Introduction SSL avec exemple de transaction et échange de paquets

Contenu

Introduction

Ce document décrit les concepts de base du protocole SSL (Secure Sockets Layer) et fournit un exemple de transaction et de capture de paquets.

Présentation des enregistrements SSL

L'unité de données de base dans SSL est un enregistrement. Chaque enregistrement se compose d'un en-tête d'enregistrement de cinq octets, suivi de données.

Format d'enregistrement

- **type :** uint8 - valeurs répertoriées
- **Version** : uint16
- **Longueur:** uint16

Type	Version		Longueur	
T	VH	VL	LH	LL

Type d'enregistrement

Il existe quatre types d'enregistrement dans SSL :

- **Échange de mains** (22, 0x16)
- **Modifier la spécification de chiffrement** (20, 0x14)
- **Alerte** (21, 0x15)
- **Données d'application** (23, 0x17)

Version d'enregistrement

La version de l'enregistrement est une valeur de 16 bits et est formatée dans l'ordre du réseau.

> **Note**: Pour SSL version 3 (SSLv3), la version est 0x0300. Pour TLSv1 (Transport Layer Security Version 1), la version est 0x0301. Le dispositif de sécurité adaptatif (ASA) de Cisco ne prend pas en charge SSL version 2 (SSLv2), qui utilise la version 0x0002, ni aucune version de TLS supérieure à TLSv1.

Longueur d'enregistrement

La longueur d'enregistrement est une valeur de 16 octets et est formatée dans l'ordre du réseau.

En théorie, cela signifie qu'un seul enregistrement peut avoir une longueur maximale de 65 535 (2^16 -1) octets. Le RFC2246 TLSv1 indique que la longueur maximale est de 16 383 (2^14 -1) octets. Les produits Microsoft (Microsoft Internet Explorer et Internet Information Services) sont connus pour dépasser ces limites.

Types d'enregistrements

Cette section décrit les quatre types d'enregistrements SSL.

Enregistrements de connexion

Les enregistrements de connexion contiennent un ensemble de messages utilisés pour la connexion. Voici les messages et leurs valeurs :

- **Demande Hello** (0, 0x00)
- **Hello client** (1, 0x01)
- **Serveur Hello** (2, 0x02)
- **Certificat** (11, 0x0B)
- **Échange de clés serveur** (12, 0x0C)
- **Demande de certificat** (13, 0x0D)
- **Serveur Hello terminé** (14, 0x0E)
- **Vérification du certificat** (15, 0x0F)
- **Échange de clés client** (16, 0x10)
- **Terminé** (20, 0x14)

Dans le cas simple, les enregistrements de connexion ne sont pas chiffrés. Cependant, un enregistrement d'échange qui contient un message terminé est toujours chiffré, car il se produit toujours après un enregistrement CCS (Change Cipher Spec).

Enregistrements CCS

Les enregistrements CCS sont utilisés pour indiquer un changement dans les chiffrement cryptographiques. Immédiatement après l'enregistrement CCS, toutes les données sont cryptées avec le nouveau chiffrement. Les enregistrements CCS peuvent être chiffrés ou non ; dans une connexion simple avec une seule connexion, l'enregistrement CCS n'est pas chiffré.

Enregistrements d'alertes

Les enregistrements d'alerte sont utilisés afin d'indiquer à l'homologue qu'une condition s'est produite. Certaines alertes sont des avertissements, tandis que d'autres sont fatales et provoquent l'échec de la connexion. Les alertes peuvent être chiffrées ou non et peuvent se produire lors d'une connexion ou d'un transfert de données. Il existe deux types d'alertes :

