Groupe de travail Réseau
Request for Comments : 4252
Catégorie : Sur la voie de la normalisation
Traduction Claude Brière de L'Isle

T. Ylonen, SSH Communications Security Corp
C. Lonvick, éd. ,Cisco Systems, Inc.

janvier 2006

Protocole d'authentification Secure Shell (SSH)

Résumé
Secure Shell (SSH) est un protocole pour la connexion à distance sécurisée et autres services réseau sécurisés sur un réseau non sûr. Le présent document décrit le cadre du protocole d'authentification SSH et les méthodes de clé publique, de mot de passe et d'authentification du client fondées sur l'hôte. Des méthodes d'authentification supplémentaires sont décrites dans des documents séparés. Le protocole d'authentification SSH fonctionne par dessus le protocole de couche transport SSH et fournit un seul tunnel authentifié pour le protocole de connexion SSH.

Table des matières

1. Introduction

Le protocole d'authentification SSH est un protocole générique d'authentification d'utilisateur. Il est destiné à fonctionner sur le protocole de couche transport SSH [RFC4253]. Ce protocole suppose que les protocoles sous-jacents assurent la protection de l'intégrité et de la confidentialité.

Le présent document ne devrait être lu qu'après la lecture du document d'architecture SSH [RFC4251]. Le présent document utilise librement la terminologie et la notation provenant du document d'architecture sans référence ou autre explication.

Le 'nom de service' pour ce protocole est "ssh-userauth".

Quand ce protocole commence, il reçoit l'identifiant de session du protocole de niveau inférieur (c'est l'échange de hachage H provenant du premier échange de clés). L'identifiant de session identifie cette session de façon univoque et convient pour signer afin de prouver la possession d'une clé privée. Ce protocole a aussi besoin de savoir si le protocole de niveau inférieur assure la protection de la confidentialité.

2. Contributeurs

Les contributeurs majeurs originaux de cet ensemble de documents ont été Tatu Ylonen, Tero Kivinen, Timo J. Rinne, Sami Lehtinen (tous de SSH Communications Security Corp) et Markku-Juhani O. Saarinen (Université de Jyvaskyla). Darren Moffat était l'éditeur original de cet ensemble de documents et y a aussi fait de très substantielles contributions.

De nombreuses personnes ont contribué au développement de ce document au fil des ans. Les personnes qui doivent en être remerciées incluent Mats Andersson, Ben Harris, Bill Sommerfeld, Brent McClure, Niels Moller, Damien Miller, Derek Fawcus, Frank Cusack, Heikki Nousiainen, Jakob Schlyter, Jeff Van Dyke, Jeffrey Altman, Jeffrey Hutzelman, Jon Bright, Joseph Galbraith, Ken Hornstein, Markus Friedl, Martin Forssen, Nicolas Williams, Niels Provos, Perry Metzger, Peter Gutmann, Simon Josefsson, Simon Tatham, Wei Dai, Denis Bider, der Mouse, et Tadayoshi Kohno. La présence de leur noms ici ne signifie pas qu'ils approuvent le présent document, mais qu'il y ont contribué.

3. Conventions utilisées dans ce document

Tous les documents relatifs aux protocoles SSH devront utiliser 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 qui sont à interpréter comme décrit dans le BCP 14, [RFC2119].

Les mots-clés "UTILISATION PRIVÉE", "ALLOCATION HIÉRARCHIQUE ", "PREMIER ARRIVÉ, PREMIER SERVI", "REVUE D'EXPERT", "SPÉCIFICATION EXIGÉE", "APPROBATION DE L'IESG", "CONSENSUS DE L'IETF", et "ACTION DE NORMALISATION" qui apparaissent dans le présent document lorsque ils sont utilisés pour décrire l'allocation d'espace de noms sont à interpréter comme décrit dans la [RFC2434].

Les champs de protocole et les valeurs possibles pour les remplir sont définis dans cet ensemble de documents. Les champs de protocole seront définis dans les définitions de messages. Par exemple, SSH_MSG_CHANNEL_DATA est défini comme suit :

octet : SSH_MSG_CHANNEL_DATA
uint32 : canal receveur
chaîne : données

Tout au long de ces documents, quand les champs sont référencés, ils vont apparaître avec des guillemets simples. Quand des valeurs pour remplir ces champs sont référencées, elles vont apparaître avec des guillemets doubles. En utilisant l'exemple ci-dessus, les valeurs possibles pour 'données' sont "foo" et "bar".

