Groupe de travail Réseau
Request for Comments : 4819
Catégorie : Sur la voie de la normalisation
Traduction Claude Brière de L'Isle

J. Galbraith, VanDyke Software
J. Van Dyke, VanDyke Software
J. Bright, Silicon Circus
mars 2007

Sous système de clé publique Secure Shell

Résumé
Secure Shell définit un mécanisme d'authentification d'utilisateur qui se fonde sur les clés publiques, mais il ne définit aucun mécanisme pour la distribution des clés. Aucune solution commune de gestion de clé n'existe dans les mises en œuvre actuelles. Le présent document décrit un protocole qui peut être utilisé pour configurer des clés publiques d'une façon indépendante de la mise en œuvre, permettant au logiciel client de prendre la charge de cette configuration.

Le sous système de clé publique fournit un mécanisme indépendant du serveur pour que les clients ajoutent des clés publiques, suppriment des clés publiques, et fassent la liste des clés publiques actuellement connues du serveur. Les droits de gestion des clés publiques sont spécifiques et limités à l'utilisateur authentifié.

Une clé publique peut aussi être associée à diverses restrictions, incluant une commande ou sous système obligatoire.

Table des matières

1. Introduction

Secure Shell (SSH) est un protocole pour sécuriser une connexion à distance et d'autres services sûrs de réseau sur un réseau non sûr. Secure Shell définit un mécanisme d'authentification d'utilisateur qui se fonde sur les clés publiques, mais ne définit aucun mécanisme pour la distribution de clé. La pratique courante est de s'authentifier une fois avec un mot de passe d'authentification et de transférer la clé publique au serveur. Cependant, à ce jour il n'y a pas deux mises en œuvre qui utilisent le même mécanisme pour configurer la clé publique à utiliser.

Le présent document décrit un sous système qui peut être utilisé pour configurer des clés publiques d'une façon indépendante de la mise en œuvre. Cette approche permet au logiciel client de prendre la charge de cette configuration. Le protocole de sous système de clé publique est conçu pour une extrême simplicité de mise en œuvre. Il n'est pas conçu comme un remplacement de l'infrastructure de clé publique pour les certificats X.509 (PKIX, *Public Key Infrastructure for X.509 Certificates*).

Le sous système de clé publique Secure Shell a été conçu pour fonctionner par dessus la couche de transport Secure Shell [RFC4253] et les protocoles d'authentification d'utilisateur [RFC4252]. Il fournit un mécanisme simple pour que le client gère les clés publiques sur le serveur.

Le présent document ne devrait être lu qu'après les document d'architecture de Secure Shell [RFC4251] et de connexion Secure Shell [RFC4254].

Ce protocole est destiné à être utilisé à partir du protocole de connexion Secure Shell [RFC4254] comme un sous système, comme décrit au paragraphe 6.5 "Commencer une coquille ou une commande". Le nom de sous système utilisé avec ce protocole est "publickey".

Ce protocole exige que l'utilisateur soit capable de s'authentifier d'une certaine façon avant qu'il puisse être utilisé. Si l'authentification par mot de passe est utilisée, les serveurs DEVRAIENT fournir une option de configuration pour désactiver l'utilisation de l'authentification après l'ajout de la première clé publique.

2. Terminologie

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].

3. Vue d'ensemble du sous système de clé publique

Le sous système de clé publique fournit un mécanisme indépendant du serveur pour que les clients ajoutent des clés publiques, suppriment des clés publiques, et fassent la liste des clés publiques courantes connues du serveur. Le nom du sous système est "publickey".

Les clés publiques ajoutées, supprimées, et listée en utilisant ce protocole sont spécifiques et limitées à celles de l'utilisateur authentifié.

Les opérations d'ajout, de suppression, et de liste des clés publiques de l'utilisateur authentifié sont effectuées comme des paquets de demande envoyés au serveur. Le serveur envoie des paquets de réponse qui indiquent le succès ou l'échec, ainsi que les données spécifiques de réponse.

Le format des éléments de clé publique est détaillé au paragraphe 6.6, "Algorithmes de clé publique" du document du protocole de transport SSH [RFC4253].