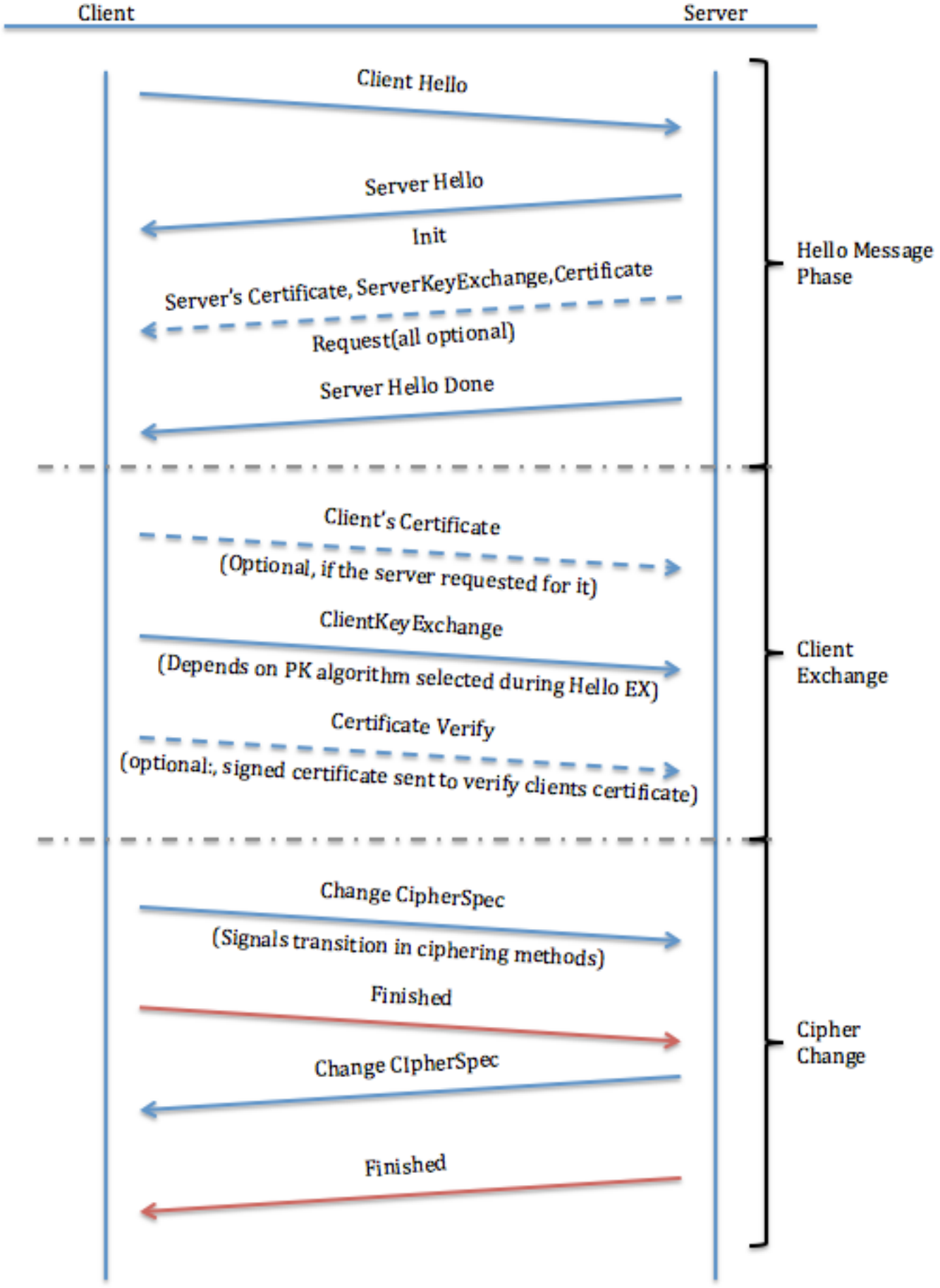
- **Alertes de fermeture :** La connexion entre le client et le serveur doit être correctement fermée afin d'éviter tout type d'attaque de troncature. Un message **close_notify** est envoyé qui indique au destinataire que l'expéditeur n'enverra plus de messages sur cette connexion.
- **Alertes d'erreur :** Lorsqu'une erreur est détectée, la partie qui détecte envoie un message à l'autre partie. Dès la transmission ou la réception d'un message d'alerte fatal, les deux parties ferment immédiatement la connexion. Voici quelques exemples d'alertes d'erreur :
  - **message_inattendu** (fatal)
  - **décompression_échec**
  - **échec_échange**

Enregistrement de données d'application

Ces enregistrements contiennent les données réelles de l'application. Ces messages sont transmis par la couche enregistrement et sont fragmentés, compressés et chiffrés, en fonction de l'état de connexion actuel.

