Internet Engineering Task Force (IETF)
**Request for Comments : 6520**
Catégorie : Sur la voie de la normalisation
ISSN: 2070-1721

R. Seggelmann, Muenster Univ. of Appl. Sciences
M. Tüexen, Muenster Univ. of Appl. Sciences
M. Williams, GWhiz Arts & Sciences
février 2012
Traduction Claude Brière de L'Isle

# Extension Heartbeat à la sécurité de la couche transport (TLS) et à la sécurité de la couche transport de datagrammes (DTLS)

**Résumé**
Le présent document décrit l'extension Heartbeat pour les protocoles de sécurité de la couche transport (TLS, *Transport Layer Security*) et sécurité de la couche de transport de datagrammes (DTLS, *Datagram Transport Layer Security*).

L'extension Heartbeat fournit un nouveau protocole pour TLS/DTLS permettant l'usage d'une fonction de maintien en vie sans effectuer de renégociation et une base pour la découverte de l'unité de transmission maximum de chemin (PMTU, *Path MTU*) pour DTLS.

**Statut de ce mémoire**
Ceci est un document de l'Internet sur la voie de la normalisation.

Le présent document a été produit par l'équipe d'ingénierie de l'Internet (IETF). Il représente le consensus de la communauté de l'IETF. Il a subi une révision publique et sa publication a été approuvée par le groupe de pilotage de l'ingénierie de l'Internet (IESG). Tous les documents approuvés par l'IESG ne sont pas candidats à devenir une norme de l'Internet ; voir la Section 2 de la RFC5741.

Les informations sur le statut actuel du présent document, tout errata, et comment fournir des réactions sur lui peuvent être obtenues à http://www.rfc-editor.org/info/rfc6520



## Table des matières

# 1. Introduction

## 1.1 Vue d'ensemble

Le présent document décrit l'extension Heartbeat pour les protocoles de sécurité de la couche transport (TLS, *Transport Layer Security*) et sécurité de la couche de transport de datagrammes (DTLS, *Datagram Transport Layer Security*) comme définis dans les [RFC5246] et [RFC6347] et leurs adaptations aux protocoles de transport spécifiques décrits dans les [RFC3436], [RFC5238], et [RFC6083].

DTLS est conçu pour sécuriser le trafic qui s'écoule sur des protocoles de transport non fiables. Généralement, de tels protocoles n'ont pas de gestion de session. Le seul mécanisme disponible à la couche DTLS pour dire si un homologue est toujours en vie est une renégociation coûteuse, en particulier quand l'application utilise du trafic unidirectionnel. De plus, DTLS a besoin d'effectuer la découverte de l'unité maximale de transmission du chemin (PMTU, *Path MTU*) mais n'a pas de type de message spécifique pour le réaliser sans affecter le transfert des messages de l'utilisateur.

TLS se fonde sur des protocoles fiables, mais il n'y a pas nécessairement de dispositif disponible pour garder la connexion en vie sans transfert de données continu.

L'extension Heartbeat décrite dans le présent document surmonte ces limitations. L'utilisateur peut utiliser le nouveau message HeartbeatRequest, auquel l'homologue doit répondre imédiatement par une HeartbeartResponse. Pour effectuer la découverte de PMTU, le message HeartbeatRequest contenant du bourrage peut être utilisé comme paquet sonde, comme décrit dans la [RFC4821].

## 1.2. Conventions

Les mots clés "DOIT", "NE DOIT PAS", "EXIGE", "DEVRA", "NE DEVRA PAS", "DEVRAIT", "NE DEVRAIT PAS", "RECOMMANDE", "PEUT", et "FACULTATIF" en majuscules dans ce document sont à interpréter comme décrit dans le BCP 14, [RFC2119].