4. Cadre du protocole d'authentification

Le serveur pilote l'authentification en disant au client quelles méthodes d'authentification peuvent être utilisées pour continuer l'échange à tout moment. Le client est libre d'essayer les méthodes énumérées par le serveur dans tout ordre. Cela donne au serveur le contrôle complet du processus d'authentification si désiré, mais donne aussi assez de souplesse au client pour utiliser les méthodes qu'il prend en charge ou qui sont les plus convenables à l'utilisateur, quand plusieurs méthodes sont offertes par le serveur.

Les méthodes d'authentification sont identifiées par leur nom, comme défini dans la [RFC4251]. La méthode "none" est réservée, et NE DOIT PAS figurer sur la liste comme prise en charge. Cependant, elle PEUT être envoyée par le client. Le serveur DOIT toujours rejeter cette demande, sauf si le client va avoir l'accès sans aucune authentification, auquel cas le serveur DOIT accepter cette demande. Le principal objet de l'envoi de cette demande est d'obtenir la liste des méthodes acceptées par le serveur.

Le serveur DEVRAIT avoir une temporisation pour l'authentification et déconnecter si l'authentification n'a pas été acceptée durant la période de temporisation. La période RECOMMANDÉE de temporisation est de 10 minutes. De plus, la mise en œuvre DEVRAIT limiter le nombre d'échecs de tentatives d'authentification qu'un client peut effectuer dans une seule session (la limite RECOMMANDÉE est de 20 tentatives). Si le seuil est excédé, le serveur DEVRAIT déconnecter.

On trouvera d'autres réflexions sur les temporisations et les essais d'authentification dans [ssh-1.2.30].

5. Demandes d'authentification

Toutes les demandes d'authentification DOIVENT utiliser le format de message suivant. Seuls les premiers champs sont définis ; les champs restants dépendent de la méthode d'authentification.

octet : SSH_MSG_USERAUTH_REQUEST *(message SSH de demande d'authentification d'utilisateur)*
chaîne : nom d'utilisateur en codage ISO-10646 UTF-8 [RFC3629]
chaîne : nom de service en US-ASCII
chaîne : nom de méthode en US-ASCII
.... : champs spécifiques de la méthode

Le 'nom d'utilisateur' et le 'nom de service' sont répétés dans chaque nouvelle tentative d'authentification, et PEUVENT changer. La mise en œuvre de serveur DOIT les vérifier attentivement dans chaque message, et DOIT purger tous les états d'authentification accumulés si ils changent. Si elle est dans l'incapacité de purger un état d'authentification, elle DOIT se déconnecter si le 'nom d'utilisateur' ou le 'nom de service' change.

Le 'nom de service' spécifie le service à commencer après l'authentification. Plusieurs services authentifiés différents peuvent être fournis. Si le service demandé n'est pas disponible, le serveur PEUT déconnecter immédiatement ou à tout moment ultérieur. L'envoi d'un message de déconnexion approprié est RECOMMANDÉ. En tout cas, si le service n'existe pas, l'authentification NE DOIT PAS être acceptée.

Si le 'nom d'utilisateur' demandé n'existe pas, le serveur PEUT déconnecter, ou PEUT envoyer une fausse liste de valeurs de 'noms de méthodes d'authentification' acceptables, mais n'en accepter jamais aucune. Cela rend possible au serveur d'éviter de divulguer des informations sur les comptes qui existent. Dans tous les cas, si le 'nom d'utilisateur' n'existe pas, la demande d'authentification NE DOIT PAS être acceptée.

Bien que généralement il y ait peu de cas où les clients envoient des demandes que le serveur n'a pas mentionnées comme acceptables, l'envoi de telles demandes n'est pas une erreur, et le serveur DEVRAIT simplement rejeter les demandes qu'il ne reconnaît pas.

Une demande d'authentification PEUT résulter en d'autres échanges de messages. Tous ces messages dépendent du 'nom de méthode' d'authentification utilisé, et le client PEUT à tout moment continuer avec un nouveau message SSH_MSG_USERAUTH_REQUEST, et dans ce cas le serveur DOIT abandonner la tentative précédente d'authentification et continuer avec la nouvelle.

Les valeurs de 'nom de méthode' suivantes sont définies :
"publickey" : EXIGÉ
"password" : FACULTATIF
"hostbased" : FACULTATIF
"none" : NON RECOMMANDÉ

Des valeurs supplémentaires de 'nom de méthode' peuvent être définies comme spécifié dans les [RFC4250] et [RFC4251].

