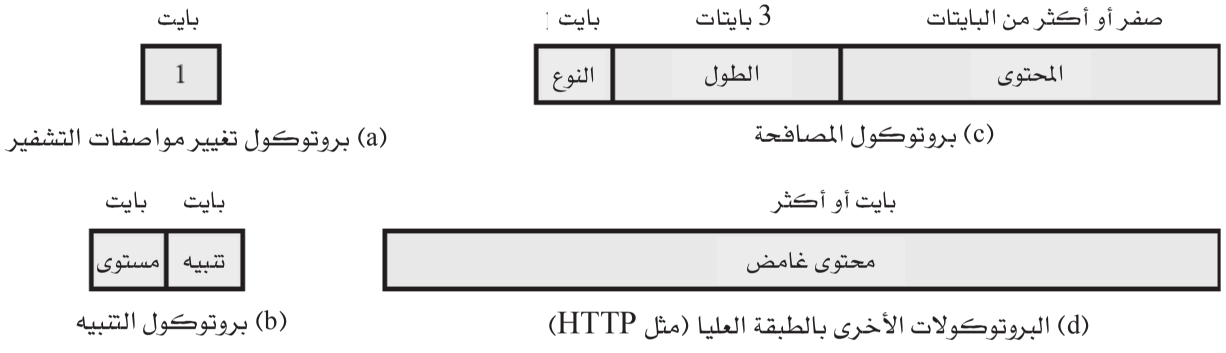
يبيّن الشكل 7-4 صيغة بروتوكول سجل SSL.



الشكل 7-4: صيغة سجل SSL.

3-2-7 بروتوكول تغيير مواصفات الشفرة

بروتوكول تغيير مواصفات الشفرة (change cipher spec protocol) هو أحد البروتوكولات الثلاثة للبرتوكولات المحددة بـ SSL التي تستخدم بروتوكول سجل SSL؛ وهو أبسطها. ويشمل هذا البروتوكول رسالة واحدة (الشكل 7-5 (a)) تتكون من بايت واحد قيمته 1. الغرض الوحيد من هذه الرسالة هو التسبب في نسخ الحالة المُعلَّقة (pending state) إلى الحالة الحالية (current state)، والتي تُحدِّث مجموعة برمجيات التشفير التي ستستخدم على هذه التوصيلة.



الشكل 7-5: حمولة بروتوكول سجل SSL.

4-2-7 بروتوكول التنبيه

بروتوكول التنبيه هو البروتوكول المسؤول عن رسائل التنبيه في SSL للكيان النظير. كما هو الحال مع التطبيقات الأخرى التي تستخدم SSL، تُضغط الرسائل وتُشفَّر كما هو محدد بالحالة الحالية.

تتكون كل رسالة في هذا البروتوكول من بايتين (الشكل 7-5 (b)). يأخذ البايت الأول القيمة 1 (تعني تحذير) أو 2 (تعني فادح) لبيان مستوى خطورة الرسالة. وإذا كان مستوى الخطورة "فادح"، فسيُغلق SSL تلك التوصيلة فوراً. بينما قد تستمر التوصيلات الأخرى ضمن نفس الجلسة، لن يتم إنشاء توصيلات جديدة ضمن تلك الجلسة. ويتضمّن البايت الثاني كوداً يبيّن رسالة التنبيه المحددة. وسنسرد هنا أولاً تلك التنبيهات التي تُعدُّ دائماً فادحة (التعريفات مُقتبسة من مواصفات SSL):

- unexpected_message (رسالة غير متوقعة): للإشارة إلى استلام رسالة غير متوقعة.
- bad_record_mac (ماك سجل خطأ): للإشارة إلى استلام ماك خطأ.
- decompression_failure (فشل إزالة الضغط): للإشارة إلى أن المُدخَل لوظيفة إزالة الضغط غير صحيح (مثلاً عدم القدرة على إزالة الضغط، أو إزالة الضغط لأكثر من الطول المسموح به).
- handshake_failure (فشل المصافحة): للإشارة إلى عدم قدرة المُرسِل على التوصل إلى مجموعة مقبولة من مُعامِلات الأمن في ظل الخيارات المتاحة للتفاوض.
- illegal_parameter (معامِل غير صحيح): للإشارة إلى وجود حقل في رسالة المصافحة قيمته خارج المدى أو متناقضة مع الحقول الأخرى.

أما بقيّة التنبيهات فهي على النحو الآتي:

- close_notify (إشعار غلق): لإشعار المُستلِم بأن المُرسِل لن يرسل رسائل أخرى على هذه التوصيلة. على كل طرف إرسال تنبيه close_notify قبل إغلاق جهة الكتابة للتوصيلة.
- no_certificate (لا توجد شهادة): قد يُرسَل هذا التنبيه رداً على طلب شهادة عند عدم توفر شهادة ملائمة.
- bad_certificate (شهادة غير صالحة): في حالة وجود خلل في الشهادة المُستلَمة (مثلاً قد تحتوي على توقيع لا يمكن تحقيقه).
- unsupported_certificate (شهادة غير مدعومة): نوع الشهادة المُستلَمة غير مدعوم.
- certificate_revoked (شهادة ملغاة): تم إلغاء الشهادة من قِبَل مُوقِّعها.
- certificate_expired (شهادة منتهية الصلاحية).
- certificate_unknown (شهادة غير معروفة): ظهرت مشكلة أخرى غير محدّدة في معالجة الشهادة جعلتها غير مقبولة.

7-2-5 بروتوكول المصافحة

يُعدُّ بروتوكول المصافحة الجزء الأكثر تعقيداً في SSL. ويسمح هذا البروتوكول للخادم والزبون بالتحقق من هوية بعضهما بعضاً وبالتفاوض على خوارزمية التشفير والماك ومفاتيح التشفير التي ستُستخدم لحماية البيانات المُرسَلة في سجل SSL. ويُستخدم بروتوكول المصافحة قبل إرسال أي بيانات من التطبيق.

يشمل بروتوكول المصافحة سلسلةً من الرسائل المُتبادلة بين الزبون والخادم، لكلٍّ منها الصيغة المبينة بالشكل 7-5 (c). وتتألف كل رسالة من ثلاثة حقول:

- النوع (بايت واحد): يدُل هذا الحقل على إحدى الرسائل العشر والمبينة بالجدول 7-2.

- الطول (3 بايتات): طول الرسالة بالبايتات.
- المحتوى (تبادل الرسائل ≤ 0 بايت): المُعامِلات المرتبطة بهذه الرسالة؛ وهي معروضة بالجدول 7-2.

يبيّن الشكل 7-6 التبادل المطلوب لإنشاء اتصال منطقي بين الزبون والخادم، وينقسم إلى أربع مراحل.

❖ المرحلة 1. تأسيس إمكانيات الأمن:

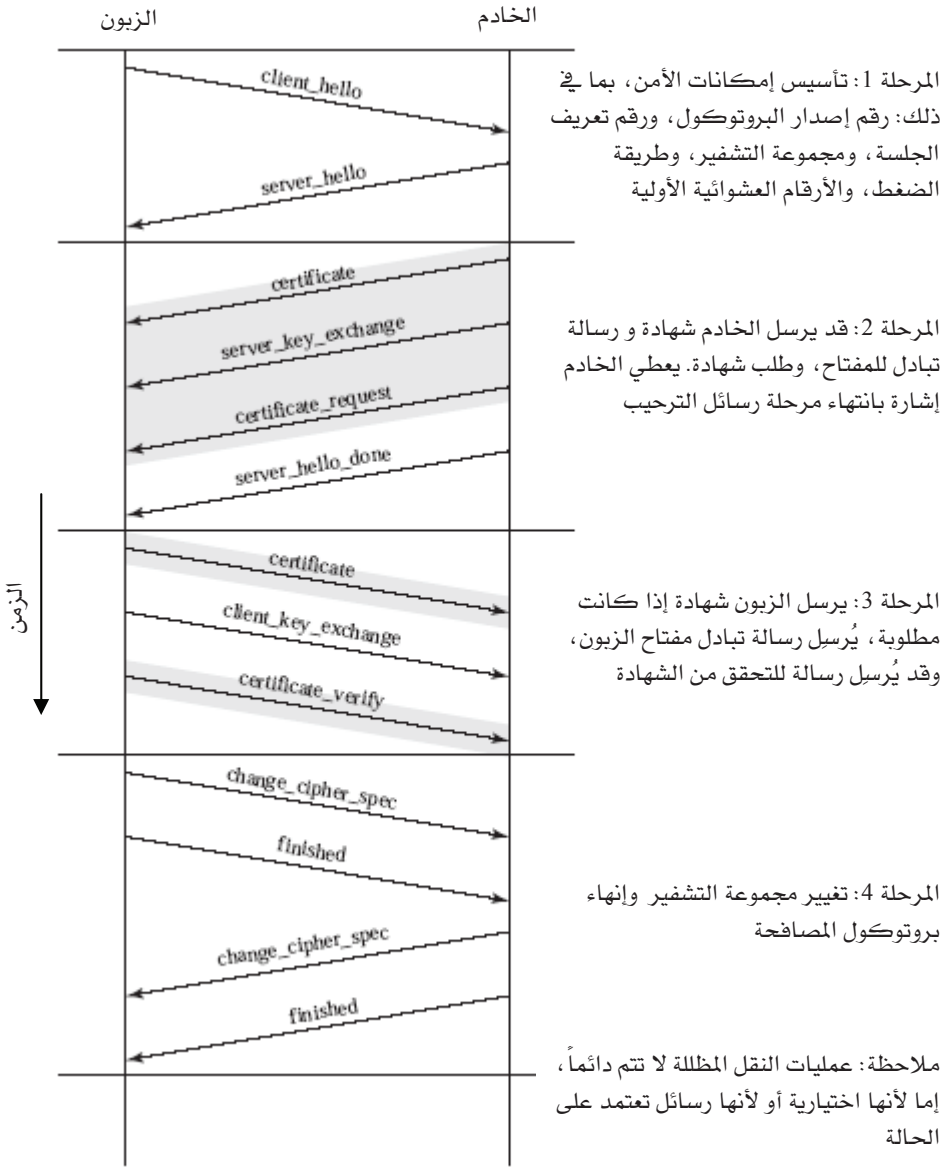
تُستخدَم هذه المرحلة لبدء توصيلة منطقية وتأسيس إمكانيات الأمن المرتبطة بها. يبدأ الزبون بإرسال رسالة ترحيب (client_hello_message) تشمل المُعامِلات الآتية:

- Version (الإصدار): أعلى رقم إصدار لـ SSL يمكن للزبون فهمه.
- Random (تركيبة عشوائية): يولدها الزبون وتشمل 32 بتاً عبارة عن خاتم وقت و28 بايتاً يولدها مُولِّد أرقام عشوائية. وتستخدم هذه القيم لمرة واحدة (nonces) أثناء عملية تبادل المفاتيح لمنع هجمات إعادة التشغيل.
- SessionID (مُعرِّف الجلسة): بطول متغيّر، وتشير قيمته غير الصفرية إلى الزبون الذي يرغب في تحديث مُعامِلات توصيلة حالية أو يريد بدء توصيلة جديدة ضمن هذه الجلسة. أما القيمة الصفرية فتشير إلى أن الزبون يرغب في تأسيس توصيلة جديدة على جلسة جديدة.
- CipherSuite (مجموعة خوارزميات التشفير): هذه قائمة تحتوي على مجموعة من خوارزميات التشفير التي يدعمها الزبون مرتبة حسب الأفضلية من الأعلى إلى الأدنى. يُعرِّف كل عنصر في تلك القائمة خوارزميةً لتبادل المفاتيح ومواصفات الشفرة (CipherSpec)؛ وسنناقشها لاحقاً.
- Compression Method (طريقة الضغط): هذه قائمة بطرق ضغط البيانات التي يدعمها الزبون.

الجدول 7-2: أنواع رسائل بروتوكول مصافحة SSL.

