Groupe de travail sur les réseaux | K. Sollins
Requête pour Commentaires : 1350 | MIT
Standard : 33 | Juillet 1992
Remplace : RFC 783
Traduction : Yves LESCOP (lycée la croix-rouge - Brest)
Relecteur : François ROPERT (www.rezalfr.org)

PROTOCOLE TFTP (REVISION 2)

Statut de ce document

Ce document spécifie un protocole standard d'Internet pour la communauté Internet, et ne sera éprouvé qu'après plusieurs discussions et suggestions. Merci de vous référer à l'édition courante du " Internet Official Protocol Standards " (STD1) pour l'état de standardisation et le statut de ce protocole. La distribution de ce document est illimitée.

Résumé

TFTP est un protocole très simple utilisé pour transférer des fichiers. Son appellation (Trivial File Transfer Protocol ou TFTP) vient de là. Chaque paquet "nonterminal" est validé séparément. Ce document décrit le protocole et ses différents types de paquets. Ce document explique aussi les raisons ayant conduit à certaines décisions.

Remerciements

Ce protocole fut conçu à l'origine par Noel Chiappa, et fut révisé par lui-même, Bob Baldwin et Dave Clark, avec des commentaires de Steve Szymanski. La présente révision de ce document inclus des modifications provenant des discussions et des suggestions de Larry Allen, Noel Chiappa, Dave Clark, Geoff Cooper, Mike Greenwald, Liza Martin, David Reed, Craig Milo Rogers (de USC-ISI), Kathy Yellick, et l'auteur. Le schéma utilisé pour la reconnaissance et la retransmission s'inspire de TCP, et le mécanisme de gestion d'erreur a été suggéré par le système de message d'avortement du PARC EFTP.

La révision de Mai 1992, remédiant au bug protocolaire de l' "apprenti sorcier" [4] et à d'autres problèmes mineurs, a été faite par Noel Chiappa.

Cette étude a été soutenue par "The Advanced Research Projects Agency of the Department of Defense" et a été contrôlée par "The Office of Naval Research" sous le contrat numéro N00014-75-C-0661.

1. Objectif

TFTP est un protocole basique de transfert de fichiers, et par conséquent est appelé "Trivial File Transfer Protocol" ou TFTP. Il a été implanté au dessus du protocole Internet UDP ("User Datagram Protocol") [2] aussi peut il être utilisé pour déplacer des fichiers entre machines sur différents réseaux implantant UDP. (Ceci n'exclut pas la possibilité d'implémenter TFTP au dessus d'un autre protocole fournissant des datagrammes). Il est conçu pour être réduit et facile à implémenter. Il lui manque donc la plupart des fonctionnalités d'un FTP ordinaire. La seule chose qu'il peut réaliser est lire et écrire des fichiers (ou du courrier) depuis ou vers un serveur distant. Il ne peut pas afficher le contenu d'un répertoire, et actuellement l'authentification des utilisateurs n'est pas prévue. Comme les autres protocoles Internet, il travaille avec des données de longueur égale à 8 bits.

Trois modes de transferts sont actuellement soutenus : "netascii" (ASCII est défini dans "USA Standard Code for Information Interchange" [1] avec les modifications précisées dans "Spécification du protocole Telnet" [3].) Noter qu'il s'agit d'un ASCII 8 bits. Le terme "netascii" sera utilisé tout au long de ce document pour indiquer cette version particulière de l'ASCII. ; Le terme "octet" (celui-ci remplace le mode "binaire" des versions précédentes de ce document) est un octet brut de 8 bits. Le terme "mail", représente des caractères netascii envoyés à un utilisateur plutôt qu'un fichier (le mode "mail" est obsolète et ne doit pas être implanté ou utilisé). Des modes supplémentaires peuvent être définis par des paires de machines coopérant entre elles.

La référence [4] (section 4.2) devra être consultée pour de précieuses directives et futures suggestions sur TFTP.