5.1 Réponses aux demandes d'authentification

Si le serveur rejette la demande d'authentification, il DOIT répondre avec ce qui suit :
octet : SSH_MSG_USERAUTH_FAILURE *(message SSH d'échec d'authentification d'utilisateur)*
liste de noms : authentifications qui peuvent continuer
booléen : succès partiel

Les 'authentifications qui peuvent continuer' est une liste de noms séparés par des virgules de valeurs de 'noms de méthodes' d'authentification qui peuvent continuer de façon productive le dialogue d'authentification.

Il est RECOMMANDÉ que les serveurs incluent dans la liste de noms seulement les valeurs de 'nom de méthode' qui sont réellement utiles. Cependant, il n'est pas illégal d'inclure des valeurs de 'nom de méthode' qui ne peuvent pas être utilisées pour authentifier l'utilisateur.

Les authentifications déjà achevées avec succès NE DEVRAIENT PAS être incluses dans la liste de noms, sauf si elles devraient être effectuées à nouveau pour une raison quelconque.

La valeur de 'succès partiel' DOIT être VRAI si la demande d'authentification à laquelle elle est une réponse était réussie. Elle DOIT être FAUX si la demande n'a pas été traitée avec succès.

Quand le serveur accepte l'authentification, il DOIT répondre avec :

octet : SSH_MSG_USERAUTH_SUCCESS *(message SSH de succès d'authentification d'utilisateur)*

Noter que ceci n'est pas envoyé après chaque étape dans une séquence d'authentification multi méthodes, mais seulement quand l'authentification est achevée.

Le client PEUT envoyer plusieurs demandes d'authentification sans attendre les réponses aux demandes précédentes. Le serveur DOIT traiter complètement chaque demande et accuser réception de tout échec de demande par un message SSH_MSG_USERAUTH_FAILURE avant de traiter la demande suivante.

Une demande qui requiert l'échange d'autres messages sera interrompue par une demande suivante. Dans ce cas, un client NE DOIT PAS envoyer une demande suivante si il n'a pas reçu de réponse du serveur pour une demande précédente. Un message SSH_MSG_USERAUTH_FAILURE NE DOIT PAS être envoyé pour une méthode interrompue.

SSH_MSG_USERAUTH_SUCCESS DOIT être envoyé une seule fois. Quand SSH_MSG_USERAUTH_SUCCESS a été envoyé, toutes les demandes authentification suivantes reçues après celle-là DEVRAIENT être ignorées en silence.

Tous les messages non d'authentification envoyés par le client après la demande qui a eu pour résultat l'envoi du SSH_MSG_USERAUTH_SUCCESS DOIVENT être passés au service qui fonctionne par dessus ce protocole. De tels messages peuvent être identifiés par leur numéro de message (voir la Section 6).

5.2 Demande d'authentification "none"

Un client peut demander une liste de valeurs de 'nom de méthode' d'authentification qui peuvent continuer en utilisant le 'nom de méthode' d'authentification "none".

Si aucune authentification n'est nécessaire pour l'utilisateur, le serveur DOIT retourner un message SSH_MSG_USERAUTH_SUCCESS. Autrement, le serveur DOIT retourner SSH_MSG_USERAUTH_FAILURE et PEUT retourner avec, dans sa valeur des 'authentifications qui peuvent continuer', une liste de méthodes qui peuvent continuer.

Ce 'nom de méthode' NE DOIT PAS être mentionné sur la liste de ceux qui sont acceptés par le serveur.

5.3 Achèvement de l'authentification d'utilisateur

L'authentification est achevée quand le serveur a répondu avec SSH_MSG_USERAUTH_SUCCESS. Tous les messages en rapport avec l'authentification reçus après l'envoi de ce message DEVRAIENT être ignorés en silence.

Après l'envoi de SSH_MSG_USERAUTH_SUCCESS, le serveur commence le service demandé.

5.4 Message bannière

Dans certaines juridictions, l'envoi d"un message d'avertissement avant l'authentification peut être pertinent pour obtenir une protection juridique. De nombreuses machines UNIX, par exemple, affichent normalement un texte provenant de /etc/issue, utilisent des enveloppes TCP, ou de logiciel similaire, pour afficher une bannière avant de produire une invite de connexion.