3.1 Ouverture du sous système de clé publique

Le sous système de clé publique est lancé par un client en envoyant une SSH_MSG_CHANNEL_REQUEST *(demande de canal de message SSH)* sur un canal d'une session existante.

Les détails de la façon d'ouvrir une session sont décrits au paragraphe 6.1 "Ouverture d'une session" dans le document du protocole de connexion SSH [RFC4254].

Pour ouvrir le sous système de clé publique, le client envoie :

octet : SSH_MSG_CHANNEL_REQUEST
uint32 : canal receveur
chaîne : "subsystem"
booléen : veut une réponse
chaîne : "publickey"

Les mises en œuvre de client DEVRAIENT rejeter cette demande ; elle est normalement seulement envoyée par le client.

Si "veut une réponse" est VRAI, le serveur DOIT répondre avec SSH_MSG_CHANNEL_SUCCESS si le sous système de clé publique a réussi à démarrer, ou SSH_MSG_CHANNEL_FAILURE si le serveur a échoué à démarrer ou ne prend pas en charge le sous système de clé publique.

Le serveur DEVRAIT répondre par SSH_MSG_CHANNEL_FAILURE si l'utilisateur n'est pas autorisé à accéder au sous système de clé publique (par exemple, parce que l'utilisateur s'est authentifié avec une clé publique interdite).

Il est RECOMMANDÉ que les clients demandent et vérifient la réponse à cette demande.

3.2 Demandes et réponses

Toutes les demandes et réponse du sous système de clé publique sont envoyées sous la forme suivante :

uint32 : longueur
chaîne : nom
... les données spécifiques de demande/réponse suivent

Le champ Longueur décrit la longueur du champ Nom et des données spécifiques de demande/réponse, mais n'inclut pas la longueur du champ Longueur lui-même. Le client DOIT recevoir un accusé de réception de chaque demande avant d'envoyer une nouvelle demande.

Le paquet Version, ainsi que toutes les demandes et réponses décrites à la Section 4, sont une description du champ 'nom' et de la partie données du paquet.

3.3 Message d'état

Il est accusé réception d'une demande par l'envoi d'un paquet d'état. Si il y a des données dans la réponse à la demande, le paquet d'état est envoyé après que toutes les données ont été envoyées.

chaîne : "status"
uint32 : code d'état
chaîne : description [RFC3629]
chaîne : étiquette de langue [RFC4646]

Un message d'état DOIT être envoyé pour tous les paquets non reconnus, et la demande NE DEVRAIT PAS clore le sous système.

3.3.1 Codes d'état

Le code d'état donne l'état dans un format plus lisible par la machine (convenant pour la localisation) et peut avoir une des valeurs suivantes :

0 SSH_PUBLICKEY_SUCCESS *(réussite de la clé publique SSH)*
1 SSH_PUBLICKEY_ACCESS_DENIED *(accès refusé à la clé publique SSH)*
2 SSH_PUBLICKEY_STORAGE_EXCEEDED *(mémorisation de clé publique SSH excédée)*
3 SSH_PUBLICKEY_VERSION_NOT_SUPPORTED *(version de clé publique SSH non prise en charge)*

4 SSH_PUBLICKEY_KEY_NOT_FOUND *(clé publique SSH non trouvée)*
5 SSH_PUBLICKEY_KEY_NOT_SUPPORTED *(clé publique SSH non prise en charge)*
6 SSH_PUBLICKEY_KEY_ALREADY_PRESENT *(clé publique SSH déjà présente)*
7 SSH_PUBLICKEY_GENERAL_FAILURE *(échec général de clé publique SSH)*
8 SSH_PUBLICKEY_REQUEST_NOT_SUPPORTED *(demande de clé publique SSH non prise en charge)*
9 SSH_PUBLICKEY_ATTRIBUTE_NOT_SUPPORTED *(attribut de clé publique SSH non pris en charge)*

Si une demande est réalisée avec succès, le serveur DOIT envoyer le code d'état SSH_PUBLICKEY_SUCCESS. La signification des codes d'échec est impliquée par leur nom.