# 2. Extension Heartbeat Hello

La prise en charge de Heartbeat est indiquée par des extensions Hello. Un homologue peut ne pas seulement indiquer que sa mise en œuvre prend en charge les Heartbeat, il peut aussi choisir si il veut recevoir des messages HeartbeatRequest et répondre avec des messages HeartbeatResponse ou si il veut seulement envoyer des messages HeartbeatRequest. Le premier est indiqué en utilisant "peer_allowed_to_send" *(l'homologue a la permission d'envoyer)* comme mode Heartbeat ; le dernier est indiqué en utilisant "peer_not_allowed_to_send" *(l'homologue n'a pas la permission d'envoyer)* comme mode Heartbeat. Cette décision peut être changée à chaque renégociation. Les messages HeartbeatRequest NE DOIVENT PAS être envoyés à un homologue qui indique "peer_not_allowed_to_send". Si un point d'extrémité qui a indiqué "peer_not_allowed_to_send" reçoit un message HeartbeatRequest, le point d'extrémité DEVRAIT éliminer le message en silence et PEUT envoyer un message d'alerte "unexpected_message" *(message non attendu)*.

Le format de l'extension de Hello Heartbeat est défini par :

```
enum {
  peer_allowed_to_send(1),
  peer_not_allowed_to_send(2),
  (255)
} HeartbeatMode;

struct {
  HeartbeatMode mode;
} HeartbeatExtension;
```

À réception d'un mode inconnu, un message d'erreur Alerte utilisant "illegal_parameter" *(paramètre illégal)* comme description d'alerte DOIT être envoyé en réponse.

## 3. Protocole Heartbeat

Le protocole Heartbeat est un nouveau protocole qui fonctionne par dessus la couche d'enregistrement. Le protocole lui-même consiste en deux types de messages : HeartbeatRequest et HeartbeatResponse.

```
enum {
  heartbeat_request(1),
  heartbeat_response(2),
  (255)
} HeartbeatMessageType;
```

Un message HeartbeatRequest peut arriver presque à tout moment pendant la durée de vie d'une connexion. Chaque fois qu'un message HeartbeatRequest est reçu, il DEVRAIT avoir pour réponse un message HeartbeatResponse correspondant.

Cependant, un message HeartbeatRequest NE DEVRAIT PAS être envoyé durant les prises de contact. Si une prise de contact est initiée pendant qu'une HeartbeatRequest est encore en transit, l'homologue envoyeur DOIT arrêter le temporisateur de retransmission DTLS pour elle. L'homologue receveur DEVRAIT éliminer le message en silence, si il arrive durant la prise de contact. Dans le cas de DTLS, les messages HeartbeatRequest provenant d'époques antérieures DEVRAIENT être éliminés.

Il NE DOIT PAS y avoir plus d'un message HeartbeatRequest en cours à chaque instant. Un message HeartbeatRequest est considéré être en cours jusqu'à la réception du message HeartbeatResponse correspondant, ou jusqu'à l'expiration du temporisateur de retransmission.

Quand on utilise un protocole de transport non fiable comme le protocole de contrôle d'encombrement de datagrammes (DCCP, *Datagram Congestion Control Protocol*) ou UDP, les messages HeartbeatRequest DOIVENT être retransmis en utilisant le simple schéma de temporisation et de retransmission que DTLS utilise pour les envois comme décrit au paragraphe 4.2.4 de la [RFC6347]. En particulier, après un certain nombre de retransmissions sans recevoir de message HeartbeatResponse correspondant avec la charge utile attendue, la connexion DTLS DEVRAIT être terminée. Le seuil utilisé pour cela DEVRAIT être le même que pour les messages de prise de contact DTLS. Noter qu'après l'expiration du temporisateur qui supervise les messages HeartbeatRequest, ce message n'est plus considéré comme en cours. Donc, le message HeartbeatRequest est éligible à la retransmission. Le schéma de retransmission, en combinaison avec la restriction qu'une seule HeartbeatRequest en cours est permise, assure que le contrôle d'encombrement est traité de façon appropriée en cas de non fourniture par le protocole de transport, comme dans le cas de DTLS sur UDP.

Quans on utilise un protocole de transport fiable comme le protocole de transmission de commandes de flux (SCTP, *Stream Control Transmission Protocol*) ou TCP, les messages HeartbeatRequest ont seulement besoin d'être envoyés une fois. La couche de transport va traiter les retransmissions. Si aucun message HeartbeatResponse correspondant n'a été reçu après un certain laps de temps, la connexion DTLS/TLS PEUT être terminée par l'application qui a initié l'envoi du message HeartbeatRequest.