Le serveur SSH peut envoyer un message SSH_MSG_USERAUTH_BANNER *(message SSH de bannière d'authentification d'utilisateur)* à tout moment après le début de ce protocole d'authentification et avant la réussite de

l'authentification. Ce message contient un texte à afficher au client utilisateur avant de tenter l'authentification. Le format est le suivant :

octet : SSH_MSG_USERAUTH_BANNER
chaîne : message en codage ISO-10646 UTF-8 [RFC3629]
chaîne : étiquette de langue [RFC3066]

Par défaut, le client DEVRAIT afficher le 'message' à l'écran. Cependant, comme le 'message' va probablement être envoyé pour chaque tentative de connexion, et comme certains logiciels de client vont avoir besoin d'ouvrir une fenêtre séparée pour cet avertissement, le logiciel client peut permettre à l'utilisateur de désactiver explicitement l'affichage de bannières provenant du serveur. Le 'message' peut consister en plusieurs lignes, avec les coupures de lignes indiquées par des paires CRLF.

Si le chaîne 'message' est affichée, le filtrage des caractères de contrôle, discuté dans la [RFC4251], DEVRAIT être utilisé pour éviter des attaques par l'envoi de caractères de contrôle de terminal.

6. Numéros de message de protocole d'authentification

Tous les numéros de message utilisés par ce protocole d'authentification sont dans la gamme de 50 à 79, qui fait partie de la gamme réservée pour les protocoles qui fonctionnent par dessus le protocole SSH de couche transport.

Les numéros de message de 80 et au dessus sont réservés pour les protocoles qui fonctionnent après ce protocole d'authentification, de sorte que recevoir l'un d'eux avant l'achèvement de l'authentification est une erreur, à laquelle le serveur DOIT répondre en se déconnectant, de préférence avec l'envoi d'un message de déconnexion approprié envoyé pour faciliter la réparation de problèmes.

Après une authentification réussie, de tels messages sont passés au service de niveau supérieur.

Voici les codes des messages généraux d'authentification :

SSH_MSG_USERAUTH_REQUEST	50
SSH_MSG_USERAUTH_FAILURE	51
SSH_MSG_USERAUTH_SUCCESS	52
SSH_MSG_USERAUTH_BANNER	53

En plus de ceux-ci, il y a une gamme de numéros de message (60 à 79) réservée pour les messages spécifiques d'une méthode. Ces messages ne sont envoyés que par le serveur (le client envoie seulement des messages SSH_MSG_USERAUTH_REQUEST). Des méthodes d'authentification différentes réutilisent les mêmes numéros de message.

7. Méthode d'authentification de clé publique "publickey"

Le seul 'nom de méthode' d'authentification EXIGÉ est l'authentification "publickey". Toutes les mises en œuvre DOIVENT prendre en charge cette méthode ; cependant, tous les utilisateurs n'ont pas besoin d'avoir des clés publiques, et la plupart des politiques locales ne vont probablement pas dans un futur proche exiger d'authentification par clé publique pour tous les utilisateurs.

Avec cette méthode, la possession d'une clé privée sert d'authentification. Cette méthode fonctionne en envoyant une signature crée avec une clé privée de l'utilisateur. Le serveur DOIT vérifier que la clé est un authentifiant valide pour l'utilisateur, et DOIT vérifier que la signature est valide. Si les deux tiennent, la demande d'authentification DOIT être acceptée ; autrement, elle DOIT être rejetée. Noter que le serveur PEUT exiger des authentifications supplémentaires après une authentification réussie.

Les clés privées sont souvent mémorisées en forme chiffrée chez l'hôte client, et l'utilisateur doit fournir une phrase de passe avant que la signature puisse être générée. Même si elles ne le sont pas, l'opération de signature implique des calculs coûteux. Pour éviter des traitements et des interactions d'utilisateur inutiles, le message suivant est fourni pour demander si l'authentification utilisant la méthode "publickey" serait acceptable.

octet : SSH_MSG_USERAUTH_REQUEST
chaîne : nom d'utilisateur codé en ISO-10646 UTF-8 [RFC3629]

chaîne : nom de service en US-ASCII
chaîne : "publickey"
booléen : FAUX
chaîne : nom d'algorithme de clé publique
chaîne : module de clé publique

Les algorithmes de clé publique sont définis dans la spécification de la couche transport [RFC4253]. Le 'module de clé publique' peut contenir des certificats.

Tout algorithme de clé publique peut être offert à l'utilisation dans l'authentification. En particulier, la liste n'est pas contrainte par ce qui a été négocié durant l'échange de clés. Si le serveur ne prend pas en charge un certain algorithme, il DOIT simplement rejeter la demande.