3.4. Paquet Version

Les deux côtés DOIVENT débuter une connexion par l'envoi d'un paquet Version qui indique la version de protocole qu'ils utilisent.

chaîne : "version"
uint32 : numéro de version du protocole

Le présent document décrit la version 2 du protocole. La version 1 était utilisée par un projet antérieur de ce document. Le numéro de version a été incrémenté après des changements dans le traitement des paquets d'état.

Les deux côtés envoient le plus haut numéro de version qu'ils mettent en œuvre. Le plus faible des numéros de version est la version de protocole à utiliser. Si l'un ou l'autre côté ne prend pas en charge la version inférieure, il devrait clore le sous système et le notifier à l'autre côté en envoyant un message SSH_MSG_CHANNEL_CLOSE *(clôture du canal de messages SSH)*. Avant de clore le sous système, un message d'état avec l'état SSH_PUBLICKEY_VERSION_NOT_SUPPORTED DEVRAIT être envoyé. Noter que, normalement, les messages d'état ne sont envoyés que par le serveur (en réponse aux demandes du client). C'est la seule occasion où le client envoie un message d'état.

Les deux côtés DOIVENT attendre de recevoir cette version avant de continuer. Le paquet "version" NE DOIT PAS être envoyé à nouveau après cet échange initial. Le code d'état SSH_PUBLICKEY_VERSION_NOT_SUPPORTED ne doit pas être envoyé en réponse à toute autre demande.

Les mises en œuvre PEUVENT utiliser le 15 premiers octets du paquet Version comme "mouchard magique" pour éviter de traiter des résultats parasites de la coquille de l'utilisateur (comme décrit au paragraphe 6.5 de la [RFC4254]). Ces octets vont toujours être :

0x00 0x00 0x00 0x0F 0x00 0x00 0x00 0x07 0x76 0x65 0x72 0x73 0x69 0x6F 0x6E

4. Opérations du sous système de clé publique

Le sous système de clé publique définit actuellement quatre opérations : add, remove, list, et listattributes.

4.1 Ajout d'une clé publique

Si le client souhaite ajouter une clé publique, il envoie :

chaîne : "add"
chaîne : nom d'algorithme de clé publique
chaîne : figure de clé publique
booléen : outrepasser
uint32 : compte d'attribut
chaîne : nom d'attribut
chaîne : valeur d'attribut
bool : critique
nombre de répétitions de compte d'attribut

Le serveur DOIT tenter de mémoriser la clé publique pour l'utilisateur dans la localisation appropriée afin que la clé

publique puisse être utilisée pour les authentifications suivantes de clé publique . Si le champ Outrepasser est réglé à faux et si la clé spécifiée existe déjà, le serveur DOIT retourner SSH_PUBLICKEY_KEY_ALREADY_PRESENT. Si le serveur retourne cela, le client DEVRAIT donner à l'utilisateur l'option d'outrepasser la clé. Si le champ Outrepasser est réglé à vrai et si la clé spécifiée existe déjà, mais ne peut pas être outrepassée, le serveur DOIT retourner SSH_PUBLICKEY_ACCESS_DENIED.

Les noms des attributs sont définis suivant le même schéma qu'exposé pour les noms d'algorithmes dans la [RFC4251]. Si le serveur ne met pas en œuvre un attribut critique, il DOIT faire échouer l'ajout, avec le code d'état SSH_PUBLICKEY_ATTRIBUTE_NOT_SUPPORTED. Pour les besoins d'un attribut critique, la simple mémorisation de l'attribut n'est pas suffisante, le serveur doit plutôt comprendre et mettre en œuvre l'intention de l'attribut.

Les attributs suivants sont actuellement définis :

"comment" *(commentaire)* : la valeur de l'attribut "comment" contient du texte spécifié par l'utilisateur sur la clé publique. Le serveur DEVRAIT faire tous ses efforts pour préserver cette valeur et la retourner avec la clé durant toute opération de liste suivante. Le serveur NE DOIT PAS tenter d'interpréter ou agir d'aucune façon sur le contenu du champ "comment". L'attribut "comment" doit être spécifié en format UTF-8 [RFC3629]. Le champ Comment est utile pour que l'utilisateur puisse identifier la clé sans avoir à comparer son empreinte digitale. Cet attribut NE DEVRAIT PAS être critique.

