Groupe de travail Réseau
Request for Comments : 2817
RFC mise à jour : 2616
Catégorie : En cours de normalisation

R. Khare, 4K Associates / UC Irvine
S. Lawrence, Agranat Systems, Inc.
mai 2000
Traduction Claude Brière de L'Isle

Mise à niveau de TLS au sein de HTTP/1.1

Résumé
Le présent mémoire explique comment utiliser le mécanisme Upgrade dans HTTP/1.1 pour initier la sécurité de la couche Transport (TLS, *Transport Layer Security*) sur une connexion TCP existante. Cela permet au trafic HTTP non sécurisé et sécurisé de partager le même accès bien connu (dans le cas présent, http: à 80 plutôt que https: à 443). Il permet aussi un "hôte virtuel", de sorte qu'un seul serveur HTTP + TLS peut désintriquer du trafic destiné à plusieurs noms d'hôtes à une seule adresse IP.

Comme HTTP/1.1 [1] définit Upgrade comme un mécanisme bond par bond, le présent mémoire expose aussi la méthode HTTP CONNECT pour l'établissement de tunnels de bout en bout à travers des mandataires HTTP. Enfin, le présent mémoire établit de nouveaux registres IANA pour les codes d'état de HTTP public, ainsi que pour les jetons de produit Upgrade publics ou privés.

Le présent mémoire N'AFFECTE PAS la définition actuelle du schéma d'URI 'https', qui définit déjà un espace de noms séparé (http://example.org/ et https://example.org/ ne sont pas équivalents).

Table des Matières

1. Motivation

La pratique historique de développement de HTTP sur SSL3 [3] a distingué la combinaison de HTTP seul avec un schéma d'URI unique et avec le numéro d'accès TCP. Le schéma 'http' visé dans le protocole HTTP seul sur l'accès 80, alors que 'https' visait le protocole HTTP sur SSL à l'accès 443. Les numéros d'accès bien connus parallèles ont été demandés de façon similaire – et dans certains cas, accordés -- pour distinguer les utilisations sécurisées et non sécurisées des autres protocoles d'application (par exemple, snews, ftps). Cette approche diminue effectivement de moitié le nombre d'accès bien connus disponibles.

À la réunion de Washington DC de l'IETF en décembre 1997, les directeurs de domaines d'applications et l'IESG ont réaffirmé que la pratique de production de numéros d'accès "sécurisés" parallèles devraient être déconseillée. Le mécanisme Upgrade de HTTP/1.1 peut appliquer la sécurité de la couche Transport [6] à une connexion HTTP ouverte.

Dans les presque deux années qui ont suivi, il y a eu une large acceptation du concept qui sous-tend cette proposition, mais peu d'intérêt pour mettre en œuvre des solutions de remplacement à l'accès 443 pour la navigation générique sur la Toile. En fait, rien dans le présent mémoire n'affecte l'interprétation actuelle des URI https:. Cependant, les nouveaux protocoles d'application construits par dessus HTTP, tels que le protocole d'impression sur Internet [7], réclament justement un tel mécanisme afin de d'avancer dans le processus de normalisation de l'IETF.

Le mécanisme Upgrade résout aussi le problème de "l'hébergement virtuel". Plutôt que d'allouer plusieurs adresses IP à un seul hôte, un serveur HTTP/1.1 va utiliser l'en-tête Host: pour ôter l'ambiguïté sur le service de la Toile recherché. Comme l'utilisation de HTTP/1.1 est devenue prédominante, de plus en plus de FAI *(fournisseur d'accès Internet)* offrent de l'hébergement virtuel fondé sur le nom, retardant ainsi l'épuisement de l'espace d'adresse IP.

TLS (et SSL) ont été entravés par la même limitation que les versions antérieures de HTTP : la prise de contact initiale ne spécifie pas le nom d'hôte de destination, s'appuyant exclusivement sur l'adresse IP. L'utilisation d'un préambule Upgrade: HTTP/1.1 en clair à la prise de contact TLS – en choisissant les certificats fondés sur l'en-tête Host: initial – va permettre aux FAI de fournir aussi de l'hébergement virtuel sécurisé fondé sur le nom.