2. Vue d'ensemble du protocole

N'importe quel transfert démarre par une demande de lecture ou d'écriture de fichier, qui aussi sert de demande de connexion. Si le serveur autorise la requête, la connexion est ouverte et le fichier est envoyé par blocs d'une taille fixe de 512 octets. Chaque paquet de données contient un bloc de données, et doit être acquitté par un paquet "accusé de réception" avant que le paquet suivant ne puisse être émis. Un paquet de données de moins de 512 octets signale la terminaison du transfert. Si un paquet se perd sur le réseau, une fin d'attente se déclenchera chez le destinataire et il pourra retransmettre son dernier paquet (qui peut être de données ou un accusé de réception), provoquant ainsi la retransmission du paquet perdu par l'émetteur. L'émetteur, depuis l'étape garantissant que tous les anciens paquets ont bien été reçus, ne doit conserver qu'un paquet en mémoire pour la retransmission. Observons que les deux machines concernées par un transfert sont considérées comme émettrices et réceptrices. L'une envoie des données et reçoit des accusés de réception, l'autre envoie des accusés de réception et reçoit des données.

La plupart des erreurs provoquent la rupture de la connexion. Une erreur est signalée par l'émission d'un paquet "erreur". Ce paquet n'est ni acquitté ni retransmis (par exemple., un serveur TFTP ou un utilisateur peut rompre sa connexion après l'envoi d'un message d'erreur), alors l'autre extrémité de la connexion peut ne pas l'avoir reçu. Par conséquent des délais d'attente sont utilisés quand le paquet "erreur" a été perdu pour détecter une telle terminaison. Les erreurs sont provoquées par trois types d'événements : incapacité à satisfaire la demande (par exemple, fichier non trouvé, violation d'accès, ou utilisateur inexistant), réception d'un paquet qui ne peut s'expliquer par un délai ou une duplication sur le réseau (par exemple, un paquet mal formé), ou la perte de l'accès à une ressource nécessaire (par exemple, disque plein ou accès refusé pendant le transfert).

TFTP ne reconnaît qu'une seule condition d'erreur qui ne provoque pas la rupture : le port source d'un paquet reçu est incorrect. Dans ce cas, un paquet d'erreur est émis vers la machine d'origine.

Ce protocole est très restrictif afin de simplifier l'implémentation. Par exemple, la taille fixe des blocs prépare franchement l'allocation future, et l'étape d'attente d'accusé de réception fournit un mécanisme de contrôle de flux et évite le besoin de remettre dans l'ordre les paquets entrants.

3. Relation aux autres Protocoles

Comme mentionné, TFTP est conçu pour être implanté au dessus d'un protocole datagramme (UDP). Puisque le Datagramme est implanté sur le protocole Internet, les paquets possèdent une en-tête Internet, une en-tête de datagramme, et une en-tête TFTP. En plus, les paquets peuvent avoir une en-tête (LNI, ARPA, etc.) pour leur permettre de circuler sur le support de transmission local. Comme indiqué dans la figure 3-1, l'ordre des contenus d'un paquet sera : en-tête support local, si utilisé, en-tête Internet, en-tête Datagramme, en-tête TFTP, suivis par le reste du paquet TFTP. Ceci peut ou non dépendre des données et du type de paquet comme indiqué dans l'en-tête TFTP. TFTP ne précise directement aucune valeur dans l'en-tête Internet. Par contre, le port source et le port destination de l'en-tête du Datagramme (son format est donné dans l'appendice) sont utilisés par TFTP et le champ longueur reflète la taille du paquet TFTP. Les identificateurs de transfert (TID) utilisés par TFTP sont transmis à la couche Datagramme pour être utilisés comme ports ; donc ils doivent être compris entre 0 et 65535. L'initialisation des TID est discutée dans la section sur le protocole initial de connexion.