Le serveur DOIT répondre à ce message avec SSH_MSG_USERAUTH_FAILURE ou avec ce qui suit :
octet : SSH_MSG_USERAUTH_PK_OK
chaîne : nom d'algorithme de clé publique provenant de la demande
chaîne : module de clé publique provenant de la demande

Pour effectuer l'authentification réelle, le client PEUT alors envoyer une signature générée en utilisant la clé privée. Le client PEUT envoyer la signature directement sans d'abord vérifier si la clé est acceptable. La signature est envoyée en utilisant le paquet suivant :
octet : SSH_MSG_USERAUTH_REQUEST
chaîne : nom d'utilisateur
chaîne : nom de service
chaîne : "publickey"
booléen : VRAI
chaîne : nom d'algorithme de clé publique
chaîne : clé publique à utiliser pour l'authentification
chaîne : signature

La valeur de 'signature' est une signature par la clé privée correspondante sur les données suivantes, dans l'ordre donné :
chaîne : identifiant de session
octet : SSH_MSG_USERAUTH_REQUEST
chaîne : nom d' utilisateur
chaîne : nom de service
chaîne : "publickey"
booléen : VRAI
chaîne : nom d'algorithme de clé publique
chaîne : clé publique à utiliser pour l'authentification

Quand le serveur reçoit ce message, il DOIT vérifier si la clé fournie est acceptable pour l'authentification, et si oui, il DOIT vérifier si la signature est correcte.

Si les deux vérifications réussissent, cette méthode est réussie. Noter que le serveur peut exiger des authentifications supplémentaires. Le serveur DOIT répondre avec SSH_MSG_USERAUTH_SUCCESS (si d'autres authentifications ne sont pas nécessaires) ou SSH_MSG_USERAUTH_FAILURE (si la demande a échoué, ou si plus d'authentifications sont nécessaires).

Les numéros de message spécifiques de la méthode suivants sont utilisés par la méthode d'authentification "publickey" :

SSH_MSG_USERAUTH_PK_OK 60

8. Méthode d'authentification par mot de passe "password"

L'authentification par mot de passe utilise les paquets suivants. Noter qu'un serveur PEUT demander qu'un utilisateur change le mot de passe. Toutes les mises en œuvre DEVRAIENT prendre en charge l'authentification par mot de passe.

octet : SSH_MSG_USERAUTH_REQUEST
chaîne : nom d'utilisateur
chaîne : nom de service
chaîne : "mot de passe"

booléen : FAUX
chaîne : mot de passe en clair codé en ISO-10646 UTF-8 [RFC3629]

Noter que la valeur de 'mot de passe en clair' est codée en ISO-10646 UTF-8. Il appartient au serveur de déterminer comment interpréter le mot de passe et le valider par rapport à la base de données de mots de passe. Cependant, si le client lit le mot de passe dans un autre codage (par exemple, ISO 8859-1 - ISO Latin1) il DOIT convertir le mot de passe en ISO-10646 UTF-8 avant de le transmettre, et le serveur DOIT convertir le mot de passe en le codage utilisé sur ce système pour mots de passe.

Du point de vue de l'internationalisation, il est désiré que si un utilisateur entre son mot de passe, le processus d'authentification fonctionne sans considération du système d'exploitation et du logiciel de client qu'utilise l'utilisateur. Faire ainsi exige la normalisation. Les systèmes qui prennent en charge des mots de passe non ASCII DEVRAIENT toujours normaliser les mots de passe et les noms d'utilisateurs chaque fois qu'ils sont ajoutés à la base de données, ou comparés (avec ou sans hachage) aux entrées existantes dans la base de données. Les mises en œuvre de SSH qui mémorisent les mots de passe et les comparent DEVRAIENT utiliser la [RFC4013] pour la normalisation.

Noter que bien que le mot de passe en clair soit transmis dans le paquet, le paquet entier est chiffré par la couche transport. Le serveur et le client devraient tous deux vérifier si la couche transport sous-jacente assure la confidentialité (c'est-à-dire, si le chiffrement est utilisé). Si aucune confidentialité n'est fournie (chiffrement "none") l'authentification par mot de passe DEVRAIT être désactivée. Si il n'y a pas de confidentialité ou pas de MAC, le changement de mot de passe DEVRAIT être désactivé.