نوع الرسالة	المُعامِلات
طلب ترحيب (hello_request)	لايوجد (null)
ترحيب من الزبون (client_hello)	رقم الإصدار، رقم عشوائي، رقم تعريف الجلسة، مجموعة التشفير، طريقة ضغط البيانات
ترحيب من الخادم (server_hello)	رقم الإصدار، رقم عشوائي، رقم تعريف الجلسة، مجموعة التشفير، طريقة ضغط البيانات
شهادة (certificate)	سلسلة من شهادات X.509v3
تبادل مفتاح الخادم (server_key_exchange)	مُعامِلات، توقيع
طلب شهادة (certificate_request)	النوع، سلطات التصديق
اكتمال الخدمة (server_done)	لا يوجد (null)
التحقق من الشهادة (certificate_verify)	توقيع
تبادل مفتاح الزبون (client_key_exchange)	مُعامِلات، توقيع
الانتهاء (finished)	قيمة كود التحوير

بعد أن يرسل الزبون رسالة الترحيب، ينتظر رسالة ترحيب من الخادم (server_hello message) تتضمّن نفس المُعامِلات المتضمنة في رسالة الترحيب التي أرسلها. تنطبق الاصطلاحات التالية على رسالة الترحيب من الخادم. ويحتوي حقل الإصدار (Version) على أعلى قيمة إصدار يدعمها الخادم من بين قيم الإصدارات المُقترحة من الزبون. ويُولَّد الحقل العشوائي (Random) لدى الخادم بشكلٍ مستقل عن الحقل العشوائي لدى الزبون. وإذا كانت قيمة حقل مُعرِّف الجلسة (SessionID) لدى الزبون لا تساوي صفراً، يستخدم الخادم نفس القيمة؛ وإلا فسيتضمّن هذا الحقل لدى الخادم القيمة الخاصة بجلسة جديدة.



الشكل 7-6: طريقة عمل بروتوكول المصافحة.

يتضمّن حقل مجموعة التشفير (CipherSuite) مجموعة التشفير التي اختارها الخادم من بين المجموعات المقترحة من قِبَل الزبون. كما يتضمّن حقل الضغط (Compression) طريقة الضغط التي اختارها الخادم من بين الطرق المقترحة من قِبَل الزبون.

يتضمن العنصر الأول من مُعامِل مجموعة التشفير طريقة تبادل المفاتيح (الطريقة التي يتم بها تبادل مفاتيح التشفير للتشفير التقليدي والماك). يوفر SSL دعماً للطرق الآتية لتبادل المفاتيح:

- طريقة RSA: يُشفَّر المفتاح السري بمفتاح RSA العام للمُستلِم. ويجب توفُّر شهادة مفتاح عام لمفتاح المُستلِم.
- طريقة ديفي - هيلمان الثابتة (Fixed Deffie-Hellman): هي طريقة ديفي - هيلمان لتبادل المفاتيح وفيها تتضمّن شهادة الخادم مُعامِلات ديفي - هيلمان العامة موقَّعة من سلطة الشهادات (CA). يوفر الزبون مُعامِلات ديفي - هيلمان للمفتاح العام الخاص به إما في شهادة (إذا كان توثيق الزبون مطلوباً)، أو في رسالة تبادل مفاتيح. وتؤدي هذه الطريقة إلى توفير مفتاح سري ثابت بين الطرفين على أساس حساب ديفي - هيلمان باستخدام المفاتيح العامة الثابتة.
- طريقة ديفي - هيلمان العابرة (Ephemeral Diffie-Hellman): تُستخدَم هذه الطريقة لتكوين مفاتيح سرية عابرة (مؤقتة، تُستخدم لمرة واحدة). في هذه الحالة، يتم تبادل مفاتيح ديفي - هيلمان العامة، مُوقَّعةً باستخدام المفتاح الخاص للمُرسِل في أسلوب RSA أو DSS. ويمكن أن يستخدم المُستلِم المفتاح العام المُناظر للتحقق من التوقيع. وتُستخدم الشهادات للتحقق من المفاتيح العامة. وتبدو هذه الطريقة الأكثر أمناً بين خيارات ديفي - هيلمان الثلاثة؛ حيث تؤدّي إلى مفتاح مؤقت موثَّق.
- طريقة ديفي - هيلمان مجهولة المصدر (Anonymous Diffie-Hellman): تُستخدم خوارزمية ديفي - هيلمان الأساسية بدون توثيق. أي يرسل كل

جانب مُعامِلات ديفي - هيلمان العامة إلى الآخرين بدون توثيق. هذه الطريقة عرضة لهجمات "رجل في الوسط"، حيث يستخدم المهاجم طريقة ديفي - هيلمان مجهولة المصدر مع كلا الطرفين المتصلين (وذلك بالتعامل مع كلٍّ منهما على أنه الطرف الآخر).

- طريقة Fortezza: وهي الطريقة المُعرَّفة لأسلوب Fortezza.

بعد تعريف طريقة تبادل المفاتيح يأتي مُعامِل CipherSpec، ويتضمّن الحقول الآتية:

- CipherAlgorithm (خوارزمية التشفير): أيٌّ من الخوارزميات التي ذكرت سابقاً: RC4، RC2، DES، 3DES، DES40، IDEA، Fortezza.
- MACAlgorithm (خوارزمية كود التحوير (الماك)): MD5 أو SHA-1.
- CipherType (نوع التشفير): تشفير كتل أو تشفير تدفق.
- IsExportable (إمكانية التصدير): نعم (true) أو لا (false).
- HashSize (طول كود التحوير): بقيمة 0 أو 16 بايتاً (لـ MD5)، أو 20 بايتاً (لـ SHA-1).
- key material (مادّة المفتاح): سلسلة من البايتات تتضمّن البيانات المُستخدَمة في توليد مفاتيح الكتابة.
- IV Size (طول متجه التهيئة): طول متجه التهيئة لأسلوب التشفير "سَلسَلة الكتل المُشفَّرة" (CBC).

❖ المرحلة 2. توثيق الخادم وتبادل المفاتيح:

يبدأ الخادم هذه المرحلة بإرسال الشهادة الخاصة به، إذا كان هناك حاجة للتحقق من هويته؛ وتتضمّن الرسالة شهادة أو سلسلة من شهادات X.509. يحتاج الأمر إلى رسالة الشهادة عند استخدام أي طريقة متفق عليها لتبادل المفاتيح باستثناء طريقة ديفي - هيلمان مجهولة المصدر. لاحظ أنه في حالة استخدام طريقة ديفي -

هيلمان الثابتة، فإن رسالة الشهادة هذه تُستغَل كرسالة تبادل لمفتاح الخادم؛ لأنها تتضمّن مُعامِلات ديفي - هيلمان العامة للخادم.

بعد ذلك قد تُرسَل رسالة تبادل مفتاح الخادم (server_key_exchange) إذا تطلب الأمر ذلك. ولا يلزم إرسال تلك الرسالة في إحدى حالتين: (1) إذا كان الخادم قد أرسل شهادة مع مُعامِلات ديفي - هيلمان الثابتة، (2) في حالة استخدام طريقة RSA لتبادل المفاتيح. ويلزم إرسال تلك الرسالة في الحالات الآتية:

- طريقة ديفي - هيلمان مجهولة المصدر: يتضمن محتوى الرسالة قيمتي ديفي -هيلمان العامتين (عدد أولي prime number وجذر بدائي primitive root لذلك العدد)، بالإضافة إلى مفتاح ديفي - هيلمان العام للخادم (انظر الشكل 3-10).
- طريقة ديفي - هيلمان العابرة: يتضمّن محتوى الرسالة مُعامِلات ديفي - هيلمان الثلاثة الخاصة بطريقة ديفي - هيلمان مجهولة المصدر، بالإضافة إلى توقيع لتلك المُعامِلات.
- طريقة RSA لتبادل المفاتيح التي يستخدم الخادم فيها خوارزمية RSA ولكن لديه مفتاح RSA للتوقيع فقط: وفقاً لذلك، لا يستطيع الزبون أن يرسل ببساطة مفتاحاً سرياً مُشفَّراً بمفتاح الخادم العام. بدلاً من ذلك، يجب أن يُنشئ الخادم زوجاً من مفاتيح RSA العامة/الخاصة المؤقتة، وأن يستخدم رسالة تبادل مفتاح الخادم (server_key_exchange) لإرسال المفتاح العام. يتضمّن محتوى الرسالة اثنين من مُعامِلات مفتاح RSA العام المؤقتة (الأس exponent ومعامِل حساب الباقي modulus؛ انظر الشكل 3-8)، بالإضافة إلى توقيع لتلك المُعامِلات.
- طريقة Fortezza.

لعله من المفيد هنا ذكر بعض التفاصيل الأخرى فيما يتعلق بالتوقيعات. كالمعتاد يتم إنشاء توقيع بإجراء تحوير للرسالة وتشفير الناتج بالمفتاح الخاص للمرسل. وفي هذه الحالة، يحسب كود التحوير من

Hash(ClientHello.radom || ServerHello.random || ServerParams)

وهكذا يغطي كود التحوير ليس فقط مُعامِلات ديفي - هيلمان أو RSA، ولكن أيضاً قيمتي الأعداد التي تُستخدم لمرة واحدة (nonces) من رسائل الترحيب الأولى. ويضمن هذا الحماية ضد هجمات إعادة التشغيل والتمثيل الخاطئ. وفي حالة توقيع DSS، يُحسب كود التحوير باستخدام خوارزمية SHA-1. وفي حالة توقيع RSA، يحسب كود التحوير باستخدام كلٍّ من MD5 وSHA-1، ثم يوصل كودي التحوير ويشفر الناتج (36 بايتاً) بالمفتاح الخاص للخادم.

بعد ذلك يمكن أن يطلب خادم غير مجهول (أي لا يستخدم طريقة ديفي - هيلمان مجهولة المصدر) شهادةً من الزبون. تتضمّن رسالة طلب الشهادة (certificate_request) معامِلين: نوع الشهادة، وسلطات الشهادة. ويبيّن نوع الشهادة خوارزمية المفتاح العام واستخدامها:

- RSA، توقيع فقط.
- DSS، توقيع فقط.
- RSA لطريقة ديفي - هيلمان الثابتة؛ في هذه الحالة يُستخدَم التوقيع فقط للتوثيق بإرسال الشهادة موقَّعة بـ RSA.
- DSS لطريقة ديفي - هيلمان الثابتة؛ مرة أخرى تُستخدَم فقط للتوثيق.
- RSA لطريقة ديفي - هيلمان العابرة.
- DSS لطريقة ديفي - هيلمان العابرة.
- Fortezza.

أما المُعامِل الثاني في رسالة طلب الشهادة فعبارة عن قائمة بالأسماء المميزة للسلطات المعتمدة لإصدار الشهادات. الرسالة النهائية في المرحلة 2، تُعدُّ رسالة مطلوبة دائماً، هي رسالة اكتمال الخدمة (server_done)، يرسلها الخادم للإشارة إلى الانتهاء من ترحيبه والرسائل المتعلقة به. وبعد إرسال تلك الرسالة ينتظر الخادم رداً من الزبون. وهذه الرسالة ليس لها مُعامِلات.

❖ المرحلة 3. توثيق الزبون وتبادل المفاتيح:

بعد استلام رسالة اكتمال الخدمة (server_done)، يجب أن يتحقق الزبون من أن الخادم قد أعطى شهادة صحيحة إذا كان ذلك مطلوباً وأن يتأكد من أن مُعامِلات رسالة ترحيب الخادم مقبولة. وإذا تم كل ذلك بشكلٍ مرضٍ، يرد الزبون بإرسال رسالة أو أكثر إلى الخادم. وإذا كان الخادم قد طلب شهادة، يبدأ الزبون هذه المرحلة بإرسال رسالة شهادة (certificate). وإذا لم تتوفر شهادة مناسبة، يُرسل الزبون بدلاً من ذلك رسالة تنبيه بعدم وجود شهادة.

بعد ذلك تأتي رسالة تبادل مفتاح الزبون (client_key_exchange)، ويجب إرسالها في تلك المرحلة. ويعتمد محتوى تلك الرسالة على نوع عملية تبادل المفاتيح المُستخدَمة؛ كما هو مبيّن أدناه:

- RSA: يولِّد الزبون السرّ قبل الرئيس (pre-master secret) ويتكون من 48 بايتاً ويشفره بالمفتاح العام الذي حصل عليه من شهادة الخادم أو بمفتاح RSA المؤقت من رسالة تبادل مفتاح الخادم (server_key_exchange). سنوضح كيفية استخدام ذلك السر لحساب "السر الرئيس" (master secret) فيما بعد.
- طريقة ديفي - هيلمان العابرة أو مجهولة المصدر: تُرسَل مُعامِلات ديفي - هيلمان العامة الخاصة بالزبون.