2. Introduction

TLS, autrement dit, SSL (Secure Sockets Layer, *couche de connexion sécurisée*), établit une connexion privée de bout en bout, incluant facultativement une forte authentification mutuelle, en utilisant divers systèmes de chiffrement. Au départ, une phase de prise de contact utilise trois sous protocoles pour établir une couche d'enregistrement, des points d'extrémité d'authentification, des paramètres de réglage, ainsi que des rapports d'erreur. Puis, il y a un protocole d'enregistrement sur la couche en cours qui traite le chiffrement, la compression, et le réassemblage pour le reste de la connexion. Cette dernière est destinée à être complètement transparente. Par exemple, il n'y a pas de dépendance entre les marqueurs d'enregistrements et/ou certificats de TLS et le codage ou l'authentification par tronçons de HTTP/1.1.

Le client ou le serveur peut utiliser le mécanisme Upgrade de HTTP/1.1 [1 (paragraphe 14.42)] pour indiquer qu'une connexion sécurisée par TLS est souhaitée ou nécessaire. Le présent mémoire définit le jeton Upgrade "TLS/1.0", et un nouveau code d'état HTTP, "426 Upgrade exigé".

La Section 3 et la Section 4 décrivent le fonctionnement d'un client et d'un serveur directement connectés. Les mandataires intermédiaires doivent établir un tunnel de bout en bout avant d'appliquer ces opérations, comme expliqué à la Section 5.

2.1 Terminologie des exigences

Dans le présent document, les mots clé "DOIT", "NE DOIT PAS", "EXIGE", "DEVRAIT" NE DEVRAIT PAS" et "PEUT" sont à interpréter comme décrit dans la RFC 2119 [11].

3. Mise à niveau de HTTP demandée par le client sur TLS

Lorsque le client envoie une demande HTTP/1.1 avec un champ d'en-tête Upgrade qui contient le jeton "TLS/1.0", il demande au serveur d'achever la demande HTTP/1.1 en cours après être passé à TLS/1.0.

3.1 Mise à niveau facultative

Un client PEUT offrir de passer au fonctionnement sécurisé durant toute demande HTTP claire lorsque une réponse non

sécurisée serait acceptable :

GET http://example.bank.com/acct_stat.html?749394889300 HTTP/1.1
Host: example.bank.com
Upgrade: TLS/1.0
Connection: Upgrade

Dans ce cas, le serveur PEUT répondre normalement à l'opération HTTP claire, OU passer au fonctionnement sécurisé (comme précisé au paragraphe suivant).

Noter que HTTP/1.1 [1] spécifie que "le mot clé upgrade DOIT être fourni au sein d'un champ d'en-tête Connection (paragraphe 14.10) chaque fois que Upgrade est présent dans un message HTTP/1.1".

3.2 Mise à niveau obligatoire

Si une réponse non sécurisée serait inacceptable, un client DOIT envoyer d'abord une demande OPTIONS pour achever le passage à TLS/1.0 (si possible).

OPTIONS * HTTP/1.1
Host: example.bank.com
Upgrade: TLS/1.0
Connection: Upgrade

3.3 Acceptation par le serveur de la demande de mise à niveau

Comme spécifié dans HTTP/1.1 [1], si le serveur est prêt à initier la prise de contact TLS, il DOIT envoyer le "101 Changement de protocole" intermédiaire et DOIT inclure un en-tête de réponse Upgrade qui spécifie les jetons de la pile de protocole à laquelle il passe :

HTTP/1.1 101 Changement de protocoles
Upgrade: TLS/1.0, HTTP/1.1
Connection: Upgrade

Noter que les jetons de protocole dont la liste figure dans l'en-tête Upgrade d'une réponse 101 Changement de protocoles spécifient une pile ordonnée de "bas en haut".

Comme spécifié dans HTTP/1.1 [1], paragraphe 10.1.2: "Le serveur va changer de protocoles pour ceux définis par le champ d'en-tête Upgrade de la réponse immédiatement après la ligne vide qui termine la réponse 101".

Une fois achevée avec succès la prise de contact TLS, le serveur DOIT continuer par la réponse à la demande d'origine. Tout échec de prise de contact TLS DOIT conduire à la déconnexion, selon la spécification d'alerte d'erreur TLS.