L'en-tête TFTP est constituée d'un champ de 2 octets qui indique le type du paquet (par exemple, DONNEE, ERREUR, etc.). Ces codes et le format des différents types de paquets sont discutés plus loin dans la section sur les paquets TFTP.

```
 ---------------------------------------------------------
|  Support Local  |  Internet  |  Datagramme  |  TFTP  |
 ---------------------------------------------------------
```

Figure 3-1: Ordre des en-têtes

4. Protocole initial de connexion

Un transfert est établi par l'émission d'une requête (WRQ pour écrire vers un système de fichier étranger, ou RRQ pour le lire), et la réception d'une réponse positive, un accusé de réception pour écrire, ou le premier paquet de données à lire. En général un paquet "accusé de réception" doit contenir le numéro de bloc du paquet de données qui doit être acquitté. A chaque paquet de données est associé un numéro de bloc ; les numéros de blocs sont consécutifs et démarrent à 1. Puisque la réponse positive à une demande d'écriture est un paquet "accusé de réception", dans ce cas particulier le numéro de bloc sera zéro. Normalement, puisqu'un paquet "accusé de réception" valide un paquet de données, le paquet "accusé de réception" doit contenir le numéro de bloc du paquet de données à valider. Si la réponse est un paquet « erreur », alors la requête est rejetée.

Chaque extrémité choisit en priorité un TID pour elle même afin d’établir une connexion. Il sera utilisé durant cette connexion. Le TID d'une connexion sera choisit aléatoirement, ainsi la probabilité que le même nombre soit choisit deux fois de suite est très faible. A chaque paquet sont associés les deux TID de la phase de connexion, le TID source et le TID destination. Ces TID sont remis au support UDP (ou un autre protocole datagramme) comme ports source et destination. Une machine effectuant une demande choisit son TID source comme décrit ci-dessus, et émet sa requête initiale avec le TID réservé 69 en décimal (105 en octal) pour la machine destinataire. La réponse à la demande, en fonctionnement normal, utilise le TID choisi par le serveur comme TID source et le TID choisi par le requérant dans son message préalable comme TID destination. Les deux TID choisis sont alors utilisés pour le reste du transfert.

Par exemple, la suite montre les étapes utilisées pour établir une connexion dans le but d'écrire un fichier. Noter que WRQ, ACK, et DATA sont les noms respectifs des types de paquets demande d'écriture, accusé de réception, et données. L'appendice contient un exemple similaire pour la lecture d'un fichier.

1. La machine A émet un "WRQ" vers la machine B avec source = TID de A, destination = 69.

2. La machine B émet un "ACK" (avec numéro de bloc = 0) vers la machine A avec source = TID de B, destination = TID de A.

A cet instant la connexion a été établie et le premier paquet de données peut être émis par la machine A avec un numéro de séquence à 1. Dans la prochaine étape, et dans toutes les suivantes, les machines doivent s'assurer que le TID source est égal à la valeur convenue lors des étapes 1 et 2. Si un TID source n'est pas le même, le paquet doit être considéré comme provenant par erreur d'un autre endroit. Un paquet "erreur" doit être envoyé à la source du mauvais paquet, tandis que le transfert n'est pas perturbé. Ceci ne peut être réalisé que si le TFTP reçoit effectivement un paquet avec un TID incorrect. Si le protocole de transport ne le permet pas, cette condition d'erreur particulière n'arrivera pas.

L'exemple suivant illustre une opération correcte du protocole dans laquelle la situation ci-dessus peut survenir. La machine A émet une requête vers la machine B. Quelque part sur le réseau, la demande est dupliquée, et deux accusés de réceptions sont renvoyés vers la machine A, avec des

TID différents choisis par la machine B en réponse aux deux requêtes. Quand la première réponse arrive, la machine A continue la connexion. Quand la seconde réponse à la demande arrive, elle doit être rejetée, mais il n'y a aucune raison de fermer la première connexion. Donc, si des TID différents sont choisis pour les deux connexions sur la machine B et que la machine A vérifie le TID source des messages reçus, la première connexion peut être maintenue tandis que la seconde est rejetée par le renvoi d'un paquet "erreur".