- طريقة ديفي - هيلمان الثابتة: نظراً لأن مُعامِلات ديفي - هيلمان العامة الخاصة بالزبون يكون قد سبق إرسالها في رسالة شهادة، فإن محتوى تلك الرسالة يكون فارغاً.
- Fortezza: تُرسَل مُعامِلات Fortezza الخاصة بالزبون.

في نهاية هذه المرحلة، قد يرسل الزبون رسالة للتحقق من الشهادة (certificate_verify) لتوفير تحقق صريح من شهادة الزبون. وترسل هذه الرسالة فقط بعد أي شهادة زبون لها قابلية للتوقيع (أي كل الشهادات ما عدا تلك التي تتضمن مُعامِلات ديفي - هيلمان الثابتة). وتوقَّع هذه الرسالة كود تحوير بناءً على الرسائل السابقة حسب التعريف الآتي:

```
CertificateVerify.signature.md5_hash
MD5(master_secret || pad_2 || MD5(handshake_messages ||
master_secret || pad_1));
Certificate.signature.sha_hash
SHA(master_secret || pad_2 || SHA(handshake_messages ||
master_secret || pad_1));
```

حيث يرمز المتغيران pad_1 وpad_2 للقيم المُعرَّفة سابقاً لكود توثيق الرسالة (الماك)، ويشير المتغير handshake_messages لجميع رسائل بروتوكول المصافحة التي أُرسِلت أو استُلِمت بعد رسالة ترحيب الزبون (client_hello) باستثناء تلك الرسالة. المتغير master_secret هو السرّ الرئيس المحسوب (وسنوضح كيفية حسابه لاحقاً في هذا الجزء). إذا كان المفتاح الخاص للمُستخدِم هو DSS، فإنه يُستخدَم لتشفير كود تحوير SHA-1. وإذا كان المفتاح الخاص للمُستخدِم هو RSA، فإنه يُستخدَم لتشفير أكواد التحوير لـ MD5 وSHA-1 بعد وصلها. وفي كلتا الحالتين يكون الهدف هو التحقق من ملكية الزبون للمفتاح الخاص لشهادة الزبون. حتى إذا أساء شخص ما استخدام شهادة الزبون، فإنه لن يتمكن من إرسال تلك الرسالة.

❖ **المرحلة 4. الإنهاء:**

تُنهي هذه المرحلة عملية تأسيس توصيلة آمنة. يرسل الزبون رسالة تغيير مواصفات التشفير (change_cipher_spec) وينسخ CipherSpec المعلّق إلى CipherSpec الحالي. لاحظ أن هذه الرسالة لا تُعدُّ جزءاً من بروتوكول المصافحة ولكنها تُرسَل باستخدام بروتوكول تغيير مواصفات التشفير. بعد ذلك مباشرةً يرسل الزبون رسالة الإنهاء (finished) مستخدماً الخوارزميات والمفاتيح والأسرار الجديدة.

تؤكد رسالة الإنهاء إنجاز عمليات تبادل المفاتيح والتوثيق بنجاح. ويتكون محتوى رسالة الإنهاء من قيمتين موصولتين من كودي تحوير:

```
MD5(master_secret || pad_2 || MD5(handshake_messages ||
      Sender || master_secret || pad_1))

SHA(master_secret || pad_2 || SHA(handshake_messages ||
      Sender || master_secret || pad_1))
```

حيث يمثل Sender الكود الذي يتحقق من أن المُرسِل هو الزبون، وتمثل handshake_messages كل البيانات من رسائل المصافحة حتى ما قبل تلك الرسالة.

استجابةً لهاتين الرسالتين، يرسل الخادم رسالة تغيير مواصفات التشفير (change_cipher_spec) الخاصة به، وينقل CipherSpec المعلّق إلى CipherSpec الحالي، ويرسل رسالة إنهاء. عندئذٍ تكون عملية المصافحة قد اكتملت، ويصبح بوسع الزبون والخادم البدء في تبادل بيانات طبقة التطبيقات.

6-2-7 حسابات التشفير

هناك عنصران آخران جديران بالاهتمام: توليد سرٍّ رئيس مشترك بواسطة تبادل المفاتيح، وتوليد مُعامِلات التشفير من السرّ الرئيس.

❖ توليد سرّ رئيس:

يتألف السر الرئيس المشترك من 48 بايتاً (384 بتاً)، ويتم توليده لهذه الجلسة بواسطة تبادل مفاتيح آمن؛ وذلك على مرحلتين. في المرحلة الأولى يتم تبادل السرّ قبل الرئيس (pre-master secret)، وفي الثانية يُحسب السرّ الرئيس من قِبَل كلا الطرفين. هناك طريقتان لتبادل السرّ قبل الرئيس:

- RSA: يتألف السرّ قبل الرئيس من 48 بايتاً ويولَّد من قِبَل الزبون، ويشفَّر بمفتاح RSA العام للخادم، ويُرسَل إلى الخادم. يقوم الخادم بإزالة التشفير باستخدام مفتاحه الخاص لاستعادة السرّ قبل الرئيس.
- ديفي - هيلمان: يولِّد كلٌّ من الزبون والخادم مفتاح ديفي - هيلمان العام. وبعد تبادل هذين المفتاحين، يقوم كل جانب بعملية ديفي - هيلمان الحسابية لتوليد السرّ قبل الرئيس المشترك.

الآن يحسب كلا الجانبين السرّ الرئيس على النحو الآتي:

```
master_secret =
MD5(pre_master_secret || SHA('A' || pre_master_secret || ClientHello.random ||
    ServerHello.random)) ||MD5(pre_master_secret || SHA('BB' ||
    pre_master_secret || ClientHello.random || ServerHello.random)) ||
    MD5(pre_master_secret || SHA('CCC' || pre_master_secret ||
    ClientHello.random || ServerHello.random))
```

حيث ClientHello.random وServerHello.random هما قيمتا العددين اللذين يستخدمان لمرة واحدة (nonces) وتم تبادلهما في رسائل الترحيب الأولى.

❖ توليد مُعامِلات التشفير:

تتطلب مواصفات التشفير CipherSpecs سرّ ماك للكتابة للزبون، وسرّ ماك للكتابة للخادم، ومفتاح كتابة للزبون، ومفتاح كتابة للخادم، ومتجه تهيئة (IV) للكتابة للزبون، ومتجه تهيئة (IV) للكتابة للخادم. وتولَّد تلك المُعامِلات من السرّ

الرئيس على التوالي بحساب كود التحوير للسرّ الرئيس للحصول على سلسلة بايتات آمنة بطول كافٍ لكل المُعامِلات المطلوبة. ويستخدم توليد مادّة المفتاح من السرّ الرئيس نفس الصيغة المتّبعة لتوليد السرّ الرئيس من السرّ قبل الرئيس:

```
key_block =
MD5(master_secret || SHA('A' || master_secret || ServerHello.random ||
    ClientHello.random)) || MD5(master_secret || SHA('BB' ||
    master_secret || ServerHello.random || ClientHello.random)) ||
    MD5(master_secret || SHA('CCC' || master_secret ||
    ServerHello.random || ClientHello.random)) || . . .
```

ويكرر ذلك إلى أن يتم توليد ناتج كافٍ. ويمثل هذا التركيب الخوارزمي دالةً شبه عشوائية. ويمكن اعتبار السرّ الرئيس كقيمة البذرة (seed value) شبه العشوائية للدالة (تعرف هذه القيمة أيضاً بالقيمة المبدئية لمولد الأعداد العشوائية). ويمكن اعتبار الأعداد العشوائية للزبون والخادم كقيم الملح (salt values) التي تقوم بتعقيد عملية تحليل الشفرة (راجع الفصل التاسع لمناقشة قيم الملح).

7-2-7 أمن طبقة النقل

بروتوكول TLS هو مبادرة معيارية للجنة هندسة الإنترنت (IETF) تهدف إلى إصدار معيار للإنترنت من SSL. تم تعريف TLS كمعيار مقترح للإنترنت في الوثيقة RFC 2246؛ وهو يشبه بدرجة كبيرة جداً SSLv3. نسلط الضوء في هذا الجزء على الاختلافات بينهما.

❖ رقم الإصدار:

صيغة سجل TLS هي نفسها تماماً صيغة سجل SSL (الشكل 7-4)، وحقول الترويسة لها نفس المعاني في كليهما. الاختلاف هو في قيم رقم الإصدار. لإصدار TLS الحالي يكون رقم الإصدار الرئيس هو 3 ورقم الإصدار الثانوي هو 1.

❖ كود توثيق الرسالة (الماك):

يختلف أسلوب الماك المستخدم في كلٍّ من SSLv3 وTLS في نقطتين: الخوارزمية الفعلية ومجال حساب الماك. يستخدم TLS خوارزمية HMAC المُعرَّفة في RFC 2104. ذكرنا في الفصل الثالث أن HMAC مُعرَّف على النحو الآتي:

$$HMAC_K(M) = H[(K^+ \oplus opad) \| H[(K^+ \oplus ipad) \| M]]$$

حيث:

H = دالة التحوير المتضمَّنة (يستخدم TLS دالة MD5 أو SHA-1).

M = رسالة المُدخَل لـ HMAC.

K^+ = السرّ الرئيس مطولٌ بأصفار من ناحية اليسار حتى يصبح الناتج مساوياً لطول الكتلة لكود التحوير (في حالة MD5 وSHA-1 طول الكتلة يساوي 512 بتاً).

ipad = 00110110 (أي 36 في النظام الست عشري) مكررة 64 مرة (ليصبح الطول 512 بتاً).

opad = 01011100 (أي 5C في النظام الست عشري) مكررة 64 مرة (ليصبح الطول 512 بتاً).

يستخدم SSLv3 نفس الخوارزمية باستثناء أن بايتات الحشوة (padding bytes) تُوصَل بالمفتاح السري بدلاً من حساب XOR بينها وبين المفتاح السري وطول الكتلة. مستوى الأمن هو تقريباً نفسه في الحالتين.

في حالة TLS، يتضمن حساب الماك الحقول المذكورة في التعبير الآتي:

```
HMAC_hash(MAC_write_secret, seq_num || TLSCompressed.type ||
          TLSCompressed.version || TLSCompressed.length ||
          TLSCompressed.fragment)
```

يغطي حساب الماك كل الحقول المغطاة في حساب SSLv3، بالإضافة إلى حقل TLSCompressed.version والذي يشير لرقم إصدار البروتوكول المُستخدَم.