4. Mise à niveau de HTTP demandée par le serveur sur TLS

Le champ d'en-tête de réponse Upgrade annonce une possible mise à niveau de protocole qu'un serveur PEUT accepter. En conjonction avec le code d'état "426 Mise à niveau exigée", un serveur peut annoncer la ou les mises à niveau de protocole exactes qu'un client DOIT accepter pour achever la demande.

4.1 Annonce facultative

Comme spécifié dans HTTP/1.1 [1], le serveur PEUT inclure un en-tête Upgrade dans toute réponse autre que 101 ou 426 pour indiquer sa volonté de passer à toute combinaison des protocoles figurant sur la liste.

4.2 Annonce obligatoire

Un serveur PEUT indiquer qu'une demande d'un client ne peut pas être achevée sans que TLS utilise le code d'état "426 Mise à niveau exigée", qui DOIT inclure un champ d'en-tête Upgrade spécifiant le jeton de la version TLS exigée.

HTTP/1.1 426 Upgrade Required
Upgrade: TLS/1.0, HTTP/1.1

Connection: Upgrade

Le serveur DEVRAIT inclure un corps de message dans la réponse 426 qui indique en forme lisible par l'homme la cause de l'erreur et décrive des façons de procéder alternatives qui seraient disponibles pour l'utilisateur.

Noter que même si un client veut utiliser TLS, il doit utiliser les opérations de la Section 3 pour le faire ; la prise de contact TLS ne peut pas commencer immédiatement après la réponse 426.

5. Mise à niveau à travers des mandataires

Comme en-tête bond par bond, Upgrade est négocié entre chaque paire de partenaires HTTP. Si un agent d'utilisateur envoie une demande avec un en-tête Upgrade à un mandataire, c'est une demande de changement du protocole entre lui-même et le mandataire et non un changement de bout en bout.

Comme TLS exige, en particulier, que la connexité de bout en bout fournisse l'authentification et empêche les attaques par interposition, le présent mémoire spécifie la méthode CONNECT pour établir un tunnel à travers les mandataires.

Une fois qu'un tunnel est établi, toutes les opérations de la Section 3 peuvent être utilisées pour établir une connexion TLS.

5.1 Implications de la mise à niveau bond par bond

Si un serveur d'origine reçoit un en-tête Upgrade d'un mandataire et répond avec un 101 Changement de protocoles, il ne change le protocole que sur la connexion entre le mandataire et lui-même. De même, un mandataire peut retourner une réponse 101 à son client pour changer le protocole sur cette connexion indépendamment des protocoles qu'il utilise pour communiquer vers le serveur d'origine.

Ces scénarios compliquent aussi le diagnostic d'une réponse 426. Comme Upgrade est un en-tête bond par bond, un mandataire qui ne reconnaît pas 426 pourrait retirer l'en-tête Upgrade adjoint et empêcher le client de déterminer le changement de protocole requis. Si un client reçoit un état 426 sans qu'il soit accompagné d'un en-tête Upgrade, il va avoir besoin de demander une connexion tunnel de bout en bout comme décrit au paragraphe 5.2 et de répéter la demande afin d'obtenir les informations de mise à niveau requises.

Cette définition bond par bond de Upgrade était un choix délibéré. Elle permet un déploiement par incrément de part et d'autre du mandataire, et une optimisation des protocoles entre des mandataires en cascade sans le porter à la connaissance des parties qui ne sont pas impliquées dans le changement.

5.2 Demande d'un tunnel avec CONNECT

Une méthode CONNECT demande qu'un mandataire établisse une connexion tunnel en son nom. La portion URI de demande de la Ligne de demande est toujours une "autorité" comme défini par la syntaxe générique d'URI [2], c'est à dire le nom d'hôte et le numéro d'accès de destination de la connexion demandée, séparés par deux points :

```
CONNECT server.example.com:80 HTTP/1.1
Host: server.example.com:80
```

Les autres mécanismes de HTTP peuvent être utilisés normalement avec la méthode CONNECT -- excepté les demandes Upgrade de protocole de bout en bout, bien sûr, car le tunnel doit être établi d'abord.

