Groupe de travail Réseau
Request for Comments : 1413
RFC rendue obsolète : 931
M. St. Johns, US Department of Defense
février 1993
Traduction Claude Brière de L'Isle

Protocole d'identification

Statut de ce mémoire
La présente RFC spécifie un protocole normalisé par l'IAB pour la communauté de l'Internet et appelle à des discussions et suggestions pour son amélioration. Prière de se reporter à l'édition en cours des "normes officielles de protocole de l'IAB" pour l'état de normalisation et le statut de ce protocole. La distribution du présent mémoire n'est soumise à aucune restriction.

1. Introduction

Le protocole d'identification (aussi dit "ident", ou "Ident Protocol") donne le moyen de déterminer l'identité d'un usager d'une connexion TCP particulière. Étant donnée une paire de numéros d'accès TCP, il retourne une chaîne de caractères qui identifie le propriétaire de cette connexion sur le système du serveur.

Le protocole d'identification était précédemment appelé le protocole de serveur d'authentification. Il a été débaptisé pour mieux refléter cette fonction. Le présent document a été produit par le groupe de travail Protocole d'identité du client TCP de l'équipe d'ingénierie de l'Internet (IETF, *Internet Engineering Task Force*).

2. Généralités

C'est une application fondée sur la connexion sur TCP. Un serveur écoute les connexions TCP sur l'accès TCP 113 (décimal). Une fois qu'une connexion est établie, le serveur lit une ligne de données qui spécifie la connexion qui l'intéresse. Si il existe, l'identifiant d'utilisateur, dépendant du système, de la connexion concernée est envoyé en réponse. Le serveur peut alors soit clore la connexion, soit continuer à lire/répondre les diverses interrogations.

Le serveur devrait clore la connexion après un délai configurable sans interrogation – un délai de 60 à 180 secondes d'inactivité est recommandé. Le client peut clore la connexion à tout moment ; cependant, pour tenir compte des délais du réseau, le client devrait attendre au moins 30 secondes (ou plus) après une interrogation avant d'abandonner et clore la connexion.

3. Restrictions

Les interrogations ne sont permises que pour les connexions pleinement spécifiées. L'interrogation contient la paire d'accès local/distant – la paire d'adresses locale/distante utilisée pour spécifier pleinement la connexion est tirée de l'adresse locale et de l'adresse distante de la connexion qui interroge. Cela signifie qu'un usager à l'adresse A peut seulement interroger le serveur à l'adresse B sur les connexions entre A et B.

4. Format d'interrogation/réponse

Le serveur accepte des demandes d'interrogation en texte simple de la forme :

<accès-sur-serveur> , <accès-sur-client>

où <accès-sur-serveur> est l'accès TCP (en décimal) sur le système cible (où le serveur "ident" fonctionne) et <accès-sur-client> est l'accès TCP (en décimal) sur le système source (client).

Note : si un client sur l'hôte A veut interroger un serveur sur l'hôte B sur une connexion spécifiée en local (sur la machine du client) comme 23, 6191 (une connexion TELNET entrante) le client doit en fait interroger sur 6191, 23 – qui est la façon dont la connexion va être spécifiée sur l'hôte B.

Par exemple pour :

6191, 23

La réponse est de la forme :

<accès-sur-serveur> , <accès-sur-client> : <resp-type> : <add-info>

où <accès-sur-serveur>,<accès-sur-client> sont la même paire que l'interrogation, <resp-type> est un mot clé qui identifie le type de réponse, et <add-info> dépend du contexte.

Les informations retournées sont celles associées à la connexion TCP pleinement spécifiée identifiée par <adresse-serveur>, <adresse-client>, <accès-sur-serveur>, <accès-sur-client>, où <adresse-serveur> et <adresse-client> sont les adresses IP locale et distante de la connexion qui interroge –c'est-à-dire, la connexion TCP au serveur du protocole d'identification. (<accès-sur-serveur> et <accès-sur-client> sont tirés de l'interrogation.)