❖ الدالة شبه العشوائية:

يستخدم TLS دالة شبه عشوائية تُعرف باسم PRF لتوسيع قيمة السرّ إلى كتل بيانات تُستخدم لتوليد المفاتيح أو التحقق من صحتها. ويكمن الهدف من ذلك في التمكن من استخدام سرٍّ مشتركٍ صغيرٍ نسبياً ومع ذلك توليد كتل أطول من البيانات بطريقة آمنة إزاء أنواع الهجمات التي قد تُشن على دوال التحوير والماك. تعتمد PRF على الدالة الآتية لتوسيع البيانات (الشكل 7-7):

$$\begin{aligned}P\_hash(secret, seed) = {} & HMAC\_hash(secret, A(1) \,||\, seed) \,|| \\ & HMAC\_hash(secret, A(2) \,||\, seed) \,|| \\ & HMAC\_hash(secret, A(3) \,||\, seed) \,|| \ldots\end{aligned}$$

حيث تُعرَّف $A()$ على النحو الآتي:

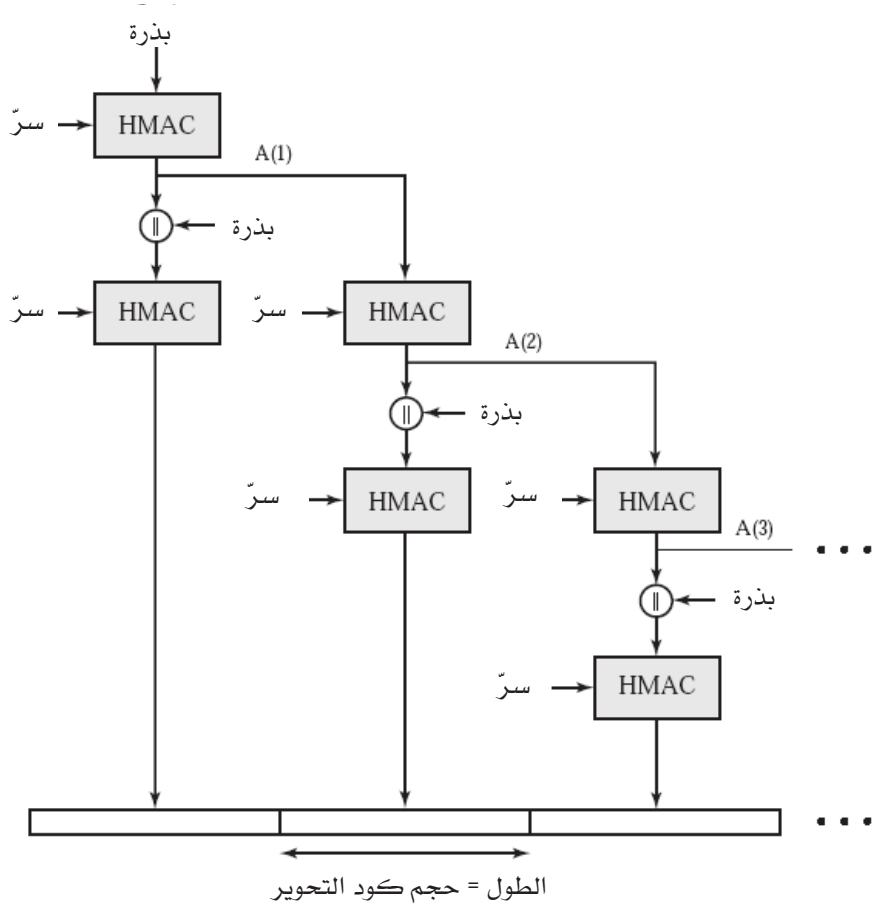
$$A(0) = seed$$
$$A(i) = HMAC\_hash(secret, A(i\text{-}1))$$

تستخدم دالة توسيع البيانات خوارزمية HMAC، مع MD5 أو SHA-1 كدالة التحوير التحتية. وكما ترى يمكن تكرار P_hash أيّ عدد من المرات نحتاجه لتوليد كمية البيانات المطلوبة. فعلى سبيل المثال، إذا استخدمنا P_SHA-1 لتوليد 64 بايتاً من البيانات، يجب تكررها أربع مرات، ومن ثم إنتاج 80 بايتاً من البيانات، ويتم إهمال الـ 16 بايتاً الأخيرة منها. وفي هذه الحالة، نحتاج لتكرار P_MD5 أيضاً أربع مرات، مما ينتج 64 بايتاً بالضبط. لاحظ أن كل تكرار يتضمن تنفيذ HMAC مرتين، يتضمن كلٌّ منهما بدوره تنفيذ خوارزمية التحوير التحتية مرتين.

لكي تكون دالة PRF آمنة بأقصى قدرٍ ممكن فإنها تستخدم خوارزميتي تحوير بطريقةٍ تضمن أمنها إذا ما بقت إحدى الخوارزميتين آمنة. تُعرّف دالة PRF على النحو الآتي:

$$\begin{aligned}PRF(secret, label, seed) = {} & P\_MD5(S1, label \,||\, seed) \oplus \\ & P\_SHA\text{-}1(S2, label \,||\, seed)\end{aligned}$$

تأخذ PRF قيمة سرية ووسمة تعريف وقيمة بذرة، وتنتج مُخرجاً بطولٍ اعتباطي. ويتم إنشاء الناتج بتقسيم قيمة السرّ إلى نصفين (S1 وS2)، ثم حساب P_hash لكل نصف باستخدام MD5 على نصف وSHA-1 على النصف الآخر. ثم تُجرى عملية XOR على النتيجتين للحصول على المُخرج النهائي؛ لهذا الغرض وبشكلٍ عام تُكرّر P_MD5 أكثر من P_SHA-1 لإنتاج كمية متساوية من البيانات التي تدخل لعملية XOR.



الشكل 7-7: دالة P_hash(secret, seed) لـ TLS.

❖ أكواد التنبيهات:

يدعم TLS كل أكواد التنبيهات المُعرَّفة في SSLv3 باستثناء no_certificate (لا توجد شهادة). هناك عدة أكواد إضافية معرَّفة في TLS؛ منها الأكواد الآتية التي تُعدُّ دائماً تنبيهات فادحة (fatal):

- decryption_failed (فشل إزالة التشفير): تم إزالة تشفير النص على نحو خاطئ؛ إما لكونه ليس من المضاعفات الزوجية لطول الكتلة، أو لأنه عند فحص قيم الحشوة (padding) وجد أنها خطأ.
- record_overflow (فيض السجل): تم استلام سجل TLS بحمولة (نص مُشفَّر) يزيد طولها عن 2^{14}+2048 بايتاً، أو أن النص الناتج من إزالة التشفير يزيد طوله عن 2^{14}+1024 بايتاً.
- unknown_ca (سلطة شهادات غير معروفة): تم استلام سلسلة أو سلسلة جزئية للشهادة صحيحة، ولكن الشهادة لم تُقبل نظراً لأنه تعذّر تحديد مكان سلطة الشهادات أو لعدم تطابقها مع سلطة شهادات معروفة ومؤتمنة.
- access_denied (ممنوع الوصول): تم استلام شهادة صحيحة، غير أنه عند تطبيق إجراء التحكم في الوصول قرَّر المُرسِل عدم الاستمرار في التفاوض.
- decode_error (خطأ في إزالة التكويد): تعذّر إزالة تكويد الرسالة نظراً لأن قيمة أحد الحقول تقع خارج المدى المحدّد أو أن طول الرسالة غير صحيح.
- export_restriction (قيود على التصدير): تم اكتشاف تفاوض على طول المفتاح غير متوافق مع القيود على التصدير.
- protocol_version (رقم إصدار البروتوكول): تم التعرف على رقم إصدار البروتوكول الذي يستخدمه الزبون، ولكن وُجد أنه غير مدعوم.

- insufficient_security (أمن غير كافٍ): يتم إرسالها بدلاً من handshake_failed عندما يفشل التفاوض بالتحديد لأن الخادم يتطلّب شفرةً أكثر أمناً من تلك التي يدعمها الزبون.
- internal_error (خطأ داخلي): طرأ خطأ داخلي غير مرتبط بالنظير أو بصحة البروتوكول، مما أدى إلى استحالة الاستمرار.

تتضمّن بقيّة التنبيهات الجديدة مايلي:

- decrypt_error (خطأ في إزالة التشفير): فشلت عملية من عمليات التشفير الخاصة بالمصافحة؛ بما في ذلك عدم التمكن من التحقق من التوقيع أو إزالة تشفير تبادل مفتاح أو التحقق من سلامة رسالة إنهاء.
- user_canceled (تم الإلغاء من قِبَل المستخدم): تم إلغاء هذه المصافحة لسببٍ ما لا علاقة له بفشل البروتوكول.
- no_renegotiation (لا إعادة للتفاوض): تُرسَل من قِبَل الزبون ردّاً على طلب ترحيب، أو تُرسَل من قِبَل الخادم ردّاً على رسالة ترحيب الزبون بعد المصافحة الأولى. ويؤدّي أيٌّ من هاتين الرسالتين في العادة إلى إعادة التفاوض، ولكن رسالة التنبيه هذه تشير إلى أن المُرسِل غيرُ قادر على إعادة التفاوض. وهذه الرسالة دائماً تحذيرية.

مجموعات التشفير (cipher suites): هناك عدّة اختلافات طفيفة بين مجموعات التشفير المتوفرة في كلٍّ من SSLv3 وTLS:

- تبادل المفاتيح: يدعم TLS كلّ أساليب تبادل المفاتيح المتوفرة في SSLv3 ما عدا Fortezza.
- خوارزميات التشفير المتماثل: يتضمّن TLS كل خوارزميات التشفير المتماثل الموجودة في SSLv3 ما عدا Fortezza.

❖ أنواع شهادات الزبون:

يُعرِّف TLS أنواع الشهادات الآتية التي يمكن أن تُطلب في رسالة طلب شهادة (certificate_request): rsa_sig، dss_sign، rsa_fixed_dh، dss_fixed_dh. وكل هذه الأنواع مُعرَّفة في SSLv3. بالإضافة لذلك يتضمن SSLv3 الأنواع الآتية: rsa_ephemeral_dh، dss_ephemeral_dh، fortezza_kea. تتضمّن طريقة ديفي - هيلمان العابرة توقيع مُعامِلات ديفي - هيلمان بـ RSA أو DSS؛ لكن في TLS يُستخدم rsa_sign أو dss_sign لهذا الغرض، ولا يلزم استخدام نوع توقيع مستقل لتوقيع مُعامِلات ديفي - هيلمان. لا يتضمّن TLS طريقة Fortezza.

❖ رسائل التحقق من الشهادة ورسائل الإنهاء:

في رسالة التحقق من الشهادة بـ TLS، تُحسب أكواد MD5 وSHA-1 للتحوير فقط على رسائل المصافحة. تذكّر أنه في SSLv3 يتضمن حساب كود التحوير أيضاً السرّ الرئيس وبايتات الحشو. وهناك إحساس بأن هذه الحقول الإضافية لا تؤدي إلى أي أمن إضافي.

كما هو الحال مع رسالة الإنهاء في SSLv3، رسالة الإنهاء في TLS هي عبارة عن كود تحوير يعتمد على السرّ الرئيس المشترك، ورسائل المصافحة السابقة، والوسمة التي تميّز الزبون من الخادم. ويختلف الحساب نوعاً ما، ففي TLS:

```
PRF(master_secret, finished_label, MD5(handshake_messages) ||
    SHA-1(handshake_messages))
```

حيث finished_label هي سلسلة البتات التي تمثل حالة "الزبون انتهى" لدى الزبون وحالة "الخادم انتهى" لدى الخادم.

❖ حسابات التشفير:

يُحسب السرّ قبل الرئيس في TLS بنفس الطريقة المستخدمة في SSLv3. كما في SSLv3، يُحسب السرّ الرئيس في TLS كناتج دالة التحوير على السرّ قبل

الرئيس والعددين العشوائيين في رسالتي الترحيب. ويختلف شكل حساب TLS عن SSLv3، ويُعرَّف على النحو الآتي:

```
master_secret = PRF(pre_master_secret, "master secret",
                    ClientHello.random || ServerHello.random)
```

يُكرر تطبيق الخوارزمية حتى تنتج مُخرجاً شبه عشوائي يتألف من 48 بايتاً. ويُعرَّف حساب مادة كتلة المفتاح (مفاتيح ماك السرية، ومفاتيح تشفير الجلسة، ومتجهات التهيئة كالتالي:

```
key_block = PRF(master_secret, "key expansion",
                SecurityParameters.server_random ||
                SecurityParameters.client_random)
```

وتكرار ذلك إلى أن يتولد ناتجٌ كافٍ. كتلة المفتاح (key_block) في SSLv3 دالة في السرّ الرئيس والأعداد العشوائية للزبون والخادم؛ لكن تختلف الخوارزمية الفعلية في TLS.

❖ الحشوة (padding):

كمية الحشوة المضافة في SSL قبل تشفير بيانات المُستخدِم هي الحدّ الأدنى المطلوب لكي يصبح الحجم الكلي للبيانات التي ستُشفَّر من مضاعفات حجم كتلة الشفرة. أما في TLS فيمكن أن تكون الحشوة المضافة بأي كمية تجعل المجموع من مضاعفات حجم كتلة الشفرة، بما لا يزيد على 255 بايتاً. على سبيل المثال، إذا كان حجم النص الأصلي (plaintext) (أو النص المضغوط في حالة استخدام الضغط) مضافاً إليه الماك وبايت طول الحشوة (padding.length) يساوي 79 بايتاً، فإن طول الحشوة بالبايتات يمكن أن يكون 1 أو 9 أو 17 وهكذا حتى 249. قد يُستخدم طول حشوة متغيّر لإحباط الهجمات المعتمدة على تحليل أطوال الرسائل المتبادلة.