Par exemple, l'authentification de mandataire pourrait être utilisée pour établir l'autorité de création d'un tunnel :

```
CONNECT server.example.com:80 HTTP/1.1
Host: server.example.com:80
Proxy-Authorization: basic aGVsbG86d29ybGQ=
```

Comme toute autre demande HTTP/1.1 traitée en parallèle, les données à tunneler peuvent être envoyées immédiatement après la ligne blanche. Les avertissements usuels s'appliquent aussi : les données peuvent être éliminées si la réponse finale est négative, et la connexion peut être réinitialisée sans réponse si plus d'un segment TCP est en instance.

5.3 Établissement d'un tunnel avec CONNECT

Toute réponse de succès (2xx) à une demande CONNECT indique que le mandataire a établi une connexion avec l'hôte et l'accès requis, et est passé au tunnelage de la connexion en cours vers cette connexion de serveur.

Il peut être le cas que le mandataire lui-même ne puisse atteindre le serveur d'origine requis qu'à travers un autre mandataire. Dans ce cas, le premier mandataire DEVRAIT faire une demande CONNECT au prochain mandataire, en demandant un tunnel à l'autorité. Un mandataire NE DOIT PAS répondre par un code d'état 2xx si il n'a pas une connexion directe ou en tunnel établie avec l'autorité.

Un serveur d'origine qui reçoit une demande CONNECT pour lui-même PEUT répondre par un code d'état 2xx pour indiquer qu'une connexion est établie.

Si à un point quelconque, l'un des homologues se trouve déconnecté, toutes les données en instance qui proviennent de cet homologue seront passées à l'autre, et après cela, l'autre connexion sera aussi terminée par le mandataire. Si il reste des données non livrées pour cet homologue, elles seront éliminées.

6. Raisons de l'utilisation d'un code d'état 4xx (erreur client)

La négociation fiable, interopérable des dispositifs Upgrade exige un signal d'échec sans ambiguïté. Le code d'état exigé 426 Upgrade permet à un serveur de déclarer de façon définitive les extensions de protocole précises qui doivent être appliquées à une ressource donnée.

Il pourrait apparaître au premier abord que la réponse aurait dû être une certaine forme de redirection (un code 3xx), par analogie avec une redirection de l'ancien style https: URI. Les agents d'utilisateur qui ne comprennent pas Upgrade: empêchent cela.

Supposons qu'un code 3xx ait été alloué pour "Mise à niveau exigée" ; un agent d'utilisateur qui ne la reconnaît pas va la traiter comme 300. Il va alors chercher un en-tête "Location" dans la réponse et tenter de répéter la demande à l'URL qui se trouve dans ce champ d'en-tête. Comme il ne sait pas mettre à niveau pour incorporer la couche TLS, il va au mieux échouer encore au nouvel URL.

7. Considérations pour l'IANA

L'IANA devra créer des registres pour deux espaces de noms, comme décrit dans le BCP 26 [10]:
- o Codes d'état HTTP
- o Jetons Upgrade HTTP.

7.1 Registre des codes d'état HTTP

Le registre des codes d'état HTTP définit l'espace de noms pour le jeton Code-d'état dans la ligne Status d'une réponse HTTP. Les valeurs initiales pour cet espace de noms sont celles spécifiées par :
1. le projet de norme pour HTTP/1.1 [1]
2. les auteurs et versions répartis sur la Toile [4] [définitions 420-424]
3. les collections WebDAV évoluées [5] (Travail en cours) [définition 425]
4. la Section 6 [définition 426]

Les valeurs à ajouter à cet espace de nom sont soumises à révision par l'IETF ([12], paragraphe 4.1). *(Corrigé selon l'errata n° 1801 du 18-07-2009)*

7.2 Registre des jetons HTTP Upgrade

Le registre de jetons HTTP Upgrade définit l'espace de noms pour les jetons de produits utilisés pour identifier les protocoles dans le champ d'en-tête Upgrade HTTP. Chaque jeton enregistré devrait être associé à une ou un ensemble de spécifications, et des informations de contact.