Exemple de transaction

Cette section décrit un exemple de transaction entre le client et le serveur.



Hello Exchange

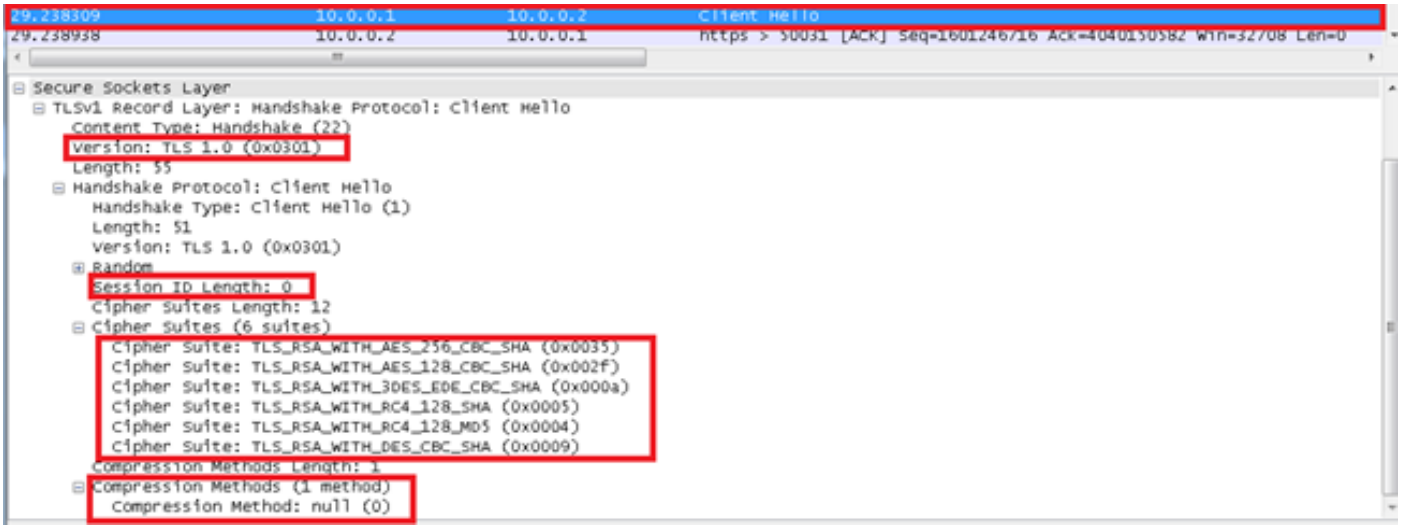
Lorsqu'un client et un serveur SSL commencent à communiquer, ils s'accordent sur une version de protocole, sélectionnent des algorithmes de chiffrement, s'authentifient mutuellement, et utilisent des techniques de chiffrement à clé publique afin de générer des secrets partagés. Ces processus sont exécutés dans le protocole de connexion. En résumé, le client envoie un message Hello du client au serveur, qui doit répondre avec un message Hello du serveur ou une erreur fatale se produit et la connexion échoue. Le client Hello et le serveur Hello sont utilisés pour établir des capacités d'amélioration de la sécurité entre le client et le serveur.

Allô du client

Le client Hello envoie ces attributs au serveur :

- **Version du protocole :** Version du protocole SSL par lequel le client souhaite communiquer au cours de cette session.
- **ID de session :** ID d'une session que le client souhaite utiliser pour cette connexion. Dans le premier Hello du client de l'échange, l'ID de session est vide (reportez-vous à l'écran de capture de paquets tiré après la note).
- **Suite Cipher :** Ceci est transmis du client au serveur dans le message Hello du client. Il contient les combinaisons d'algorithmes cryptographiques pris en charge par le client par ordre de préférence du client (premier choix en premier). Chaque suite de chiffrement définit à la fois un algorithme d'échange de clés et une spécification de chiffrement. Le serveur sélectionne une suite de chiffrement ou, si aucun choix acceptable n'est présenté, renvoie une alerte d'échec de la connexion et ferme la connexion.
- **Méthode de compression:** Inclut une liste d'algorithmes de compression pris en charge par le client. Si le serveur ne prend en charge aucune méthode envoyée par le client, la connexion échoue. La méthode de compression peut également être null.