3-7 المُعامَلات الإلكترونية الآمنة (SET)

المُعامَلات الإلكترونية الآمنة (SET) هي نظام مفتوح للتشفير والأمن صُمِّم لحماية استخدام بطاقات الائتمان على الإنترنت. والنسخة الحالية منه هي SETv1، وقد ظهرت لتلبية الحاجة إلى معايير أمن من قِبَل شركات بطاقات الائتمان الماستر كارد (MasterCard) والفيزا (Visa) في فبراير/شباط عام 1996. وشاركت في تطوير المواصفات الأولية مجموعة كبيرة من الشركات كان من بينها: IBM، ومايكروسوفت، ونيتسكيب، وRSA، وTerisa، وVerisign. وبدأت منذ عام 1996 عدة محاولات لاختبار تلك الفكرة، وبحلول عام 1998 ظهرت أول موجة من المنتجات المتوافقة مع نظام SET.

لا يُعدُّ نظام SET في حد ذاته نظاماً للدفع (أي تسديد المبالغ المستحقة)؛ ولكنه بالأحرى مجموعة من بروتوكولات وصيغ الأمن التي تمكّن المستخدمين من استعمال البنية التحتية لأنظمة الدفع ببطاقات الائتمان الحالية على شبكة مفتوحة كالإنترنت بشكلٍ آمنٍ. يوفر SET ثلاث خدمات:

- قناة اتصال آمنة بين جميع الأطراف المشتركة في المُعامَلة.
- نظام للثقة باستخدام X.509 للشهادات الرقمية.
- ضمان الخصوصية لأن المعلومات تكون متوفرة فقط للأطراف المعنية بتلك المُعامَلة عندما وحيثما يكون ذلك ضرورياً.

مواصفات SET معقّدة ومُعرَّفة في ثلاثة كتب صدرت في مايو عام 1997:

- الكتاب الأول: وصف للمتطلبات التجارية (80 صفحة).
- الكتاب الثاني: دليل المبرمِج (629 صفحة).
- الكتاب الثالث: التعريف الرسمي للبروتوكول (262 صفحة).

أي ما يُشكِّل في مجموعه 971 صفحة من المواصفات. في المقابل، كانت مواصفات SSLv3 مؤلَّفة من 63 صفحة، بينما تألَّفت مواصفات TLS من 80 صفحة.

ولذا فسنقتصر في هذا الجزء فقط على إعطاء ملخص سريع لهذا المعيار متعدد الجوانب.

7-3-1 نظرة عامة على SET

من الطرق الجيدة لبدء مناقشة SET أن ننظر إلى المتطلبات التجارية له وميزاته الرئيسة والمشاركين في مُعامَلاته.

❖ **المتطلبات:**

يعرض الكتاب الأول للمواصفات المتطلبات التجارية الآتية لأنظمة الدفع الآمنة ببطاقات الائتمان على الإنترنت والشبكات الأخرى:

- توفير السرية لمعلومات الدفع والطلبية: من الضروري لحاملي البطاقات الاطمئنان إلى أن هذه المعلومات آمنة ويمكن الوصول إليها فقط بواسطة المستلم المعنيّ. وتقلل السرية أيضاً خطر الاحتيال بواسطة أيٍّ من الأطراف المشتركة في المُعامَلة أو من قِبَل أطراف ثالثة مؤذية. ويستخدم SET التشفير لتوفير السرية.
- ضمان سلامة البيانات المُرسَلة: أي ضمان عدم حدوث تغييرات (دون اكتشافها) في محتوى رسائل SET أثناء انتقالها. وتُستخدَم التوقيعات الرقمية لضمان ذلك.
- توفير آليات للتحقق من أن حامل البطاقة هو مُستخدِم مُصرَّح له: إن وجود آلية للربط بين حامل البطاقة ورقم حساب معيّن تقلل من حوادث الاحتيال والكلفة الإجمالية لمعالجة عملية الدفع. وتستخدم التوقيعات الرقمية والشهادات لإثبات أن حامل البطاقة هو المُستخدِم الحقيقي لحساب صحيح.
- توفير آليات للتحقق من أن التاجر يقبل الدفع ببطاقة الائتمان من خلال علاقته بمؤسسة مالية: يُعدُّ ذلك تكملةً للمطلب السابق. فمن الضروري أن يكون حاملو البطاقات قادرين على تمييز التجار الذين يمكنهم إجراء

مُعامَلات آمنة معهم. ومرة أخرى تُستخدَم التوقيعات الرقمية والشهادات لهذا الغرض.

- ضمان استخدام أفضل الممارسات الأمنية وأساليب تصميم النظام لحماية كل الأطراف الشرعية في صفقة تجارة عبر الإنترنت: يُعدُّ SET مواصفاتٍ مجرّبة بشكلٍ جيد ومبنيةٍ على خوارزميات وبروتوكولات تشفير آمنة للغاية.
- عدم اعتماد البروتوكول على استخدام آليات النقل الآمن، وكذلك عدم منع استخدامها: يمكن تشغيل SET بشكلٍ آمن على رصّة بروتوكولات TCP/IP. ومع ذلك لا يتداخل SET مع استخدام آليات الأمن الأخرى مثل IPSec وSSL/TLS.
- تسهيل العمل البيني بين موفري البرمجيات وموفري خدمة الشبكة وتشجيعه: ينبغي ألا تعتمد بروتوكولات وصيغ SET على المكونات المادية ونظم التشغيل وبرامج الويب.

7-3-2 الميزّات الرئيسة لنظام SET

لتلبية المتطلبات التي عرضناها آنفاً، يتضمن SET الميزّات الآتية:

- سرية المعلومات: يتم تأمين حساب حامل البطاقة ومعلومات الدفع أثناء انتقالها عبر الشبكة. من الميزات المثيرة المهمة لـ SET أنه يمنع التاجر من معرفة رقم بطاقة الائتمان لحامل البطاقة؛ هذا شأن خاص فقط بالمصرف الذي أصدرها. ويُستخدَم التشفير التقليدي بواسطة DES لتوفير السرية.
- سلامة البيانات: تتضمّن معلومات الدفع المُرسَلة من حاملي البطاقات إلى التجار معلومات الطلبية، وبيانات شخصية، وتعليمات الدفع. يوفر SET ضماناً بعدم تعديل محتويات تلك الرسائل أثناء انتقالها. وتُستخدم توقيعات RSA الرقمية بكود التحوير SHA-1 وسيلةً لفحص سلامة الرسائل. ويتم أيضاً حماية رسائل معينة عن طريق HMAC باستخدام SHA-1.

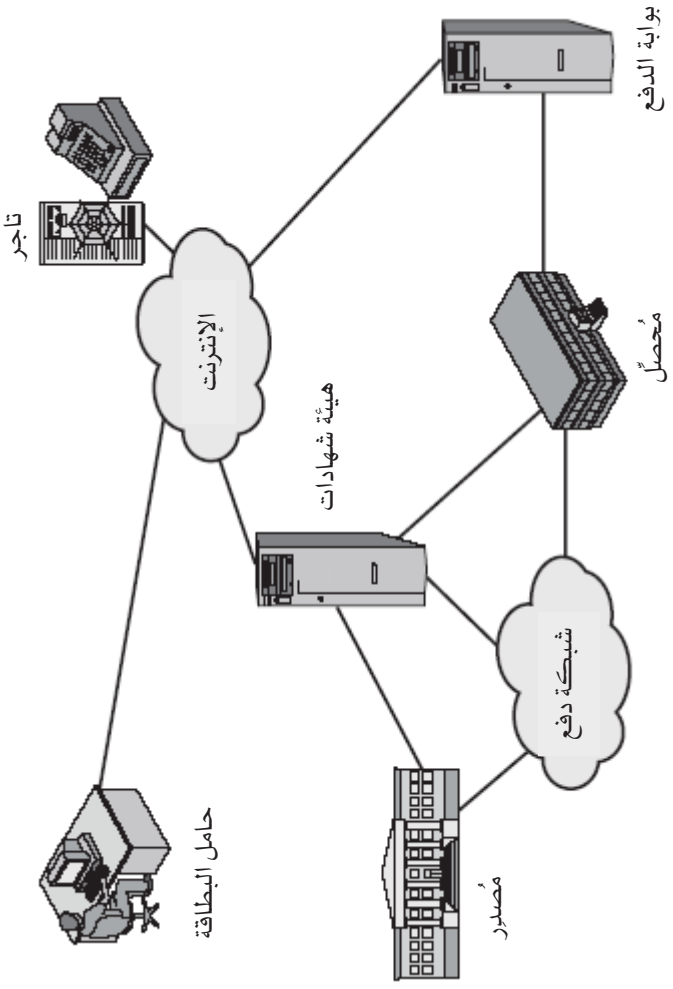
- التحقّق من حساب حامل البطاقة: يُمكِّن SET التجار من التحقق من أن حامل البطاقة هو المُستخدِم الحقيقي لرقم حساب بطاقة صحيح. ويستخدم SET لهذا الغرض شهادات X.509v3 الرقمية مع توقيعات RSA.
- التحقّق من التاجر: يُمكِّن SET حاملي البطاقات من التحقق من أن التاجر له علاقة بمؤسسة مالية تسمح له بقبول بطاقات الدفع. مرة أخرى يستخدم SET لهذا الغرض شهادات X.509v3 الرقمية بتوقيعات RSA.

لاحظ أنه على خلاف IPSec وSSL/TLS، يوفر SET خياراً واحداً فقط لكل خوارزمية تشفير. وهذا أمر متوقّع، فنظام SET يخدم تطبيقاً واحداً فقط يتضمّن مجموعة واحدة من المتطلبات، بينما يهدف IPSec وSSL/TLS إلى دعم تشكيلة من التطبيقات المختلفة.

❖ الأطراف المشاركة في SET:

يبيّن الشكل 7-8 الأطراف المشاركة في SET، وتشمل ما يلي:

- حامل البطاقة (cardholder): في البيئة الإلكترونية، يتفاعل العميل (المستهلك أو المشتري) مع التاجر من خلال الحاسبات الشخصية المتصلة بالإنترنت. حامل البطاقة هو حاملٌ مصرحٌ له لبطاقة دفع (كبطاقة الماستر كارد أو الفيزا) يُصدرها مُصدِر (issuer).
- التاجر (merchant): هو شخصٌ أو منظمة لديها سلعٌ أو خدمات للبيع لحاملي بطاقات الدفع. في العادة تُعرض هذه السلع أو الخدمات عن طريق موقع ويب أو رسائل البريد الإلكتروني. ويجب أن يكون للتاجر الذي يقبل بطاقات الدفع علاقة بمُحصِّل (acquirer).
- المُصدِر (issuer): هو مؤسسة مالية (كمصرف) توفر بطاقات الدفع لحاملي البطاقات. في العادة يتم التقدم لفتح الحسابات وفتحها إما بالبريد أو شخصياً. في النهاية يتحمّل المُصدِر مسؤولية دفع دين حامل البطاقة.



الشكل 7-8: مكوِّنات التجارة الإلكترونية الآمنة.

- المُحصِّل (acquirer): هو المؤسسة المالية التي تؤسّس حساباً للتاجر وتقوم بمعالجة عمليات التفويض والدفع بالبطاقات. وعادةً ما يقبل التجار أكثر من نوع من بطاقات الائتمان لكنهم لا يرغبون في التعامل بشكلٍ فردي مع هيئات مختلفة للبطاقات أو مع مُصدِرين متعددين. ويوفر المُحصِّل للتاجر خدمة التحقق من أن حساب البطاقة نشِط وأن مبلغ الشراء المقترح لا يتجاوز حدّ الائتمان المصرّح به. ويقوم المُحصِّل أيضاً بتحويل المبلغ المستحق لحساب التاجر إلكترونياً. وبعد ذلك، يُعوَّض المحصِّل من قبل المُصدِر عبر شبكةٍ من شبكات الدفع لتحويل الأموال بشكلٍ إلكتروني.
- بوابة الدفع (payment gateway): هي وظيفة يقوم بها المُحصِّل أو طرف ثالث معيّن، وذلك لمعالجة رسائل الدفع من التجار. وتمثل بوابة الدفع واجهة بين SET وشبكات الدفع الحالية للبطاقات المصرفية، وتقوم بوظائف الدفع والتفويض به. يتبادل التاجر رسائل SET مع بوابة الدفع على الإنترنت، بينما تتصل بوابة الدفع مباشرةً أو من خلال شبكة بنظام المعالجة المالية للمُحصِّل.
- سلطة الشهادات (CA): هي كيان مؤتمن لإصدار شهادات المفاتيح العامة X.509v3 لحاملي البطاقات والتجار وبوابات الدفع. يعتمد نجاح SET على توفر بنية تحتية لسلطات الشهادات لهذا الغرض. وكما نوقش في الفصول السابقة يُستخدم تركيب هرمي من سلطات الشهادات، بحيث لا يحتاج المشاركون للحصول على شهاداتهم مباشرةً من هيئة الجذر (root authority).