Le projet de norme pour HTTP/1.1 [1] spécifie que ces jetons obéissent à la production de 'product' :

product = jeton ["/" product-version]
product-version = jeton

Les enregistrements devraient être reçus sur la base du premier arrivé premier servi comme décrit dans le BCP 26 [10]. Ces spécifications n'ont pas besoin d'être des documents de l'IETF ni d'être soumis à révision de l'IESG, mais devraient obéir aux règles suivantes :

1. Un jeton, une fois enregistré, reste enregistré pour toujours.
2. L'enregistrement DOIT nommer un responsable de l'enregistrement.
3. L'enregistrement DOIT nommer un point de contact.
4. L'enregistrement PEUT nommer la documentation requise pour le jeton.
5. Le responsable PEUT changer l'enregistrement à tout moment. L'IANA tiendra un journal de ces changements, et le mettra à disposition sur demande.
6. Le responsable du premier enregistrement d'un jeton de "produit" DOIT approuver les enregistrements ultérieurs d'un jeton de "version" conjointement avec le jeton "produit" avant qu'ils puissent être enregistrés.
7. Si c'est absolument exigé, l'IESG PEUT réallouer la responsabilité d'un jeton. Cela ne sera normalement utilisé que dans le cas où le responsable ne peut pas être contacté.

La présente spécification définit le jeton de protocole "TLS/1.0" comme l'identifiant du protocole spécifié par le protocole TLS [6].

Il N'EST PAS exigé que les spécifications pour les jetons upgrade soient disponibles au public, mais les informations de contact pour leur enregistrement DEVRAIENT l'être.

8. Considérations pour la sécurité

Le potentiel d'attaque par interposition (en supprimant l'en-tête Upgrade) reste le même que dans la pratique mixte actuelle de http/https :

- o Retirer l'en-tête Upgrade est similaire à réécrire des pages de la toile pour changer les liens https:// en liens http://.
- o Le risque n'est présent que si le serveur veut vendre de telles informations en premier lieu à la fois sur un canal sécurisé et un canal non sécurisé.
- o Si le client sait de source sûre qu'un serveur est conforme à TLS, il peut ne pas insister en envoyant une demande Upgrade avec une méthode no-op comme OPTIONS.
- o Enfin, la spécification https: prévient que "les utilisateurs devraient examiner avec soin le certificat présenté par le serveur pour déterminer si il satisfait à leurs attentes".

De plus, pour les clients qui n'essayent pas explicitement d'invoquer TLS, les serveurs peuvent utiliser l'en-tête Upgrade dans toute réponse autre que 101 ou 426 pour annoncer la conformité à TLS. Comme la conformité à TLS devrait être considérée comme un dispositif de serveur et non de la ressource traitée, il devrait être suffisant de l'envoyer tout de suite, et de laisser les clients mettre ce fait en antémémoire.

8.1 Implications pour le schéma d'URI https:

Bien que rien dans le présent mémoire n'affecte la définition du schéma d'URI 'https', l'adoption largement répandue de ce mécanisme pour les contenus HyperText pourrait s'appuyer sur 'http' pour identifier les ressources aussi bien sécurisées que non sécurisées.

Le choix des caractéristiques de sécurité exigées sur la connexion est laissé au client et au serveur. Cela permet à l'une et l'autre partie d'utiliser toutes les informations disponibles pour cette détermination. Par exemple, les agents d'utilisateur peuvent s'appuyer sur les réglages préférés de l'utilisateur ou sur les informations sur la sécurité du réseau telles que "TLS exigé sur toutes les opérations POST qui ne sont pas sur mon réseau local", ou les serveurs peuvent appliquer des règles d'accès aux ressources telles que " le FORMULAIRE de cette page doit être rempli et soumis en utilisant TLS".

8.2 Considérations de sécurité pour CONNECT

Les risques de sécurité sont lourds pour un tunnel TCP générique. D'abord, une telle autorisation devrait être limitée à un petit nombre d'accès connus. Le mécanisme Upgrade: défini ici n'exige que le tunnelage avant à l'accès 80. Ensuite, comme les données tunnelées sont opaques pour le mandataire, il y a des risques supplémentaires de tunneler à d'autres accès bien connus ou réservés. Un client HTTP supposé qui se CONNECTE à l'accès 25 pourrait, par exemple, servir de relais à des pourriels via SMTP.