  Note: L'adresse IP du serveur dans les captures est 10.0.0.2 et l'adresse IP du client est 10.0.0.1.



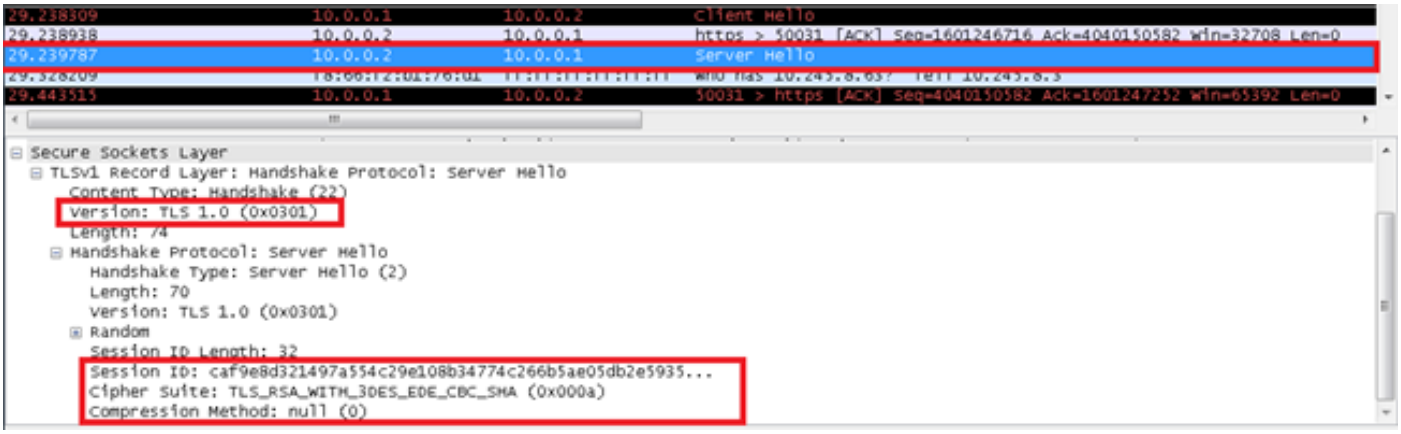
Allô du serveur

Le serveur renvoie ces attributs au client :

- **Version du protocole :** Version choisie du protocole SSL que le client prend en charge.
- **ID de session :** Identité de la session qui correspond à cette connexion. Si l'ID de session

envoyé par le client dans le message Hello du client n'est pas vide, le serveur recherche une correspondance dans le cache de session. Si une correspondance est trouvée et que le serveur est prêt à établir la nouvelle connexion à l'aide de l'état de session spécifié, le serveur répond avec la même valeur que celle fournie par le client. Cela indique une reprise de session et exige que les parties passent directement aux messages finis. Sinon, ce champ contient une valeur différente qui identifie la nouvelle session. Le serveur peut renvoyer un **session_id** vide afin d'indiquer que la session ne sera pas mise en cache et ne pourra donc pas être reprise.

- **Suite Cipher :** Sélectionné par le serveur dans la liste envoyée par le client.
- **Méthode de compression:** Sélectionné par le serveur dans la liste envoyée par le client.
- **requête de certificat:** Le serveur envoie au client une liste de tous les certificats qui y sont configurés et permet au client de sélectionner le certificat qu'il souhaite utiliser pour l'authentification.