Par exemple :

6193, 23 : USERID : UNIX : stjohns
6195, 23 : ERROR : NO-USER

5. Types de réponse

Une réponse peut être d'un des deux types suivants :

USERID *(identifiant d'utilisateur)*

Dans ce cas, <add-info> est une chaîne qui consiste en un nom de système d'exploitation (avec un identifiant de jeu de caractère facultatif), suivi par ":", suivi par une chaîne d'identification.

Le jeu de caractères (si il est présent) est séparé du nom du système d'exploitation par ",". L'identifiant de jeu de caractères est utilisé pour indiquer le jeu de caractères de la chaîne d'identification. L'identifiant de jeu de caractères, si il est omis, prend par défaut "US-ASCII" (voir ci-dessous).

Les noms de systèmes d'exploitation et les noms de jeux de caractères permis sont spécifiés dans la RFC 1340, "Numéros alloués" ou ses successeurs.

En plus des noms de système d'exploitation et de jeux de caractères spécifiés dans "Numéros alloués", il y a un cas particulier d'identifiant de système d'exploitation - "OTHER".

Sauf si "OTHER" est spécifié comme type de système d'exploitation, le serveur est supposé retourner l'identification d'utilisateur "normale" du propriétaire de cette connexion. "Normal" dans ce contexte peut être pris comme signifiant une chaîne de caractères qui identifie de façon univoque le propriétaire de la connexion comme un identifiant d'utilisateur alloué par l'administrateur du système et utilisé par un tel usager comme identifiant de messagerie, ou comme la partie "usager" d'une paire usager/mot de passe utilisée pour obtenir l'accès aux ressources du système. Lorsque un système d'exploitation est spécifié (par exemple, tout sauf "OTHER"), l'identifiant d'utilisateur est supposé être sous une forme plus ou moins immédiatement utilisable – par exemple, quelque chose qui pourrait être utilisé comme un argument pour "finger" ou comme adresse de messagerie.

"OTHER" indique que l'identifiant est une chaîne de caractères non formatée consistant en caractères imprimables dans le jeu de caractères spécifié. "OTHER" devrait être spécifié si l'identifiant d'utilisateur ne satisfait pas aux contraintes du paragraphe précédent. L'envoi d'un jeton d'audit chiffré; ou le retour d'autres informations qui ne sont pas l'identifiant d'utilisateur sur un usager (comme le nom et le numéro de téléphone réel d'un usager à partir d'un fichier UNIX passwd) sont deux exemples de quand "OTHER" devrait être utilisé.

Les identifiants d'utilisateur retournés sont supposés être imprimables dans le jeu de caractères indiqué.

L'identifiant est une chaîne d'octets non formatée – tous les octets sont permis SAUF les octals 000 (NUL), 012 (LF) et 015 (CR).

Note : les caractères espace (040) suivant le séparateur deux-points FONT partie de la chaîne d'identifiant et ne doivent pas être ignorés. Une chaîne de réponse est quand même terminée normalement par un CR/LF. Une chaîne peut être imprimable, mais n'est pas *nécessairement* imprimable.

ERROR *(erreur)*

Pour une raison quelconque, le propriétaire de l'accès n'a pas pu être déterminé, <add-info> dit pourquoi. Les valeurs qui suivent sont permises pour <add-info> avec leur signification :

INVALID-PORT *(accès invalide)*
L'accès local ou distant a été improprement spécifié. Cela devrait être retourné si l'un des identifiants d'accès, ou les deux sont hors gamme (les numéros d'accès TCP sont de 1 à 65 535), sont des entiers négatifs, réels, ou d'une façon quelconque non reconnus comme entiers non négatifs.

NO-USER *(pas d'utilisateur)*
La connexion spécifiée par la paire d'accès n'est pas actuellement utilisée ou pas actuellement possédée par une entité identifiable.

HIDDEN-USER *(utilisateur caché)*
Le serveur a été capable d'identifier l'utilisateur de cet accès, mais les informations n'ont pas été retournées à la demande de l'usager.

UNKNOWN-ERROR *(erreur inconnue)*
On ne peut pas déterminer le propriétaire de la connexion ; raison inconnue. Toute erreur non couverte par les cas ci-dessus devrait retourner cette valeur de code d'erreur. Facultativement, ce code PEUT être retourné à la place de tout autre code d'erreur spécifique si, par exemple, le serveur désire cacher les informations impliquées par le retour de ce code d'erreur, ou pour toute autre raison. Si un serveur met en œuvre une telle disposition, il DOIT être configurable et il DOIT par défaut retourner le message d'erreur approprié.