"comment-language" *(langage du commentaire)* : si cet attribut est spécifié, il DOIT immédiatement suivre un attribut "comment" et spécifier le langage pour cet attribut [RFC4646]. Le client PEUT spécifier plus d'un commentaire si il spécifie un langage différent pour chacun de ces commentaires. Le serveur DEVRAIT tenter de mémoriser chaque commentaire avec son attribut de langage. Cet attribut NE DEVRAIT PAS être critique.

"command-override" *(outrepasser la commande)* : "command-override" spécifie une commande à exécuter quand cette clé est utilisée. La commande devrait être exécutée par le serveur quand il reçoit une demande "exec" ou "shell" provenant du client, à la place de la commande ou coquille qui aurait autrement été exécutée par suite de cette demande. Si la chaîne de commande est vide, les deux demandes "exec" et "shell" devraient être refusées. Si aucun attribut "command-override" n'est spécifié, toutes les demandes "exec" et "shell" devraient être permises (pour autant qu'elles satisfassent aux autres vérification de sécurité ou d'autorisation que le serveur peut effectuer). Cet attribut DEVRAIT être critique.

"subsystem" : "subsystem" spécifie une liste séparée par des virgules des sous systèmes qui peuvent être démarrés (en utilisant une demande "subsystem") quand cette clé est utilisée. Cet attribut DEVRAIT être critique. Si la valeur est vide, aucun sous système ne peut démarrer. Si l'attribut "subsystem" n'est pas spécifié, aucune restriction n'est mise aux sous systèmes qui peuvent être démarrés lors de l'authentification en utilisant cette clé.

"x11" : "x11" spécifie que la transmission X11 ne peut pas être effectuée quand cette clé est utilisée. Le champ Valeur d'attribut DEVRAIT être vide pour cet attribut. Cet attribut DEVRAIT être critique.

"shell" : "shell" spécifie que les demandes de canal de session "shell" devraient être refusées quand cette clé est utilisée. Le champ Valeur d'attribut DEVRAIT être vide pour cet attribut. Cet attribut DEVRAIT être critique.

"exec" : "exec" spécifie que les demandes de canal de session "exec" devraient être refusées quand cette clé est utilisée. Le c0amp attribute-value DEVRAIT être vide pour cet attribut. Cet attribut DEVRAIT être critique.

"agent" : "agent" spécifie que les demandes de canal de session "auth-agent-req" devraient être refusées quand cette clé est utilisée. Le champ Valeur d'attribut DEVRAIT être vide pour cet attribut. Cet attribut DEVRAIT être critique.

"env" : "env" spécifie que les demandes de canal de session "env" devraient être refusées quand cette clé est utilisée. Le champ Valeur d'attribut DEVRAIT être vide pour cet attribut. Cet attribut DEVRAIT être critique.

"from" : "from" spécifie une liste séparée par des virgules des hôtes desquels la clé peut être utilisée. Si un hôte qui n'est pas dans la liste tente d'utiliser cette clé à des fins d'autorisation, la tentative d'autorisation DOIT être refusée. Le serveur DEVRAIT faire une entrée du journal d'incidents pour cela. Le serveur PEUT fournir une méthode pour que les administrateurs interdisent l'apparition d'un hôte dans cette liste. Le serveur devrait utiliser toute méthode appropriée pour sa plate-forme pour identifier l'hôte – par exemple, pour les réseaux fondés sur IP, de vérifier l'adresse IP ou effectuer une recherche inverse du DNS. Pour les réseaux fondés sur IP, il est prévu que chaque élément du paramètre "from" prendra la forme d'une adresse IP ou nom d'hôte spécifique.

"port-forward" : "port-forward" spécifie qu'aucune demande "direct-tcpip" ne devrait être acceptée, sauf pour les hôtes spécifiés dans la liste séparée par des virgules fournie comme valeur de cet attribut. Si la valeur de cet attribut est vide, toutes les demandes "direct-tcpip" devraient être refusées quand on utilise cette clé. Cet attribut DEVRAIT être critique.