5. Paquets TFTP

TFTP reconnaît cinq types de paquets, tous ceux-ci sont mentionnés ci-dessous :

code opération	Opération
1	Demande de lecture (RRQ)
2	Demande d'écriture (WRQ)
3	Données (DATA)
4	Accusé de réception (ACK)
5	Erreur (ERROR)

L'en-tête d'un paquet TFTP contient le code opération associé à ce paquet.

```
  2 Octets       chaîne      1 octet    chaîne   1 octet
 --------------------------------------------------------
| code op. | Nom de fichier |   0   |    Mode    |   0   |
 --------------------------------------------------------

                Figure 5-1: paquet RRQ/WRQ
```

Les paquets RRQ et WRQ (respectivement codes opération 1 et 2) ont le format illustré Figure 5-1. Le nom de fichier est une suite d'octets en "netascii" terminée par un octet à zéro. Le champ « mode » contient la chaîne de caractères "netascii", "octet", ou "mail" (ou toute combinaison de majuscules et minuscules, comme "NETASCII", "NetAscii", etc.) en "netascii" indiquant l'un des trois modes défini par le protocole. Une machine qui reçoit des données en mode "netascii" doit traduire les données dans son propre format. Le mode Octet est utilisé pour transférer un fichier qui est dans le format 8-bits de la machine de laquelle le fichier est en train d'être transféré. Cela suppose que chaque type de machine possède un seul format 8-bit ce qui est le plus courant, et que ce format est choisi. Par exemple, sur un DEC-20, une machine 36 bits, il y a quatre octets dans un mot plus quatre bits de rupture. Si une machine reçoit un fichier « octets » et le retourne, le fichier retourné doit être identique à l'original.
le mode courrier (Mail) utilise le nom d'une adresse de courrier au lieu d'un fichier, et doit commencer par un WRQ. Sinon il est identique au mode "netascii". La chaîne de caractères de l'adresse de courrier doit être de la forme "nom utilisateur" ou "nom@machine". Si la deuxième syntaxe est utilisée, elle autorise l'option de relayage du courrier par un ordinateur faisant office de relais.

La discussion ci-dessus présume que l'émetteur et le récepteur fonctionnent tous deux dans le même mode, mais il n'y a aucune raison pour que ce soit le cas. Par exemple, l'un peut être un serveur de stockage. Il n'y a aucune raison qu'une telle machine puisse traduire le "netascii" dans son propre format de texte. L'expéditeur peut envoyer des fichiers en "netascii", mais le serveur de stockage peut simplement l'enregistrer sans aucune translation en format 8-bits. Une autre situation comparable est un problème qui se produit couramment sur des systèmes DEC-20. Ni le mode "netascii", ni le mode octet ne donnent accès à tous les bits d'un mot. On peut créer pour une telle machine un mode spécial qui lit tous les bits d'un mot, mais dans lequel le récepteur enregistre l'information dans un format 8-bits. Quand un tel fichier est récupéré du site de stockage, il doit l'être sous sa forme originale pour être utilisable, alors le mode inverse doit aussi être implémenté. L'utilisateur du site devra se souvenir de cette information pour réaliser le transfert. Dans ces deux exemples, le paquet de demande devra spécifier le mode octet pour la machine étrangère, mais la machine locale devra être dans un autre mode. Aucune machine ou modes spécifiques d'application n'ont été spécifiés dans le TFTP, mais l'un d'eux pourra être compatible avec cette spécification.