D'autres valeurs peuvent éventuellement être spécifiées et définies dans de futures révisions du présent document. Si une mise en œuvre a besoin de spécifier un code d'erreur non standard, ce code doit commencer par "X".

De plus, il est permis au serveur d'abandonner la connexion interrogatrice sans répondre. Toute clôture prématurée (c'est-à-dire, où le client ne reçoit pas le EOL *(*end of line*, fin de ligne)* que ce soit en douceur ou par une interruption, devrait être considérée comme ayant la même signification que "ERROR : UNKNOWN-ERROR".

Syntaxe formelle

<request> ::= <port-pair> <EOL>

<port-pair> ::= <integer> "," <integer>

<reply> ::= <reply-text> <EOL>

<EOL> ::= "015 012" ; CR-LF indicateur de fin de ligne

<reply-text> ::= <error-reply> | <ident-reply>

<error-reply> ::= <port-pair> ":" "ERROR" ":" <error-type>

<ident-reply> ::= <port-pair> ":" "USERID" ":" <opsys-field> ":" <user-id>

<error-type> ::= "INVALID-PORT" | "NO-USER" | "UNKNOWN-ERROR" | "HIDDEN-USER" | <error-token>

<opsys-field> ::= <opsys> ["," <charset>]

<opsys> ::= "OTHER" | "UNIX" | <token> ...etc. ; (voir "Numéros alloués")

<charset> ::= "US-ASCII" | ...etc. ; (voir "Numéros alloués")

<user-id> ::= <octet-string>

<token> ::= 1*64<token-characters> ; 1 à 64 caractères

<error-token> ::= "X"1*63<token-characters> ; 2 à 64 caractères commençant par w/X

<integer> ::= 1*5<digit> ; 1 à 5 chiffres

<digit> ::= "0" | "1" ... "8" | "9" ; 0 à 9

<token-characters> ::= <Tous les caractères ASCII : a-z, A-Z, - (tiret), .!@#$%^&*()_=+.,<>/?"'~`{}[]; >
; a à z minuscules et majuscules plus les imprimables moins le caractère deux-points ":".

<octet-string> ::= 1*512<octet-characters>

<octet-characters> ::= <tout octet de 00 à 377 (octal) sauf les ASCII NUL (000), CR (015) et LF (012)>

Notes sur la syntaxe :

1) Pour promouvoir l'interopérabilité entre les variantes de mise en œuvre, à l'égard des espaces blanches, on comprendra que la syntaxe ci-dessus applique pour philosophie d'être "prudente dans ce qu'elle envoie, et libérale dans ce qu'elle accepte". Les clients et serveurs ne devraient pas générer d'espaces blanches inutiles (caractères espace et tabulations) mais devraient accepter les espaces partout sauf dans un jeton. Lors de l'analyse des réponses, des espaces peuvent survenir n'importe où, sauf au sein d'un jeton. Précisément, toute quantité d'espaces blanches est permise au début ou à la fin d'une ligne aussi bien pour les interrogations que pour les réponses. Ceci ne s'applique pas pour les réponses qui contiennent un identifiant d'utilisateur parce que tout ce qui se trouve après les deux-points qui suivent le type de système d'exploitation jusqu'au CR/LF de terminaison est considéré faire partie de l'identifiant d'utilisateur. Le CR/LF de terminaison N'EST PAS considéré comme faisant partie de l'identifiant d'utilisateur.

2) Malgré ce qu'on vient de dire, les serveurs devraient restreindre la quantité d'espaces blanches entre les jetons qu'ils envoient à la plus petite quantité raisonnable ou utile. Les clients devraient avoir toute liberté pour interrompre une connexion si ils reçoivent 1000 caractères sans recevoir une <EOL>.