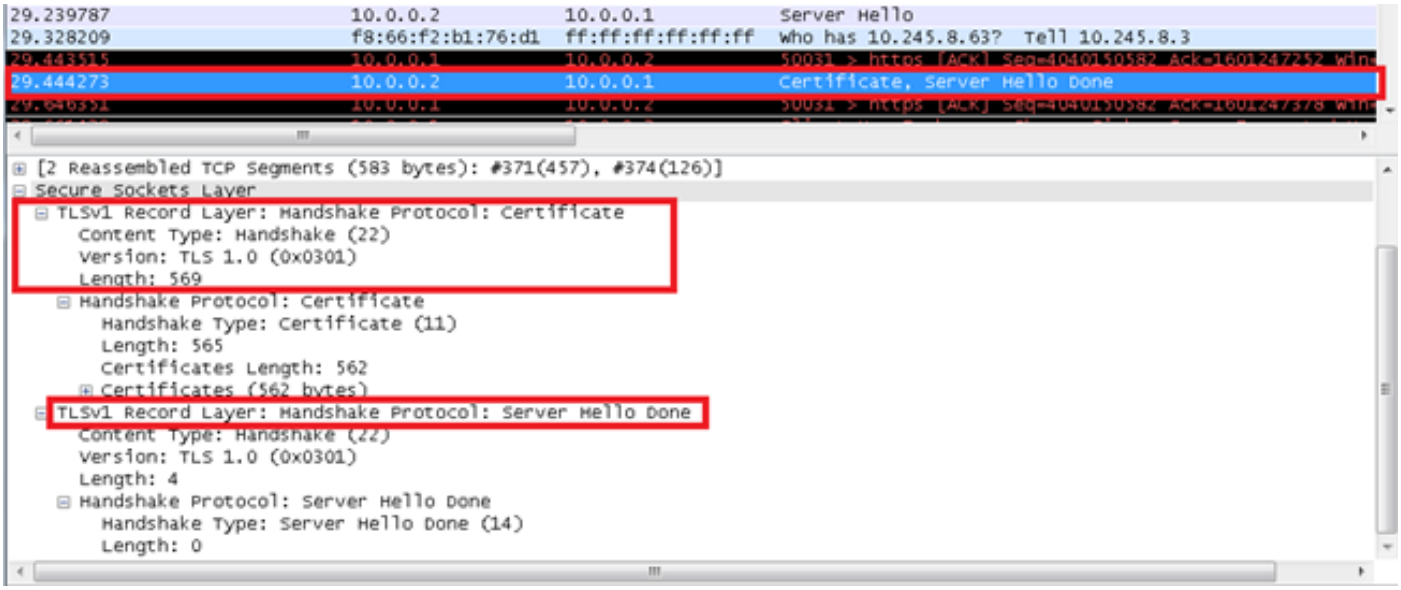


Pour les demandes de reprise de session SSL :

- Le serveur peut également envoyer une requête Hello au client. Il s'agit uniquement de rappeler au client qu'il doit commencer la renégociation avec une requête Hello du client lorsque cela est pratique. Le client ignore la requête Hello du serveur si le processus de connexion est déjà en cours.
- Les messages d'échange ont plus de priorité que la transmission des données d'application. La renégociation ne doit pas commencer plus d'une ou deux fois le temps de transmission d'un message de données d'application de longueur maximale.

Hello du serveur terminé

Le message Hello Done du serveur est envoyé par le serveur afin d'indiquer la fin du Hello du serveur et des messages associés. Après avoir envoyé ce message, le serveur attend une réponse du client. À la réception du message Hello Done du serveur, le client vérifie que le serveur a fourni un certificat valide, si nécessaire, et vérifie que les paramètres Hello du serveur sont acceptables.



Certificat serveur, échange de clés serveur et demande de certificat (facultatif)

- **Certificat serveur :** si le serveur doit être authentifié (ce qui est généralement le cas), le serveur envoie son certificat immédiatement après le message Hello du serveur. Le type de certificat doit être approprié pour l'algorithme d'échange de clé de la suite de chiffrement sélectionnée et il s'agit généralement d'un certificat X.509.v3.
- **Échange de clés serveur :** le message Échange de clés serveur est envoyé par le serveur s'il n'a pas de certificat. Si les paramètres Diffie-Hellman (DH) sont inclus dans le certificat du serveur, ce message n'est pas utilisé.
- **Demande de certificat :** un serveur peut éventuellement demander un certificat au client, le cas échéant pour la suite de chiffrement sélectionnée.