Il est aussi possible de concevoir d'autres modes pour des paires de machines coopérantes, bien que cela doit être réalisé avec précaution. Il n'y a aucune obligation pour que d'autres machines puissent implémenter cela. Il n'existe aucune autorité de contrôle pour définir ces modes ou désigner leurs noms.

```
  2 octets     2 octets      n octets
 --------------------------------------
| Code op. |   Bloc #  |   Données     |
 --------------------------------------
```

Figure 5-2: paquet DONNEES

Les données sont effectivement transférées dans des paquets DONNEES illustrés dans la Figure 5-2. Les paquets DONNEES (code opération = 3) possèdent un numéro de bloc et un champ de données. Le numéro de bloc du paquet de données débute à 1 et s'incrémente pour chaque nouveau bloc de données. Cette limitation permet au programme de n'utiliser qu'un seul nombre pour discriminer les nouveaux paquets ou ceux qui sont dupliqués. Le champ de données est long de 0 à 512 octets. S'il est long de 512 octets, le bloc n'est pas le dernier bloc de données; s'il est long de 0 à 511 octets, il indique la fin du transfert. (pour plus de détails voir la section sur la Conclusion Normale.)

Tous les paquets autres que les accusés de réception dupliqués et ceux utilisés pour la conclusion sont acquittés à moins qu'une fin d'attente ne survienne [4]. Emettre un paquet de données est une manière de reconnaître le premier paquet "accusé de réception" du paquet de données préalable. Les paquets WRQ et DONNEES sont acquittés par les paquets ACK ou ERREUR, tandis que les paquets RRQ et ACK sont acquittés par les paquets DONNEES ou ERREUR.

```
  2 octets     2 octets
 ----------------------
| Code op. |   Bloc #  |
 ----------------------
```

Figure 5-3: paquet ACK

La Figure 5.3 décrit un paquet ACK ; le code opération est 4. Le numéro de bloc dans ACK est l'écho du numéro de bloc du paquet de données qui doit être acquitté. Un paquet WRQ est acquitté par un paquet ACK ayant un numéro de bloc nul.

```
 2 octets     2 octets      chaîne   1 octet
 -------------------------------------------
| Code op. | Code erreur |  msg Err |   0   |
 -------------------------------------------

          Figure 5-4: paquet ERREUR
```

Un paquet Erreur (code opération 5) prend la forme décrite dans la Figure 5-4. Un paquet erreur peut servir d'accusé de réception de n'importe quel autre type de paquet. Le code erreur est un entier précisant la nature de l'erreur. Une table des valeurs et de leurs significations est donnée dans l'appendice. Noter que plusieurs codes d'erreurs ont été ajoutés à cette version du document. Le message d'erreur est destiné à un interlocuteur humain, et doit être en "netascii". Comme toutes les autres chaînes de caractères, il est terminé par un octet nul.

6. Conclusion Normale

La fin d'un transfert est balisée par un paquet de données qui contient entre 0 et 511 octets de données (par exemple, longueur du Datagramme < 516). Ce paquet est validé par un paquet ACK comme tous les autres paquets de données. La machine qui accuse réception du paquet de données final peut terminer la connexion de son coté en émettant l'ACK final. D'autre part, il est recommandé de s'attarder, cela signifie que la machine émettant l'ACK final attendra un instant avant de conclure afin de retransmettre l'ACK final s'il a été perdu. L'émetteur de l'accusé de réception saura que l'ACK a été perdu s'il reçoit à nouveau le dernier paquet de données. La machine émettant le dernier paquet de données doit le retransmettre jusqu'à ce qu'il soit acquitté ou qu'une fin d'attente arrive. Si la réponse est un ACK, la transmission est une réussite complète. Si un délai d'attente s'écoule chez l'émetteur et qu'il n'est plus prêt à retransmettre, le transfert peut toujours être un succès, après que celui qui accuse réception ou le réseau peuvent avoir rencontré un problème. Il est aussi possible dans ce cas que le transfert soit un échec. Dans tous les cas, la connexion est rompue.

7. Conclusion Prématurée

Si une requête ne peut être accordée, ou qu'une erreur se produit durant le transfert, alors un paquet ERREUR (code opération 5) est émis. Ce n'est qu'une politesse puisqu'il ne peut être retransmis ou acquitté, ainsi il peut n'être jamais reçu. Des fins d'attente doivent aussi être utilisées pour détecter les erreurs.