"reverse-forward" : "reverse-forward" spécifie qu'aucune demande "tcpip-forward" ne devrait être acceptée, sauf pour les numéros d'accès dans la liste séparée par des virgules fournie comme valeur de cet attribut. Si la valeur de cet attribut est vide, toutes les demandes "tcpip-forward" devraient être refusées quand on utilise cette clé. Cet attribut DEVRAIT être critique.

En plus des attributs spécifiés par le client, le serveur PEUT fournir une méthode pour que les administrateurs appliquent certains attributs de façon obligatoire.

4.2 Suppression d'une clé publique

Si le client souhaite supprimer une clé publique, il envoie :

chaîne : "remove"
chaîne : nom de l'algorithme de clé publique
chaîne ; figure de clé publique

Le serveur DOIT tenter de supprimer la clé publique pour l'utilisateur à partir de la localisation appropriée, afin que cette clé publique ne puisse pas être utilisée pour des authentifications suivantes.

4.3 Liste des clés publiques

Si le client souhaite faire la liste des clés publiques connues, il envoie :

chaîne : "list"

Le serveur va répondre avec zéro, une, ou plusieurs des réponses suivantes :

chaîne : "publickey"
chaîne : nom d'algorithme de clé publique
chaîne : figure de clé publique
uint32 : compte d'attributs
chaîne : nom d'attribut
chaîne : valeur d'attribut
nombre de répétitions de compte d'attribut

Il n'est pas exigé que les réponses soient dans un ordre particulier. Bien que certaines mises en œuvre de serveur puissent envoyer les réponses dans un certain ordre, les mises en œuvre de client ne devraient pas compter que les réponses soient dans un certain ordre.

À la suite de la dernière réponse "publickey", un paquet d'état DOIT être envoyé. Les mises en œuvre DEVRAIENT accepter cette demande.

4.4 Liste des capacités de serveur

Si le client souhaite savoir quels attributs de clé le serveur prend en charge, il envoie :

chaîne : "listattributes"

Le serveur va répondre avec zéro, une ou plusieurs des réponses suivantes :

chaîne : "attribute"
chaîne ; nom d'attribut
booléen : obligatoire

Le champ "obligatoire" indique si cet attribut va être obligatoirement appliqué à toute clé ajoutée (sans considération de si l'attribut a été spécifié par le client) du fait de réglages administratifs sur le serveur. Si le serveur ne prend pas en charge les réglages administratifs de cette nature, il DOIT retourner "faux" dans le champ "obligatoire". Un exemple de l'utilisation de l'attribut "obligatoire" serait celui d'un serveur avec un fichier de configuration qui spécifie que l'utilisateur n'est pas autorisé à l'accès à la coquille. Cela étant, le serveur va retourner l'attribut "shell", avec "obligatoire" marqué vrai. Quels que soient les attributs que l'utilisateur demande ensuite au serveur d'appliquer à sa clé, le serveur va aussi appliquer l'attribut "shell", rendant impossible à l'utilisateur l'utilisation d'une coquille.

À la suite de la dernière réponse "attribute", un paquet d'état DOIT être envoyé. Une mise en œuvre PEUT choisir de ne pas accepter cette demande.

5. Considérations sur la sécurité

Ce protocole suppose qu'il fonctionne sur un canal sûr et que les points d'extrémité du canal ont été authentifiés. Donc, ce protocole suppose qu'il est protégé en externe contre les attaques de niveau réseau.

Ce protocole fournit un mécanisme qui permet que les données d'authentification de client soient téléchargées et manipulées. Il est de la responsabilité de la mise en œuvre de serveur d'appliquer tous les contrôles d'accès qui peuvent être exigés pour limiter l'accès permis à tout utilisateur particulier (l'utilisateur étant authentifié à l'extérieur de ce protocole, normalement en utilisant le protocole d'authentification d'utilisateur SSH [RFC4252]). En particulier, il est possible aux utilisateurs d'outrepasser une clé existante sur le serveur avec ce protocole, en même temps tout en spécifiant moins de restrictions pour la nouvelle clé qu'il n'en étaient présentes précédemment. Les serveurs devraient faire attention à ce que quand ils font cela, les clients ne sont pas capables d'outrepasser les pré-réglages de l'administrateur du serveur.