نستعرض الآن باختصار تسلسل الأحداث المطلوب لإجراء مُعامَلة، وبعد ذلك نناقش بعض تفاصيل التشفير.

1. يقوم الزبون بفتح حساب: يحصل الزبون على حساب بطاقة ائتمان (كالماستر كارد أو الفيزا) من مصرف يدعم الدفع الإلكتروني ونظام SET.

2. يستلم الزبون شهادة: بعد التحقق المناسب من هوية الزبون، يستلم الزبون شهادة رقمية X.509v3 موقَّعة من المصرف. توثق الشهادة مفتاح RSA العام للزبون وتاريخ انتهائه. أيضاً يتم تأسيس علاقة (يضمنها المصرف) بين زوج المفاتيح للزبون وبطاقته الائتمانية.
3. للتجار شهاداتهم الخاصة: التاجر الذي يقبل نوعاً معيناً من البطاقات يجب أن يكون لديه شهادتان لمفتاحين عامين خاصين به: واحد لتوقيع الرسائل، وآخر لتبادل المفاتيح. ويحتاج التاجر أيضاً لنسخة من شهادة المفتاح العام لبوابة الدفع.
4. يقوم الزبون بطلبية ما: قد تتضمّن هذه العملية أن يستعرض الزبون موقع الويب الخاص بالتاجر أولاً لاختيار المواد وتحديد أسعارها. ثم يرسل الزبون قائمة المواد التي سيشتريها للتاجر الذي يرُدّ بإرسال استمارة طلبية تتضمن قائمة بالمواد المطلوبة وأسعارها والسعر الإجمالي ورقم الطلبية.
5. يتم التحقق من هوية التاجر: بالإضافة إلى نموذج الطلبية، يُرسل التاجر نسخةً من شهادته، لكي يتمكن الزبون من التحقق من أنّه يتعامل مع متجر غير مزيف.
6. يتم إرسال بيانات الطلبية والدفع: يرسل الزبون المعلومات المتعلقة بكلٍّ من الطلبية والدفع للتاجر مع شهادة الزبون. تؤكد معلومات الطلبية شراء المواد المذكورة في نموذج الطلبية، بينما تتضمّن معلومات الدفع تفاصيل بطاقة الائتمان. تكون معلومات الدفع مُشفَّرة بحيث لا يمكن للتاجر قراءتها. تُمكِّن شهادة الزبون التاجر من التحقق من هوية الزبون.
7. يطلب التاجر ترخيصاً بالدفع: يرسل التاجر معلومات الدفع إلى بوابة الدفع، ويطلب تأكيداً بأن الحد الائتماني المتوفر للزبون كافٍ لعملية الشراء تلك.
8. يرسل التاجر إلى الزبون إشعاراً باستلام بيانات الطلبية.
9. يشحن التاجر السلع أو يوفر الخدمة المطلوبة للزبون.
10. يطلب التاجر الدفع: يُرسَل هذا الطلب إلى بوابة الدفع التي تقوم بكل عمليات المعالجة اللازمة لإتمام عملية الدفع.

7-3-3 التوقيع المزدوج (Dual Signature)

قبل النظر في تفاصيل بروتوكول SET، دعنا نناقش أسلوباً جديداً مهماً هو التوقيع المزدوج الذي تم استحداثه للاستخدام في SET. والهدف من التوقيع المزدوج هو الربط بين رسالتين ترسلان لمستلمين مختلفين. وفي الحالة التي نحن بصددها، يرغب الزبون في إرسال معلومات الطلبية (OI) للتاجر وإرسال معلومات الدفع (PI) للمصرف. ولا يحتاج التاجر لمعرفة رقم بطاقة الائتمان، ولا يحتاج المصرف لمعرفة تفاصيل الطلبية. ويتوفر للزبون حماية إضافية من حيث السرية وذلك بجعل كلٍّ من هذين العنصرين مستقلاً عن الآخر. ومع ذلك ينبغي الربط بين هذين العنصرين بطريقة ما بحيث يمكن أن تستخدم لحل النزاعات عند الضرورة. وهذا الربط مطلوب حتى يمكن أن يثبت الزبون أن عملية الدفع هذه مقصودة لهذه الطلبية بعينها وليست لسلعة أو خدمة أخرى.

لإدراك مدى الحاجة لهذا الربط، افترض أن الزبون أرسل رسالتين للتاجر: رسالة OI موقَّعة ورسالة PI مُوقَّعة. وافترض أن التاجر مرّر رسالة PI تلك إلى المصرف. وإذا تمكن التاجر من التقاط رسالة OI أخرى من هذا الزبون، فإنه بوسعه أن يدّعي بأنّ رسالة الـ OI الأخيرة، وليس رسالة الـ OI الأصلية، خاصة بذلك الـ PI . تمنع عملية الربط حدوث ذلك.

يبيّن الشكل 7-9 استخدام التوقيع المزدوج لتلبية المطلب في الفقرة السابقة. يأخذ الزبون كود التحوير (باستخدام SHA-1) لكلٍ من PI وOI. ثم يوصل هذين الكودين ويأخذ كود التحوير للناتج. وأخيراً، يُشفّر الزبون كود التحوير النهائي بمفتاحه الخاص لتوقيعه، ومن ثمَّ ينتج التوقيع المزدوج. ويمكن تلخيص تلك العملية على النحو الآتي:

$$DS = E(PR_c, [H(H(PI)||H(OI))])$$

حيث تمثل PR_c المفتاح الخاص لتوقيع الزبون. الآن افترض أن التاجر بحوزته التوقيع المزدوج (DS) وOI وPIMD (أي خلاصة رسالة PI). وافترض أيضاً أن التاجر لديه أيضاً المفتاح العام للزبون ويمكنه الحصول عليه من شهادة الزبون. عندئذ يمكن أن يحسب التاجر القيم:

H(PIMD||H[OI]); D(PU_c, DS)

حيث PU_c هو المفتاح العام لتوقيع الزبون. إذا كانت هاتان القيمتان متساويتين، يكون التاجر قد تحقق من صحة التوقيع. بنفس الطريقة، إذا كان لدى المصرف DS وPI وOIMD (خلاصة رسالة OI) ومفتاح الزبون العام، عندئذ يمكن أن يحسب المصرف:

H(H[OI]||OIMD); D(PU_c, DS)

مرة أخرى إذا تساوت هاتان القيمتان، يكون المصرف قد تحقق من صحة التوقيع.

الخلاصة:

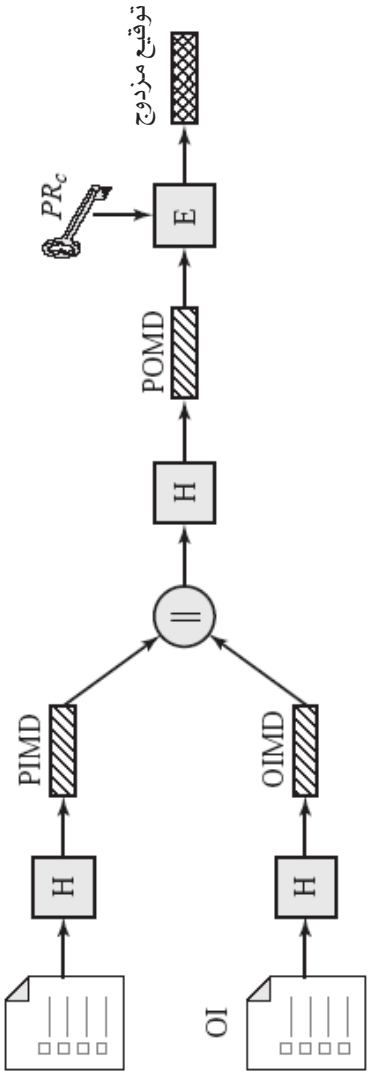
1. استلم التاجر OI وتحقق من صحة التوقيع.
2. استلم المصرف PI وتحقق من صحة التوقيع.
3. ربط الزبون بين OI وPI، ويمكنه إثبات الصلة بينهما.

على سبيل المثال، افترض أن التاجر يرغب في استبدال OI آخر في هذه المُعامَلة لغرضٍ ما يعود عليه بفائدة. عليه عندئذ أن يجد OI آخر له كود تحوير متوافق مع OIMD الحالية. عند استخدام SHA-1، يُعدُّ هذا الأمر غير ممكن. وعليه، لا يستطيع التاجر ربط OI آخر بذلك الـ PI.

4-3-7 معالجة عملية الدفع

يعرض الجدول 7-3 قائمةً بأنواع المُعامَلات التي يدعمها SET. سنناقش بعض تفاصيل المُعامَلات الآتية:

- طلب الشراء (purchase request).
- تصريح الدفع (payment authorization).
- تحصيل المبلغ المستحق (payment capture).



الشكل 7-9: إنشاء التوقيع المزدوج.

الجدول 7-3: أنواع مُعامَلات SET.

تسجيل حامل البطاقة	يجب على حاملي البطاقات التسجيل لدى CA قبل أن يمكنهم إرسال رسائل SET للتجار.
تسجيل التاجر	يجب أن يسجل التجار مع CA قبل أن يمكنهم تبادل رسائل SET مع الزبائن وبوابات الدفع.
طلب شراء	رسالة من الزبون إلى التاجر تتضمن معلومات طلب الشراء (OI) للتاجر ومعلومات الدفع (PI) للمصرف.
تفويض الدفع	تبادل بين التاجر وبوابة الدفع للتخويل بدفع مبلغ الشراء من حساب بطاقة الائتمان المُعطى.
تحصيل المبلغ المستحق	تسمح للتاجر بطلب مبلغ الشراء من بوابة الدفع.
استفسار عن حالة طلب شهادة	إذا لم تتمكن CA من إكمال معالجة طلب الشهادة (certificate request) بسرعة، تقوم بإرسال رد لحامل البطاقة أو التاجر تشير بأن عليه مراجعتها مرة أخرى لاحقاً. بعد ذلك يقوم حامل البطاقة أو التاجر بإرسال استفسار (certificate inquiry) لتحديد حالة طلب الشهادة واستلامها إذا كان قد تم التصديق على الطلب.
استفسار عن شراء	تسمح لحامل البطاقة بالاستفسار عن حالة أمر شراء بعد استلام الرد على أمر الشراء. لاحظ أن هذه الرسالة لا تتضمّن معلومات مثلاً عن حالة السلع المتأخرة وقام التاجر بطلبها، لكنها تشير إلى حالة التفويض بالدفع وتحصيل المبلغ المستحق.
تصحيح تفويض الدفع	تسمح للتاجر بتصحيح طلبات التفويض السابقة. إذا كان من المتعذر إكمال الطلبية، يقوم التاجر بتصحيح التفويض بالكامل. أما إذا تعذّر إكمال جزء من الطلبية (مثلاً عند عدم توفر سلع مطلوبة مؤقتاً لدى التاجر ولكنه قام بطلبها)، يقوم التاجر بتصحيح ذلك الجزء وحده فقط.
تصحيح المبلغ المستحق	تسمح للتاجر بتصحيح الأخطاء في طلبات التحصيل (كقيمة المبلغ المستحق) التي قد تنجم عن خطأ في الإدخال من قِبَل الموظف.
تقييد مبلغ لحساب العميل	تسمح للتاجر بتقيد مبلغ لحساب حامل البطاقة، وذلك في حالات، منها: إرجاع بعض السلع أو تلفها أثناء الشحن. لاحظ أن رسالة SET الخاصة بذلك تبدأ دائماً من قِبَل التاجر وليس حامل البطاقة. تتم كافة الاتصالات بين حامل البطاقة والتاجر وهي تؤدي إلى إجراء معالجة لتقييد مبلغ لحساب حامل البطاقة خارج SET.
تصحيح مبلغ الاعتماد	تسمح للتاجر بتصحيح مبلغ الاعتماد سبق للتاجر أن طلب تقييده لحساب العميل.
طلب شهادة بوابة الدفع	تسمح للتاجر بطلب شهادات التوقيع واستلامها وتبادل المفتاح من بوابة الدفع
إدارة بالدفعات (batch administration)	تسمح للتاجر بإرسال معلومات عن عدة أوامر شراء تقدّم دفعة واحدة إلى بوابة الدفع.
رسالة خطأ	رسالة تدل على حدوث خطأ عند اختبار محتوى أو صيغة رسالة مُستلَمه.