## 4. Messages de demande et réponse Heartbeat

Les messages du protocole Heartbeat consistent en leur type, une charge utile arbitraire et un bourrage.

```
struct {
  type HeartbeatMessageType ;
  uint16 payload_length;
  charge utile opaque [HeartbeatMessage.payload_length];
  bourrage opaque [padding_length];
} HeartbeatMessage;
```

La longueur totale d'un message Heartbeat NE DOIT PAS excéder 2^14 ou max_fragment_length quand elle est négociée comme défini dans la [RFC6066].

type : le type de message, heartbeat_request ou heartbeat_response.

payload_length : longueur de la charge utile.

charge utile : le contenu de la charge utile est arbitraire.

Bourrage : le bourrage est un contenu aléatoire qui DOIT être ignoré par le receveur. La longueur d'un message Heartbeat est TLSPlaintext.length pour TLS et DTLSPlaintext.length pour DTLS. De plus, la longueur du champ type est d'un octet, et la longueur de payload_length est 2. Donc, la longueur du bourrage est TLSPlaintext.length - payload_length - 3 pour TLS et DTLSPlaintext.length - payload_length - 3 pour DTLS. La longueur de bourrage DOIT être au moins 16.

L'envoyeur d'un message Heartbeat DOIT utilisr un bourrage aléatoire d'au moins 16 octets. Le bourrage d'un message Heartbeat reçu DOIT être ignoré.

Si la longueur de charge utile d'un message Heartbeat reçu est trop grande, le message Heartbeat reçu DOIT être éliminé en silence.

Quand un message HeartbeatRequest est reçu et que l'envoi d'une HeartbeatResponse n'est ps interdit comme décrit ailleurs dans ce document, le receveur DOIT envoyer un message HeartbeatResponse correspondant portant une copie exacte de la charge utile de la HeartbeatRequest reçue.

Si un message HeartbeatResponse reçu ne contient pas la charge utile attendue, le message DOIT être éliminé en silence. Si il contient bien la charge utile attendue, le temporisateur de retransmission DOIT être arrêté.

## 5. Cas d'utilisation

Chaque point d'extrémité envoie des messages HeartbeatRequest à un taux et avec le bourrage requis pour le cas d'utilisation particulier. Le point d'extrémité ne devrait pas s'attendre à ce que son homologue envoie des HeartbeatRequest. Les deux directions sont indépendantes.

### 5.1 Découverte de la MTU de chemin

DTLS effectue la découverte de la MTU de chemin comme décrit au paragraphe 4.1.1.1 de la [RFC6347]. Une description détaillée de la façon d'effectuer la découverte de la MTU de chemin est donnée dans la [RFC4821]. Les paquets de sonde nécessaires sont les messages HeartbeatRequest.

Cette méthode d'utilisation des messages HeartbeatRequest pour DTLS est similaire à celle du protocole de transmission de commandes de flux (SCTP) qui utilise le tronçon de bourrage (PAD-chunk) défini dans la [RFC4820].

### 5.2. Vérification de vie

L'envoi de messages HeartbeatRequest permet à l'envoyeur de s'assurer qu'il peut joindre l'homologue et que l'homologue est en vie. Même dans le cas de TLS/TCP, cela permet une vérification à un taux plus élevé que ce que le dispositif de garde en vie de TCP permettrait.

À côté de s'assurer que l'homologue est toujours joignable, l'envoi de messages HeartbeatRequest rafraîchit l'état de NAT de tous les NAT impliqués.

Les messages HeartbeatRequest DEVRAIENT seulement être envoyés après une période d'inactivité d'au moins plusieurs délais d'aller-retour. Cette période d'inactivité DEVRAIT être configurable jusqu'à une période de plusieurs minutes et de pas moins d'une période d'une seconde. Une valeur par défaut pour la période d'inactivité DEVRAIT être configurable, mais elle DEVRAIT aussi être réglable homologue par homologue.

## 6. Considérations relatives à l'IANA

L'IANA a alloué le type de contenu heartbeat (24) dans le "Registre des types de contenu TLS" spécifié dans la [RFC5246]. La référence est à la RFC 6520.

L'IANA a créé et tient maintenant un nouveau registre pour les types de messages Heartbeat. Les types de messages sont

des nombres dans la gamme de 0 à 255 (décimal). L'IANA a alloué les types de message heartbeat_request (1) et heartbeat_response (2). Les valeurs de 0 et 255 devraient être réservées. Ce registre utilise la politique de revue par expert comme décrit dans la [RFC5226]. La référence est à la RFC 6520.