Références

[1] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach et T. Berners-Lee, "Protocole de transfert hypertexte -- HTTP/1.1", RFC 2616, juin 1999. *(D.S., MàJ par 2817)*

[2] T. Berners-Lee, R. Fielding et L. Masinter, "Identifiants de ressource uniformes (URI) : Syntaxe générique", RFC 2396, août 1998.

[3] E. Rescorla, "HTTP sur TLS", RFC 2818, mai 2000. *(Information)*.

4] Y. Goland, E. Whitehead, A. Faizi, S. Carter et D. Jensen, "Extensions HTTP pour la création répartie -- WEBDAV", RFC 2518, février 1999. *(Obsolète, voir la RFC 4918)*

[5] J. Slein, E.J. Whitehead et autres, "WebDAV Advanced Collections Protocol", Travail en cours.

[6] T. Dierks et C. Allen, "Protocole TLS version 1.0", RFC 2246, janvier 1999.

[7] Herriot, R., Butler, S., Moore, P. and R. Turner, "Internet Printing Protocol/1.0: Encoding and Transport", RFC 2565, April 1999.

[8] A. Luotonen, "Tunneling TCP based protocols through Web proxy servers", Travail en cours. (Aussi disponible dans : Luotonen, Ari. Web Proxy Servers, Prentice-Hall, 1997 ISBN:0136806120.)

[9] M. Rose,"Écrire des I-D et des RFC en utilisant XML", RFC 2629, juin 1999. *(Information)*

[10] T. Narten et H. Alvestrand, "Lignes directrices pour la rédaction d'une section Considérations relatives à l'IANA dans les RFC", RFC 2434, BCP 26, octobre, 1998. *(Rendue obsolète par la RFC 5226)*.

[11] S. Bradner, "Mots clés à utiliser dans les RFC pour indiquer les niveaux d'exigence", RFC 2119, BCP 14, mars 1997.

[12] T. Narten et H. Alvestrand, "Lignes directrices pour la rédaction d'une section Considérations relatives à l'IANA dans les RFC", RFC 5226, BCP 26, mai 2008.

Adresse des auteurs

Rohit Khare
4K Associates / UC Irvine
3207 Palo Verde
Irvine, CA 92612
US
téléphone : +1 626 806 7574
mél : rohit@4K-associates.com
URI : http://www.4K-associates.com/

Scott Lawrence
Agranat Systems, Inc.
5 Clocktower Place
Suite 400
Maynard, MA 01754
US
téléphone : +1 978 461 0888
mél : lawrence@agranat.com
URI : http://www.agranat.com/

Appendice A Remerciements

La méthode CONNECT a été décrite à l'origine dans un travail en cours intitulé "Traitement en parallèle de protocoles fondés sur TCP à travers des serveurs mandataires de la Toile *(Tunneling TCP based protocols through Web proxy servers)*", [8] par Ari Luotonen de Netscape Communications Corporation. Elle a été largement mise en œuvre par les mandataires HTTP, mais n'est jamais devenue un document en cours de normalisation de l'IETF. Le nom de la méthode CONNECT était réservé, mais pas défini dans [1].

Les définitions fournies ici sont directement dérivées de ce premier mémoire, avec quelques modifications rédactionnelles et la conformité aux conventions de style établies depuis dans les autres spécifications HTTP.

Des remerciements supplémentaires sont dus à :
- Paul Hoffman pour son travail sur l'extension de la commande STARTTLS pour ESMTP.
- Roy Fielding pour son assistance sur les raisons de Upgrade: et son interaction avec OPTIONS.
- Eric Rescorla pour son travail sur la normalisation des pratiques comparées de l'https: existant.

- Marshall Rose, pour la description de type de document et les outils xml2rfc [9].
- Jim Whitehead, pour le tri de la gamme actuelle des codes d'état HTTP disponibles.
- Henrik Frystyk Nielsen, dont le travail sur le mécanisme d'extension obligatoire soulignait qu'un Upgrade bond par bond exige toujours le traitement en parallèle.
- Harald Alvestrand pour ses améliorations aux règles d'enregistrement des jetons.

Déclaration de copyright

Remerciement
Le financement de la fonction d'édition des RFC est actuellement fourni par l'Internet Society.