❖ طلب الشراء:

قبل أن يبدأ تبادل طلب الشراء، يجب أن يكون حامل البطاقة قد انتهى من التصفح والاختيار والطلب. وتأتي هذه المرحلة التمهيدية إلى نهايتها عندما يرسل التاجر نموذج طلب شراء مكتمل إلى الزبون. وتتم كل المراسلات السابقة بدون استخدام SET.

يشتمل تبادل طلب الشراء على أربع رسائل: طلب استهلالي، ردّ استهلالي، طلب الشراء، ردّ الشراء.

لكي يرسل حامل البطاقة رسائل SET للتاجر، يجب أن يكون لديه نسخة من شهادة التاجر وشهادة بوابة الدفع. ويطلب الزبون الشهادتين في رسالة الطلب الاستهلالي التي تُرسَل للتاجر. وتتضمّن هذه الرسالة نوع بطاقة الائتمان التي يستخدمها الزبون، كما تتضمّن رقماً تعريفياً (ID) مخصصاً من قبل الزبون لهذا الزوج من الطلب والردّ، بالإضافة إلى nonce يُستخدم لضمان الوقتية (timeliness) للرسالة.

يولّد التاجر ردّه ويوقّعه بمفتاح توقيعه الخاص. يتضمّن الردّ الـ nonce الذي أرسله الزبون، وnonce آخر إلى الزبون ليرده في الرسالة التالية، ورقماً تعريفياً لمعاملة الشراء هذه. بالإضافة إلى الردّ الموقّع، تتضمّن رسالة الردّ الاستهلالي شهادة توقيع التاجر وشهادة تبادل المفاتيح لبوابة الدفع.

يتحقق حامل البطاقة من صحة تلك الشهادات الخاصة بالتاجر والبوابة عن طريق توقيعات سلطات الشهادات لهما، وبعدها يُنشيء OI وPI. يوضع الرقم التعريفي المخصص لتلك المعاملة من قِبَل التاجر في كلٍّ من OI وPI. لا يحتوي OI بشكلٍ صريح على بيانات الطلب كعدد كل بند من المواد وسعره، ولكنّه يتضمّن إشارةً للطلب تم الحصول عليها في التبادل بين التاجر والزبون أثناء مرحلة التسوق التي تمت قبل رسالة SET الأولى. بعد ذلك يُعِدّ حامل البطاقة رسالة طلب الشراء

(الشكل 7-10). لهذا الغرض، يولّد حامل البطاقة مفتاح تشفير متماثل للاستخدام مرة واحدة K_s.

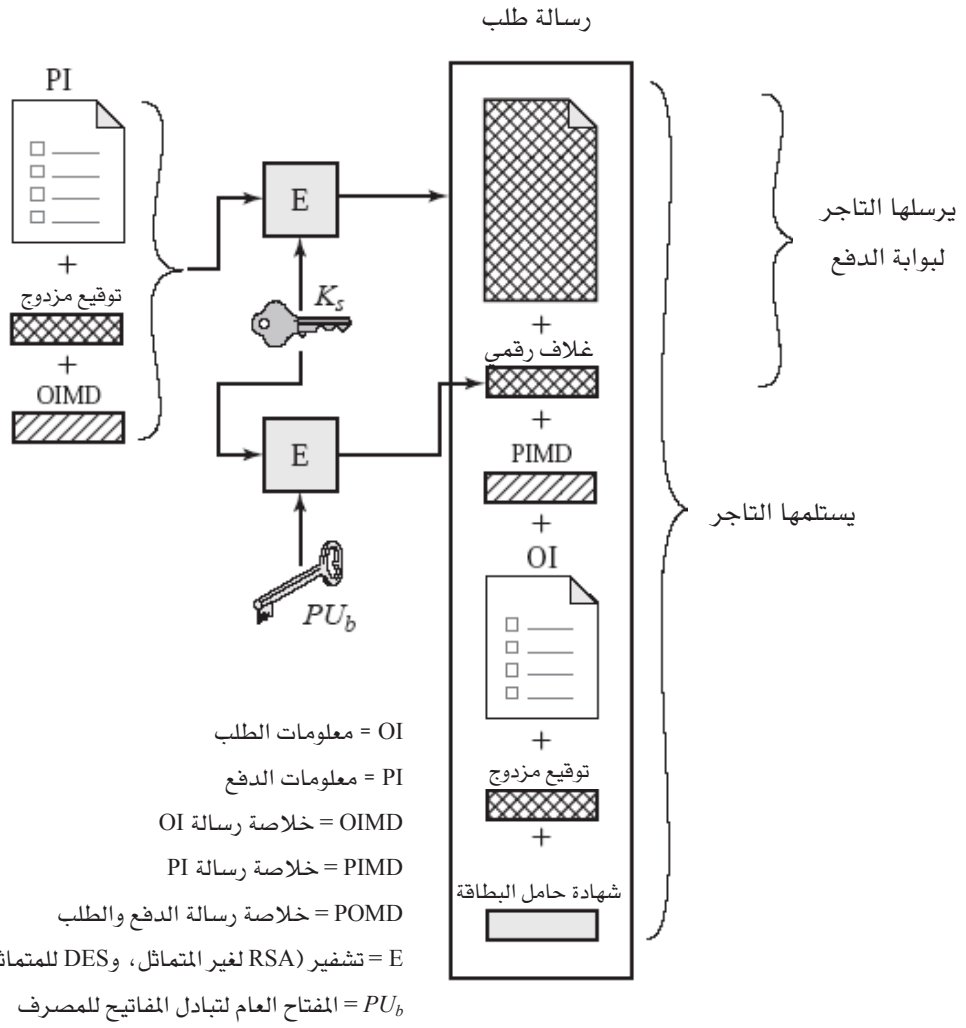
تتضمّن الرسالة ما يلي:

1. معلومات متعلقة بالشراء: يتم إرسال هذه المعلومات إلى بوابة الدفع بواسطة التاجر وتتكون من:
   - PI
   - التوقيع المزدوج: يُحسب من PI وOI ويوقّع بمفتاح توقيع الزبون الخاص
   - خلاصة رسالة OI (OIMD): وهي مطلوبة لبوابة الدفع للتحقق من صحة التوقيع المزدوج، كما أوضحنا آنفاً.

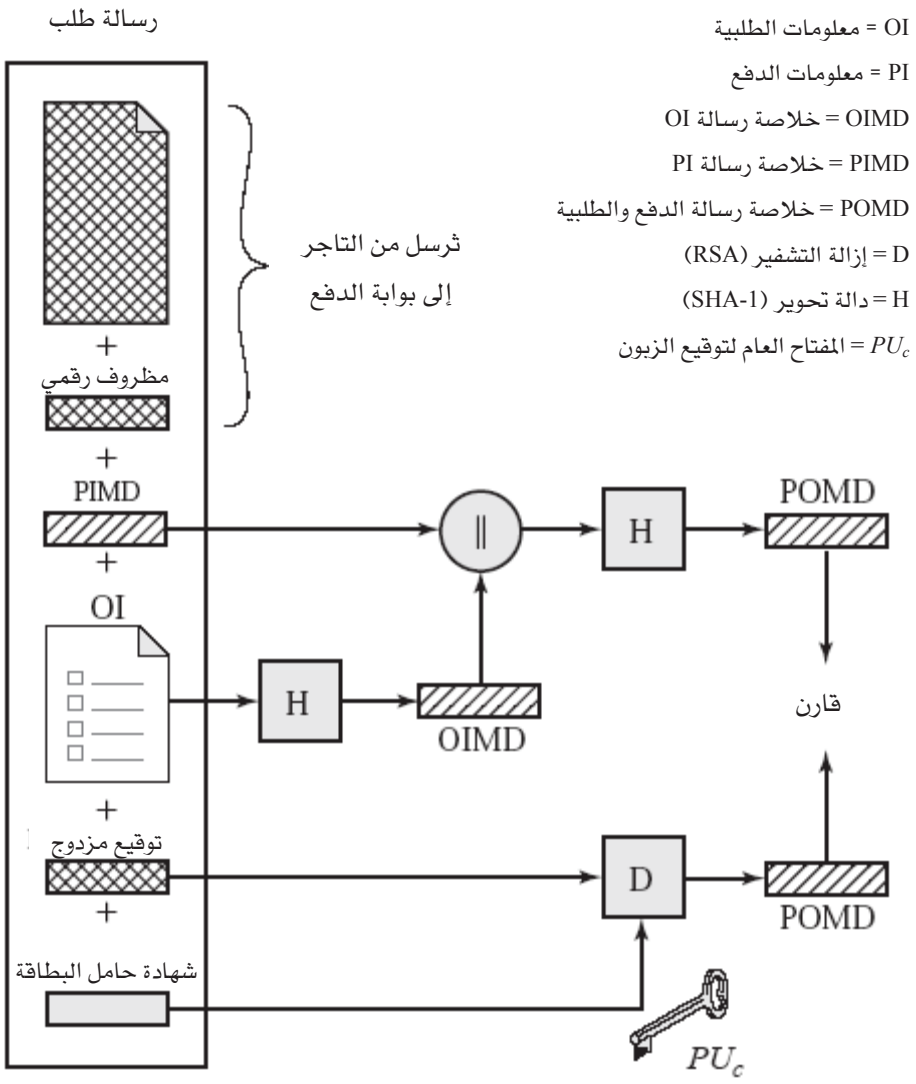
   يتم تشفير كل تلك العناصر بالمفتاح K_s. أما العنصر الأخير فهو:

   - الظرف الرقمي: ويتم حسابه بتشفير K_s بمفتاح تبادل المفاتيح العامة للبوابة. ويطلق عليه الظرف الرقمي لأنه يجب أن يُفتح (أي يُزال التشفير) قبل أن يتسنى قراءة أيٍّ من العناصر الأخرى السابقة.

   لاحظ أن قيمة K_s غير متوفرة للتاجر، ولذلك لا يستطيع التاجر قراءة أي من هذه المعلومات المتعلقة بالدفع.
2. معلومات متعلقة بالطلبية: هذه المعلومات مطلوبة للتاجر وتتضمّن:
   - OI: وتُرسَل بشكلٍ واضح (غير مُشفَّرة).
   - التوقيع المزدوج: ويحسب من PI وOI ويوقّع بمفتاح توقيع الزبون الخاص.
   - خلاصة رسالة PI (PIMD): وهي مطلوبة للتاجر لكي يتحقق من صحة التوقيع المزدوج.
3. شهادة حامل البطاقة: وتتضمّن مفتاح توقيع حامل البطاقة العام. وهي مطلوبة من قِبَل التاجر وبوابة الدفع.



الشكل 7-10: حامل بطاقة يرسل طلب شراء.



الشكل 7-11: تاجر يتحقق من طلب شراء من زبون.

عندما يستلم التاجر رسالة طلب الشراء، يقوم بالخطوات الآتية (الشكل 7-11):

1. يتحقق من شهادات حامل البطاقة عن طريق توقيع سلطة الشهادات لها.
2. يتحقق من التوقيع المزدوج باستخدام مفتاح توقيع الزبون العام. ويضمن ذلك أن الطلب لم يُعبث به أثناء النقل وأنه وُقِّع باستخدام مفتاح توقيع حامل البطاقة الخاص.
3. يعالِج الطلب ويرسل معلومات الدفع إلى بوابة الدفع للتصريح بالدفع (كما سنصف لاحقاً).
4. يرسل ردّ الشراء إلى حامل البطاقة.