L'IANA a alloué le type d'extension heartbeat (15) dans le registre TLS des "Valeurs de type d'extension" comme spécifié dans la [RFC5246]. La référence est à la RFC 6520.

L'IANA a créé et tient un nouveau registre pour les modes Heartbeat. Les modes sont des nombres dans la gamme de 0 à 255 (décimal). L'IANA a alloué les modes peer_allowed_to_send (1) et peer_not_allowed_to_send (2). Les valeurs 0 et 255 devraient être réservées. Ce registre utilise la politique de revue par expert comme décrit dans la [RFC5226]. La référence est à la RFC 6520.

## 7. Considérations sur la sécurité

Les considérations sur la sécurité des [RFC5246] et [RFC6347] s'appliquent au présent document. Ce document n'introduit aucune nouvelle considération de sécurité.

## 8. Remerciements

Les auteurs souhaitent remercier Pasi Eronen, Adrian Farrel, Stephen Farrell, Adam Langley, Nikos Mavrogiannopoulos, Tom Petch, Eric Rescorla, Peter Saint-Andre, et Juho Vaehae-Herttua de leurs précieux commentaires.

## 9. Références

### 9.1 Références normatives

[RFC2119] S. Bradner, "Mots clés à utiliser dans les RFC pour indiquer les niveaux d'exigence", BCP 14, mars 1997. *(MàJ par* RFC8174)

[RFC5226] T. Narten et H. Alvestrand, "Lignes directrices pour la rédaction d'une section Considérations relatives à l'IANA dans les RFC", BCP 26, mai 2008. (*Remplace* RFC2434 *; remplacée par* RFC8126)

[RFC5246] T. Dierks, E. Rescorla, "Version 1.2 du protocole de sécurité de la couche Transport (TLS)", août 2008. *(P.S.* ; *remplace* RFC3268, 4346, 4366 ; *MàJ* RFC4492 ; *rendue obsolète par la* RFC8446)

[RFC6066] D. Eastlake 3rd, " Extensions à la sécurité de la couche Transport (TLS) : Définitions d'extension", janvier 2011. *(P.S. ; remplace* RFC4366 *; MàJ par* RFC8446, RFC8449)

[RFC6347] E. Rescorla, N. Modadugu, "Sécurité de la couche transport de datagrammes, version 1.2", janvier 2012. *(Remplace la* RFC4347) *(P.S. ; MàJ par* RFC7905, 9146 ; *remplacée par* RFC9147)

### 9.2 Références pour information

[RFC3436] A. Jungmaier, E. Rescorla, M. Tüexen, "Sécurité de la couche Transport sur le protocole de transmission de contrôle de flux", décembre 2002. *(P.S.)*

[RFC4820] M. Tüexen et autres, "Tronçon de bourrage et paramètres pour le protocole de transmission de contrôle de flux (SCTP)", mars 2007. *(P.S.)*

[RFC4821] M. Mathis, J. Heffner, "Découverte de la MTU de chemin de couche de mise en paquet", mars 2007. *(P.S.)*

[RFC5238] T. Phelan, "Sécurité de la couche de transport de datagrammes (DTLS) sur le protocole de contrôle de l'encombrement de datagrammes (DCCP**),**" mai 2008. *(P.S.)*

[RFC6083] M. Tüexen, R. Seggelmann et E. Rescorla, "Sécurité de la couche Transport de datagrammes (DTLS) pour le protocole de transmission de contrôle de flux (SCTP)", janvier 2011. *(P.S.)*

## Adresse des auteurs

Robin Seggelmann
Muenster University of Applied Sciences
Stegerwaldstr. 39
48565 Steinfurt
DE
mél : seggelmann@fh-muenster.de

Michael Tüexen
Muenster University of Applied Sciences
Stegerwaldstr. 39
48565 Steinfurt
DE
mél : tuexen@fh-muenster.de

Michael Glenn Williams
GWhiz Arts & Sciences
2885 Denise Court
Newbury Park, CA, 91320
USA
mél : michael.glenn.williams@gmail.com