3) La limite de 512 caractères pour les identifiants d'utilisateur et la limite à 64 caractères sur les jetons devraient être comprises comme signifiant ce qui suit : a) aucun nouveau jeton (c'est-à-dire, OPSYS ou ERROR-TYPE) ne sera défini avec une longueur supérieure à 64, et b) un serveur NE DEVRAIT PAS envoyer plus de 512 octets d'identifiant d'utilisateur, et un client DOIT accepter au moins 512 octets d'identifiant d'utilisateur. À cause de cette limitation, un serveur DOIT retourner la portion de poids fort de l'identifiant d'utilisateur dans les 512 premiers octets.

4) Les jeux de caractères et leurs identifiants devraient se transposer directement en ceux définis ou référencés par la RFC 1340, "Numéros alloués" ou ses successeurs. Les identifiants de jeux de caractères ne s'appliquent qu'au champ Identification de l'usager – tous les autres champs seront définis et doivent être envoyés en US-ASCII.

5) Bien que <user-id> soit défini comme une <octet-string> ci-dessus, il doit respecter les contraintes de format et de jeu de caractères impliquées par le <opsys-field> ; voir la discussion ci-dessus.

6) Le jeu de caractères donne le contexte pour que le client imprime ou mémorise la chaîne d'identification d'usager retournée. Si le client ne reconnaît ou ne met pas en œuvre le jeu de caractères retourné, il devrait traiter la chaîne d'identification retournée comme OCTET, mais devrait de plus mémoriser ou faire rapport sur le jeu de caractères. Une chaîne OCTET devrait être imprimée, mémorisée ou traitée en notation hexadécimale (0-9a-f) en plus de toute autre représentation que le client met en œuvre – cela assure une représentation standard entre les différentes mises en œuvre.

6. Considérations sur la sécurité

Les informations retournées par ce protocole sont au plus aussi dignes de confiance que l'hôte qui les fournit OU que l'organisation qui fait fonctionner l'hôte. Par exemple, un PC dans un laboratoire ouvert a peu de contrôle, s'il en a aucun, sur lui pour empêcher un utilisateur de faire que le présent protocole retourne l'identifiant que veut l'utilisateur. De même, si l'hôte a été compromis, les informations retournées peuvent être complètement erronées et trompeuses.

Le protocole d'identification n'est pas destiné à être un protocole d'autorisation ou de contrôle d'accès. Au mieux, il apporte des informations d'audit supplémentaires à l'égard d'une connexion TCP. Au pire, il peut fournir des informations trompeuses, incorrectes, ou malveillantes.

L'utilisation des informations retournées par ce protocole pour autre chose que l'audit est fortement déconseillée. Précisément, l'utilisation des informations du protocole d'identification pour prendre des décisions de contrôle d'accès – comme méthode principale (c'est-à-dire sans autre vérification) ou comme additif à d'autres méthodes, peut résulter en un affaiblissement de la sécurité normale de l'hôte.

Un serveur d'identification peut révéler des informations sur les utilisateurs, entités, objets ou processus qui pourraient normalement être considérées comme confidentielles. Un serveur d'identification fournit un service qui est en gros analogue aux services d'identifiant de l'appelant *(CallerID)* fourni par certaines compagnies de téléphonie et beaucoup des mêmes

considérations et arguments de confidentialité qui s'appliquent au service d'identification de l'appelant s'appliquent à l'identification. Si vous voulez faire fonctionner un serveur "finger", les considérations de confidentialité peuvent s'opposer à la prise en charge de ce protocole.

7. Remerciements

Des remerciement sont adressés à Dan Bernstein qui est principalement responsable du renouvellement de l'intérêt pour ce protocole et qui a relevé des erreurs gênantes dans la RFC 931.

Références

[1] [RFC0931] M. St Johns, "Serveur d'authentification", septembre 1984. *(Rendue obsolète par la* RFC1413*)*

[2] [RFC1340] J. Reynolds et J. Postel, "Numéros alloués", STD 2, juillet 1992. *(Rendue obsolète par la RFC 1700, elle-même Historique) voir www.iana.org*)

Adresse de l'auteur

Michael C. St. Johns
DARPA/CSTO
3701 N. Fairfax Dr
Arlington, VA 22203
USA
mél : stjohns@DARPA.MIL