تتضمّن رسالة ردّ الشراء كتلة ردّ للإشعار باستلام الطلب وإرجاع رقم تعريفي لتلك المعاملة. وهذه الكتلة موقَّعة من قِبَل التاجر باستخدام مفتاح توقيعه الخاص. وتُرسَل الكتلة وتوقيعها للزبون معاً مع شهادة توقيع التاجر.

عندما تتلقى برمجيات حامل البطاقة رسالة ردّ الشراء، تقوم بالتحقق من شهادة التاجر ثم تتحقق من توقيع كتلة الردّ. وأخيراً، تقوم البرمجيات بفعلٍ ما يعتمد على الردّ؛ كعرض رسالة للمستخدم أو تحديث قاعدة البيانات بحالة الطلبية.

❖ تفويض الدفع:

أثناء معالجة طلبٍ من حامل بطاقة، يُرخّص للتاجر بالمعاملة من بوّابة الدفع. ويضمن ترخيص الدفع أن المعاملة مسموحٌ بها من قِبَل المُصدِر. ويضمن هذا التصريح حصول التاجر على المبلغ المُستحق؛ لذا يمكن أن يوفر التاجر الخدمات أو السلع للزبون. ويشمل تبادل تفويض الدفع رسالتين: طلب التفويض وردّ التفويض.

يرسل التاجر رسالة طلب تفويض لبوابة الدفع تشمل ما يأتي:

1. معلومات متعلقة بالشراء: تم الحصول على هذه المعلومات من الزبون وتتكون من:
   - PI
   - التوقيع المزدوج: يُحسب من PI وOI ويوقَّع بمفتاح توقيع الزبون الخاص
   - خلاصة رسالة OI (OIMD)
   - الظرف الرقمي
2. معلومات متعلقة بالتفويض: هذه المعلومات يتم توليدها لدى التاجر وتتكون من:
   - كتلة التفويض: وتتضمّن الرقم التعريفي للمعاملة موقّعاً بمفتاح توقيع التاجر الخاص ومُشفّراً بمفتاح متماثل مولَّد من قِبَل التاجر للاستخدام لمرة واحدة .
   - ظرف رقمي: يتم إنشاؤه بتشفير مفتاح الاستخدام لمرة واحدة بمفتاح تبادل المفاتيح العامة لبوابة الدفع.
3. الشهادات: يُضمِّن التاجر شهادة مفتاح التوقيع لحامل البطاقة (تُستخدم للتحقق من التوقيع المزدوج)، وشهادة مفتاح توقيع التاجر (تُستخدم للتحقق من توقيع التاجر)، وشهادة تبادل المفاتيح للتاجر (مطلوبة في ردّ بوابة الدفع).

تقوم بوابة الدفع بالمهام الآتية:

1. التحقق من كل الشهادات.
2. إزالة تشفير الظرف الرقمي لكتلة التفويض للحصول على المفتاح المتماثل وبعد ذلك إزالة تشفير كتلة التفويض.
3. التحقق من توقيع التاجر على كتلة التفويض.

4. إزالة تشفير الظرف الرقمي لكتلة الدفع للحصول على المفتاح المتماثل وبعد ذلك إزالة تشفير كتلة الدفع.
5. التحقق من التوقيع المزدوج على كتلة الدفع.
6. التحقق من أن الرقم التعريفي للمعاملة الذي يُستلم من التاجر يطابق الموجود في PI القادم (بشكلٍ غير مباشر) من الزبون.
7. تطلب وتستلم تفويضاً بالدفع من المُصدِر.

بعد أن حصلت بوابة الدفع على التفويض من المُصدِر، ترسل رسالة ردّ التفويض للتاجر تتضمّن العناصر الآتية:

1. معلومات متعلقة بالتفويض: تتضمّن كتلة التفويض موقَّعةً بمفتاح توقيع البوابة الخاص ومُشفَّرةً بمفتاح متماثل يُستخدم لمرة واحدة ومولَّد بالبوّابة. وتتضمّن أيضاً ظرفاً رقمياً يشتمل على المفتاح الذي يُستخدَم لمرة واحدة مُشفَّراً بمفتاح التاجر لتبادل المفاتيح العامة.
2. معلومات توكن التحصيل (capture token): تُستخدم هذه المعلومات لإنجاز الدفع لاحقاً. ولهذه الكتلة نفس شكل المعلومات في (1) (أي رسالة توكن موقَّعة ومُشفَّرة مع مظروف رقمي). ولا يُعالج هذا التوكن من قِبَل التاجر. ولكن ينبغي إعادته كما هو مع طلب الدفع.
3. الشهادة: شهادة مفتاح توقيع البوابة.

بهذا التفويض من البوّابة، يمكن أن يورّد التاجر السلع أو يوفر الخدمة للزبون.

❖ تحصيل المبلغ المستحق:

لكي يحصل التاجر على المبلغ المستحق، يطلب من بوابة الدفع القيام بمُعامَلة تحصيل تتضمّن طلب التحصيل ورسالة ردّ التحصيل.

لإنشاء رسالة طلب التحصيل، يولِّد التاجر ويُوقِّع ويُشفِّر كتلة طلب التحصيل التي تتضمّن قيمة الدفع ورقم تعريف المُعامَلة. وتتضمّن الرسالة أيضاً توكن التحصيل مُشفَّراً، وقد تم استلامه في وقت سابق (ضمن ردّ التفويض) لهذه المُعامَلة، بالإضافة إلى شهادات مفتاح توقيع التاجر ومفتاح تبادل المفاتيح.

عندما تتلقّى بوابة الدفع رسالة طلب التحصيل، تقوم بإزالة تشفير كتلة طلب التحصيل والتحقق منها، وإزالة تشفير كتلة توكن التحصيل والتحقق منها. بعد ذلك تقوم بفحص الاتساق (consistency) بين طلب التحصيل وتوكن التحصيل، ثم تقوم بإنشاء طلب تسديد يُرسَل للمُصدِر على شبكة الدفع الخاصة. ويترتب على هذا الطلب تحويل المبلغ إلى حساب التاجر.

عندئذ تُخطِر البوابة التاجر بالدفع في رسالة ردّ التحصيل. وتتضمّن الرسالة كتلة ردّ التحصيل موقَّعة ومشفَّرة من قِبَل البوابة. وتتضمّن الرسالة أيضاً شهادة مفتاح توقيع البوابة. وتخزّن برمجيات التاجر ردّ التحصيل لاستخدامه لمطابقة المبلغ المدفوع الذي تم استلامه من المُحصِّل.

4-7 توصيات للمطالعة

[DREW99] Drew, G. *Using SET for Secure Electronic Commerce.* Upper Saddle River, NJ: Prentice Hall, 1999.

[MACG97] Macgregor, R.; Ezvan, C.; Liguori, L.; and Han, J. *Secure Electronic Transactions: Credit Card Payment on the Web in Theory and Practice.* IBM RedBook SG24-4978-00, 1997. Available at www.redbooks.ibm.com.

[RESC01] Rescorla, E. *SSL and TLS: Designing and Building Secure Systems.* Reading, MA: Addison-Wesley, 2001.

5-7 مصادر للمعلومات على الويب

- صفحة SSL لدى شركة نيتسكيب: وتحتوي على مواصفات SSL.
- ميثاق أمن طبقة النقل (TLS Charter): يتضمّن أحدث مسوّدات الإنترنت (Internet drafts) وطلبات التعليق (RFCs) الخاصة ببروتوكول TLS.
- مشروع OpenSSL: وهو مشروع لتطوير برمجيات SSL وTLS مفتوحة المصدر. يتضمن الموقع مستندات وروابط لمواقع أخرى.

6-7 مصطلحات رئيسة

acquirer	مُحصِّل
cardholder	حامل البطاقة
Certification Authority (CA)	سلطة التوثيق (التصديق)
dual signature	توقيع مزدوج
issuer	مُصدِر
merchant	تاجر
payment gateway	بوابة الدفع
Secure Electronic Transaction (SET)	المُعامَلات الإلكترونية الآمنة (SET)
Secure Socket Layer (SSL)	طبقة المقابس الآمنة (SSL)
Transport Layer Security (TLS)	أمن طبقة النقل (TLS)

7-7 أسئلة للمراجعة ومسائل

7-7-1 أسئلة للمراجعة

7-1 ما مميزات كلٍّ من الطرق الثلاثة الموضحة بالشكل 7-1؟

7-2 اذكر البروتوكولات التي يتكون منها SSL.

7-3 ما الفرق بين توصيلة SSL وجلسة SSL؟

7-4 اذكر وعرِّف باختصار المُعامِلات التي تُعرِّف حالة جلسة SSL.

7-5 اذكر وعرِّف باختصار المُعامِلات التي تُعرِّف حالة توصيلة SSL.

7-6 اذكر الخدمات التي يوفرها بروتوكول سجل SSL.

7-7 اذكر خطوات النقل في بروتوكول سجل SSL.

7-8 اذكر وعرّف باختصار الفئات الرئيسة للأطراف المشتركة في SET.

7-9 ما المقصود بالتوقيع المزدوج وما الغرض منه؟

7-7-2 مسائل

7-1 في SSL وTLS لماذا يوجد بروتوكول منفصل لتغيير مواصفات التشفير بدلاً من تضمين رسالة تغيير مواصفات التشفير (change_cipher_spec) في بروتوكول المصافحة؟

7-2 لكلٍّ من التهديدات الآتية لأمن الويب، صف كيفية المواجهة باستخدام خاصية معينة في SSL:

a. الهجوم الاستقصائي على التشفير: البحث الشامل لفضاء المفتاح لخوارزمية تشفير تقليدية.

b. هجوم القاموس بمعرفة نص أصلي غير مُشفَّر: تحتوي كثير من الرسائل على نصوص أصلية يمكن توقعها مثل أمر GET في بروتوكول HTTP. وينشئ المهاجم قاموساً يتضمن كل التشفير المحتملة برسالة النص الأصلي المعروف. وعندما يعترض المهاجم

رسالة مُشفَّرة، يأخذ الجزء الذي يحتوي النص الأصلي المعروف المُشفَّر ويبحث في القاموس عن النص المُشفَّر المناظر. ويجب أن يتوافق النص المُشفَّر مع أحد المُدخَلات بالقاموس تم تشفيره بنفس المفتاح السري. إذا حدث توافق مع عدة مُدخلات، فيمكن تجريب كلٍّ منها لتحديد النص الأصلي الكامل للنص المُشفَّر. وهذا النوع من الهجوم فعّال، خصوصاً عند استخدام مفاتيح ذات أطوال قصيرة (مثلاً بطول 40 بتاً).

c. هجوم إعادة التشغيل: وفيه يتم إعادة إرسال رسائل مصافحة SSL السابقة.
d. هجوم رجل في الوسط: يتدخل المهاجم أثناء تبادل المفاتيح، ويتصرف كالزبون مع الخادم وكالخادم مع الزبون.
e. التقاط كلمة السر: يتم التنصت على حركة مرور بيانات HTTP أو أي تطبيق آخر لمعرفة كلمات السر.
f. تزييف عنوان IP: وفيه يتم استخدام عناوين IP مزيّفة لخداع المضيف لقبول بيانات ليست حقيقة.
g. اختطاف عنوان IP: وفيه يقوم المهاجم بعرقلة اتصال موثق نشيط بين مضيّفين، وأخْذ مكان أحدهما للاتصال بالآخر.
h. فيضان SYN: يرسل المهاجم رسائل SYN في بروتوكول TCP لطلب إنشاء اتصال، ولكنه لا يرد على الرسالة النهائية لإتمام عملية تأسيس الاتصال. وعادةً ما تترك وحدة TCP المهاجِمة التوصيلة نصف مفتوحة لمدّة بضع دقائق. ويمكن أن يؤدي تكرار إرسال رسائل SYN إلى إعاقة TCP عن العمل.

3-7 بناءً على ما تعلمته في هذا الفصل، هل من الممكن للمُستلِم في SSL أن يعيد ترتيب كتل سجل SSL التي تصله بترتيب مختلف عما أُرسِلت به؟ إذا كان الأمر كذلك، فوضِّح كيف يحدث ذلك. وإلا فاذكر سبب عدم إمكانية ذلك.