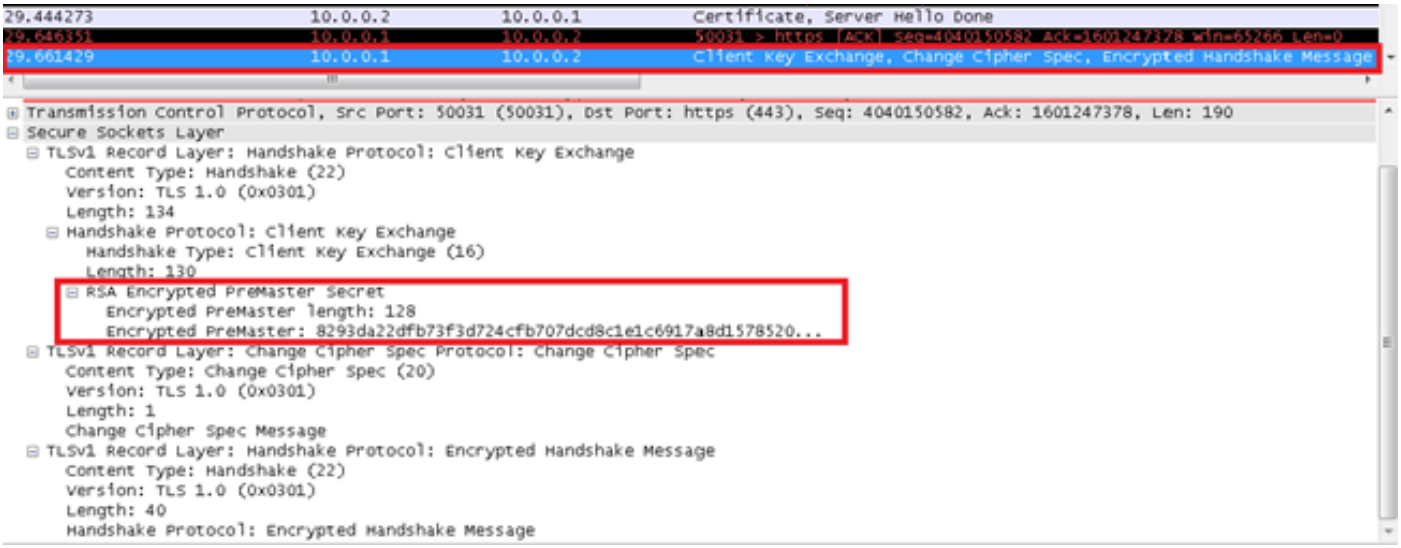
Exchange client

Certificat client (facultatif)

Il s'agit du premier message que le client envoie après avoir reçu un message Hello Done du serveur. Ce message n'est envoyé que si le serveur demande un certificat. Si aucun certificat approprié n'est disponible, le client envoie une alerte **no_certificate** à la place. Cette alerte n'est qu'un avertissement ; cependant, le serveur peut répondre par une alerte fatale de défaillance de la connexion si l'authentification du client est requise. Les certificats DH du client doivent correspondre aux paramètres DH spécifiés par le serveur.

Échange de clés client

Le contenu de ce message dépend de l'algorithme de clé publique sélectionné entre les messages Hello du client et Hello du serveur. Le client utilise soit une clé de maître chiffrée par l'algorithme RSA (Rivest-Shamir-Addleman), soit DH pour l'accord de clé et l'authentification. Lorsque RSA est utilisé pour l'authentification du serveur et l'échange de clés, un **pre_master_secret** de 48 octets est généré par le client, chiffré sous la clé publique du serveur et envoyé au serveur. Le serveur utilise la clé privée afin de déchiffrer le **pre_master_secret**. Les deux parties convertissent ensuite le **pre_master_secret** en **master_secret**.



Vérification du certificat (facultatif)

Si le client envoie un certificat avec la capacité de signature, un message de vérification de certificat signé numériquement est envoyé afin de vérifier explicitement le certificat.