Ce protocole exige que le client suppose que le serveur va mettre en œuvre correctement et respecter les attributs appliqués aux clés. Des erreurs de mise en œuvre dans le serveur pourraient être cause que les clients autorisent des clés pour un accès qu'ils n'étaient pas destinés à avoir, ou à appliquer moins de restrictions que ce qui était prévu.

6. Considérations relatives à l'IANA

Cette Section contient les conventions utilisées dans la désignation des espaces de noms, l'état initial du registre, et des instructions pour les futures allocations.

6.1 Enregistrements

Conformément au paragraphe 4.9.5 de la [RFC4250], le présent document fait les enregistrements suivants :

Le nom de sous système "publickey".

6.2 Noms

Dans les paragraphes qui suivent, les valeurs des espaces de noms sont textuelles. Les conventions et instructions à l'IANA pour les futures allocations sont données dans cette section. Les allocations initiales sont données dans leurs paragraphes respectifs.

6.2.1 Conventions pour les noms

Tous les noms enregistrés par l'IANA dans les paragraphes qui suivent DOIVENT être des chaînes de caractères US-ASCII imprimables, et NE DOIVENT PAS contenir les caractères arobase ("@"), virgule (","), ni d'espace ou de caractères de contrôle (codes ASCII inférieurs ou égaux à 32). Les noms sont sensibles à la casse, et NE DOIVENT PAS faire plus de 64 caractères.

Une disposition est prise ici pour les noms localement extensibles. L'IANA n'enregistrera et ne contrôlera pas les noms qui contiennent un arobase. Les noms qui contiennent le signe arobase vont avoir le format de "nom@nomdedomaine" (sans les guillemets) où la partie précédant l'arobase est le nom. Le format de la partie précédant l'arobase n'est pas spécifié ;

cependant, ces noms DOIVENT être des chaînes US-ASCII imprimables, et NE DOIVENT PAS contenir le caractère virgule (",") ou d'espace, ou de caractères de contrôle (codes ASCII inférieurs ou égaux à 32). La partie qui suit le signe arobase DOIT être un nom de domaine Internet valide, pleinement qualifié [RFC1034] contrôlé par la personne ou organisation qui définit le nom. Les noms sont sensibles à la casse, et NE DOIVENT PAS faire plus de 64 caractères. Il appartient à chaque domaine de gérer son espace de noms local. Il a été noté que ces noms ressemblent aux adresses de messagerie électronique de la [RFC0822] STD 11. C'est une pure coïncidence et n'a en fait rien à voir avec la STD 11 [RFC0822]. Un exemple de nom défini localement est "notre-attribut@exemple.com" (sans les guillemets).

6.2.2 Futures allocations de noms

Les demandes d'allocation de nouveaux noms DOIVENT être faites par la méthode de consensus de l'IETF décrite dans la [RFC2434].

6.3 Noms de demande de sous système de clé publique

Le tableau suivant donne la liste des allocations initiales de noms de demande de sous système de clé publique.

Nom de demande
- version
- add
- remove
- list
- listattributes

6.4. Noms de réponse de sous système de clé publique

Le tableau suivant donne la liste des allocations initiales de noms de réponse de sous système de clé publique.

Nom de réponse
- version
- status
- publickey
- attribute

6.5 Noms d'attribut de sous système de clé publique

Les attributs sont utilisés pour définir les propriétés ou restrictions sur les clés publiques. Le tableau suivant donne la liste des allocations initiales de noms de noms d'attribut de sous système de clé publique.

Nom d'attribut
- comment
- comment-language
- command-override
- subsystem
- x11
- shell
- exec
- agent
- env
- from
- port-forward
- reverse-forward

6.6 Codes d'état de sous système de clé publique

Le code d'état est une valeur d'octet, qui décrit l'état d'une demande.

6.6.1 Conventions

Les réponses d'état ont des codes d'état dans la gamme 0 à 255. Ces numéros sont alloués comme suit. Parmi eux, la gamme 192 à 255 est réservée à l'utilisation par des extensions locales privées.

6.6.2 Allocations initiales

Le tableau suivant identifie les allocations initiales des valeurs de code d'état de sous système de clé publique.