I. Appendice

Ordre des en-têtes

```
                                                      2 octets
    --------------------------------------------------------------
   |  Support Local  |  Internet  |  Datagramme  | Code op. TFTP  |
    --------------------------------------------------------------
```

Formats TFTP

```
   Type       Op #     Format sans l'en-tête

             2 octets   chaîne          1 octet   chaîne    1 octet
             ------------------------------------------------------
   RRQ/WRQ  | 01/02  |  Nom du fichier  |   0  |    Mode    |   0  |
             ------------------------------------------------------
             2 octets   2 octets      n octets
             ----------------------------------
   DATA     |  03    |   Bloc #    |   Données  |
             ----------------------------------
             2 octets   2 octets
             ---------------------
   ACK      |  04    |   Bloc #    |
             ---------------------
             2 octets   2 octets      chaîne    1 octet
             ------------------------------------------
   ERROR    |  05    | Code Erreur |  Msg Err   |   0  |
             ------------------------------------------
```

Protocole Initial de Connexion pour lire un fichier

1. La machine A émet un "RRQ" vers la machine B avec source = son TID, destination = 69.

2. La machine B émet un paquet de données "DATA" (avec numéro de bloc = 1) vers la machine A avec source = TID de B, destination = TID de A.

Codes des Erreurs

Valeur	Signification
0	Non défini, voir le message d'erreur (si présent).
1	Fichier non trouvé.
2	Violation d'accès.
3	Disque plein ou dépassement de l'espace alloué.
4	Opération TFTP illégale.
5	Transfert ID inconnu.
6	Le fichier existe déjà.
7	Utilisateur inconnu.

En-tête du datagramme Internet de l'utilisateur [2]

Ceci a été inclus uniquement par commodité. TFTP n'est pas obligatoirement implanté au-dessus d'UDP (User Datagram Protocol).

```
     Format

    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |          Port Source          |       Port Destination        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           Longueur            |       Somme de contrôle       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Valeurs des champs :	
Port Source	Choisi par l'émetteur du paquet.
Port Destination	Choisi par la machine destination (69 pour RRQ ou WRQ).
Longueur	Nombre d'octets dans le paquet UDP, en-tête UDP inclus.
Somme de contrôle	La référence 2 décrit les règles de calcul de la somme de contrôle. L'implémenteur doit être certain d'utiliser ici l'algorithme correct. Champ contenant des zéros si inutilisé.

Note : TFTP passe les identificateurs de transfert (TID) à UDP (User Datagram Protocol) pour qu'ils soient utilisés comme ports source et destination.

Références

[1] USA Standard Code for Information Interchange, USASI X3.4-1968.

[2] Postel, J., "User Datagram Protocol," RFC 768, USC/Information Sciences Institute, 28 August 1980.

[3] Postel, J., "Telnet Protocol Specification," RFC 764, USC/Information Sciences Institute, June, 1980.

[4] Braden, R., Editor, "Requirements for Internet Hosts -- Application and Support", RFC 1123, USC/Information Sciences Institute, October 1989.

Considérations Sécuritaires

Puisque TFTP n'inclus pas de mécanismes d'identification ou de contrôle d'accès, la prudence doit être de mise dans l'accord des droits au processus serveur TFTP de manière à ne pas violer la sécurité du système de fichiers de la machine serveur. TFTP est souvent installé avec des contrôles tels que seuls les fichiers ayant un accès public en lecture sont disponibles via TFTP et l'écriture via TFTP n’est pas autorisée.

Adresse de l'auteur

Karen R. Sollins
Massachusetts Institute of Technology
Laboratory for Computer Science
545 Technology Square
Cambridge, MA 02139-1986

Phone: (617) 253-6006

EMail: SOLLINS@LCS.MIT.EDU