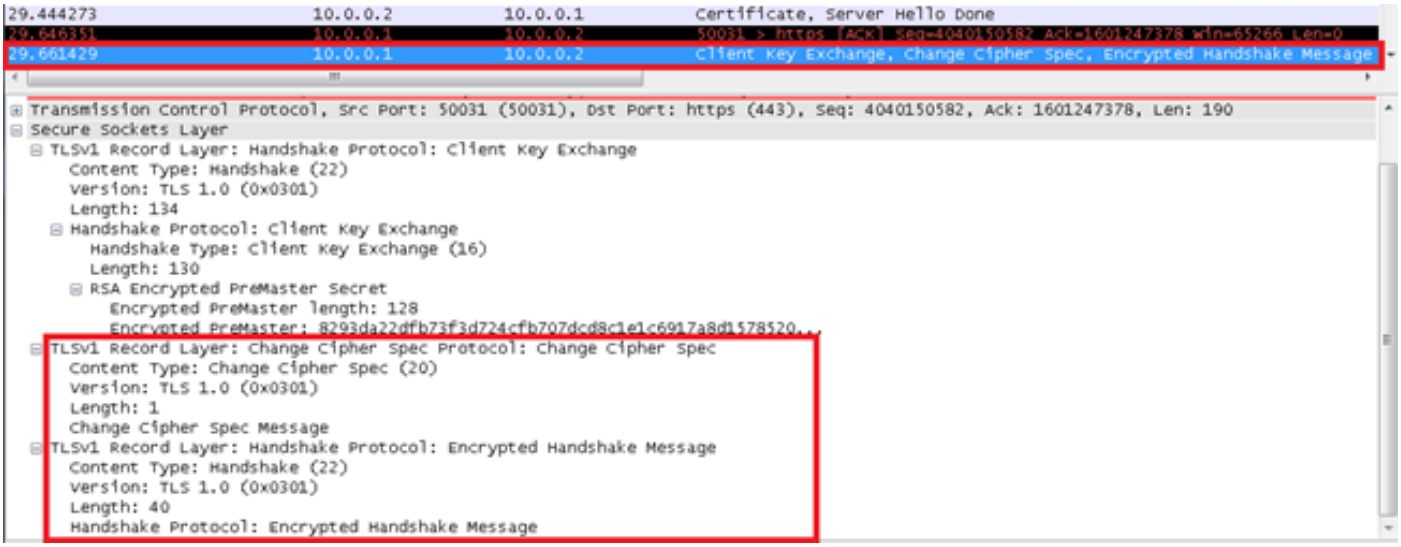
Changement de chiffre

Modifier les messages de spécification de chiffrement

Le message Change Cipher Spec est envoyé par le client et celui-ci copie la spécification Cipher en attente (la nouvelle) dans la spécification Cipher actuelle (celle précédemment utilisée). Le protocole Change Cipher Spec existe afin de signaler les transitions dans les stratégies de chiffrement. Le protocole se compose d'un seul message, qui est chiffré et compressé sous la spécification de chiffrement actuelle (et non pas en attente). Le message est envoyé à la fois par le client et le serveur afin d'avertir la partie réceptrice que les enregistrements ultérieurs sont protégés par les dernières spécifications et clés de chiffrement négociées. La réception de ce message entraîne le destinataire à copier l'état en attente de lecture dans l'état en cours de lecture. Le client envoie un message Change Cipher Spec après l'échange de clé de connexion et les messages Certificate Verify (le cas échéant), et le serveur en envoie un une après avoir traité avec succès le message d'échange de clé qu'il a reçu du client. Lorsqu'une session précédente reprend, le message Modifier la spécification du chiffrement est envoyé après les messages Hello. Dans les captures, les messages Client Exchange, Change Cipher et Terminé sont envoyés en tant que message unique du client.

Messages terminés

Un message Terminé est toujours envoyé immédiatement après un message Change Cipher Spec afin de vérifier que les processus d'échange de clés et d'authentification ont réussi. Le message Terminé est le premier paquet protégé avec les algorithmes, clés et secrets les plus récents. Aucun accusé de réception du message Terminé n'est requis ; les parties peuvent commencer à envoyer des données chiffrées immédiatement après avoir envoyé le message Terminé. Les destinataires des messages finis doivent vérifier que le contenu est correct.



Informations connexes

- [RFC 6101 - Protocole de couche de sockets sécurisés version 3.0]
- [Wireshark SSL wiki] - décrypter les paquets SSL avec Wireshark
- [Support et documentation techniques - Cisco Systems]