Code d'état	Valeur	Référence
SSH_PUBLICKEY_SUCCESS	0	RFC4819
SSH_PUBLICKEY_ACCESS_DENIED	1	RFC4819
SSH_PUBLICKEY_STORAGE_EXCEEDED	2	RFC4819
SSH_PUBLICKEY_VERSION_NOT_SUPPORTED	3	RFC4819
SSH_PUBLICKEY_KEY_NOT_FOUND	4	RFC4819
SSH_PUBLICKEY_KEY_NOT_SUPPORTED	5	RFC4819
SSH_PUBLICKEY_KEY_ALREADY_PRESENT	6	RFC4819
SSH_PUBLICKEY_GENERAL_FAILURE	7	RFC4819
SSH_PUBLICKEY_REQUEST_NOT_SUPPORTED	8	RFC4819
SSH_PUBLICKEY_ATTRIBUTE_NOT_SUPPORTED	9	RFC4819

6.6.3 Futures allocations

Les demandes d'allocation de nouveaux codes d'état dans la gamme de 0 à 191 DOIVENT être faites par la méthode d'action de normalisation décrite dans la [RFC2434].

L'IANA ne contrôlera pas la gamme de codes d'état de 192 à 255. Cette gamme est pour utilisation privée.

10. Références

10.1 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)

[RFC3629] F. Yergeau, "UTF-8, un format de transformation de la norme ISO 10646", STD 63, novembre 2003.

[RFC4251] T. Ylonen et C. Lonvick, "Architecture du protocole Secure Shell (SSH)", janvier 2006. *(P.S. ; MàJ par* RFC8308)

[RFC4252] T. Ylonen et C. Lonvick, éd., "Protocole d'authentification Secure Shell (SSH)", janvier 2006. *(P.S .; MàJ par* RFC8308, 8332)

[RFC4253] C. Lonvick, "Protocole de couche Transport Secure Shell (SSH)", janvier 2006. *(P.S., MàJ par* RFC6668, 8268, 8308, 8332, 8709)

[RFC4254] T. Ylonen et C. Lonvick, éd., "Protocole de connexion Secure Shell (SSH)", janvier 2006. *(P.S. ; MàJ par* RFC8308)

[RFC4646] A. Phillips, M. Davis, "Étiquettes d'identification des langues", BCP0047 septembre 2006. *(Remplacée par* RFC5646)

7.2 Références pour information

[RFC0822] D. Crocker, "Norme pour le format des messages de texte de l'ARPA-Internet", STD 11, août 1982. (*Obsolète, voir RFC*5322)

[RFC1034] P. Mockapetris, "Noms de domaines - Concepts et facilités", STD 13, novembre 1987. (*MàJ par* RFC1101, 1183, 1348, 1876, 1982, 2065, 2181, 2308, 2535, 4033, 4034, 4035, 4343, 4035, 4592, 5936, 8020, 8482, 8767)

[RFC2434] T. Narten et H. Alvestrand, "Lignes directrices pour la rédaction d'une section Considérations relatives à l'IANA dans les RFC", BCP 26, octobre 1998. *(Rendue obsolète par la* RFC5226)

[RFC4250] S. Lehtinen et C. Lonvick, éd., "Numéros alloués du protocole Secure Shell (SSH)", janvier 2006. *(P.S. ; MàJ par* RFC8268)

8. Remerciements

Brent McClure a contribué à la rédaction de ce document.

Adresse des auteurs

Joseph Galbraith
VanDyke Software
4848 Tramway Ridge Blvd
Suite 101
Albuquerque, NM 87111
US
téléphone : +1 505 332 5700
mél : galb@vandyke.com

Jeff P. Van Dyke
VanDyke Software
4848 Tramway Ridge Blvd
Suite 101
Albuquerque, NM 87111
US
téléphone : +1 505 332 5700
mél : jpv@vandyke.com

Jon Bright
Silicon Circus
24 Jubilee Road
Chichester, West Sussex PO19 7XB
UK
téléphone : +49 172 524 0521
mél : jon@siliconcircus.com

Déclaration complète de droits de reproduction

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