Normalement, le serveur répond à ce message par succès ou échec. Cependant, si le mot de passe est arrivé à expiration, le serveur DEVRAIT indiquer cela en répondant avec SSH_MSG_USERAUTH_PASSWD_CHANGEREQ *(message SSH de demande de changement de mot de passe d'authentification d'utilisateur)*. Dans tous les cas, le serveur NE DOIT PAS permettre qu'un mot de passe arrivé à expiration soit utilisé pour l'authentification.

octet : SSH_MSG_USERAUTH_PASSWD_CHANGEREQ
chaîne : invite codée en ISO-10646 UTF-8 [RFC3629]
chaîne : étiquette de langue [RFC3066]

Dans ce cas, le client PEUT continuer avec une méthode d'authentification différente, ou demander un nouveau mot de passe à l'utilisateur et réessayer l'authentification par mot de passe en utilisant le message suivant. Le client PEUT aussi envoyer ce message à la place de la demande normale d'authentification par mot de passe sans que le serveur la demande.

octet : SSH_MSG_USERAUTH_REQUEST
chaîne : nom d'utilisateur
chaîne : nom de service
chaîne : "mot de passe"
booléen : VRAI
chaîne : vieux mot de passe en clair codé en ISO-10646 UTF-8 [RFC3629]
chaîne : nouveau mot de passe en clair codé en ISO-10646 UTF-8 [RFC3629]

Le serveur doit répondre à chaque message de demande avec SSH_MSG_USERAUTH_SUCCESS, SSH_MSG_USERAUTH_FAILURE, ou un autre SSH_MSG_USERAUTH_PASSWD_CHANGEREQ. Leur signification est :

SSH_MSG_USERAUTH_SUCCESS : le mot de passe a été changé, et l'authentification s'est bien achevée.

SSH_MSG_USERAUTH_FAILURE avec succès partiel : le mot de passe a été changé, mais plus authentifications sont nécessaires.

SSH_MSG_USERAUTH_FAILURE sans succès partiel : le mot de passe n'a pas été changé. Soit le changement du mot de passe n'était pas accepté, soit le vieux mot de passe était mauvais. Noter que si le serveur a déjà envoyé SSH_MSG_USERAUTH_PASSWD_CHANGEREQ, on sait qu'il accepte de changer le mot de passe.

SSH_MSG_USERAUTH_PASSWD_CHANGEREQ : le mot de passe n'a pas changé parce que le nouveau mot de passe n'est pas acceptable (par exemple, trop facile à deviner).

Les numéros de message spécifiques de la méthode suivants sont utilisés par la méthode d'authentification par mot de passe :

SSH_MSG_USERAUTH_PASSWD_CHANGEREQ 60

9. Authentification fondée sur l'hôte "hostbased"

Certains sites souhaitent permettre l'authentification sur la base de l'hôte d'où vient l'utilisateur et du nom d'utilisateur sur l'hôte distant. Bien que cette forme d'authentification ne convienne pas pour les sites avec des exigences de haute sécurité, elle peut être très pratique dans de nombreux environnements. Cette forme d'authentification est FACULTATIVE. Quand elle est utilisée, une attention particulière DEVRAIT être portée à empêcher un utilisateur régulier d'obtenir la clé privée d'hôte.

Le client demande cette forme d'authentification en envoyant le message qui suit. Il est similaire aux styles d'authentification UNIX "rhosts" et "hosts.equiv", sauf que l'identité de l'hôte client est vérifiée plus rigoureusement.

Cette méthode fonctionne avec l'envoi par le client d'une signature créée avec la clé privée de l'hôte client, que le serveur vérifie avec la clé publique de l'hôte. Une fois que l'identité de l'hôte client est établie, l'autorisation (mais pas d'autre authentification) est effectuée sur la base des noms d'utilisateur sur le serveur et le client, et le nom de l'hôte client.

octet : SSH_MSG_USERAUTH_REQUEST
chaîne : nom d'utilisateur
chaîne : nom de service
chaîne : "hostbased"
chaîne : algorithme de clé publique pour clé d'hôte
chaîne : clé publique d'hôte et certificats pour hôte client
chaîne : nom d'hôte client exprimé comme FQDN en US-ASCII
chaîne : nom d'utilisateur sur l'hôte client codé en ISO-10646 UTF-8 [RFC3629]
chaîne : signature

Les noms d'algorithmes de clé publique à utiliser dans 'algorithme de clé publique pour clé d'hôte' sont définis dans la spécification de la couche transport [RFC4253]. La chaîne 'clé publique d'hôte et certificats pour hôte client' peut inclure des certificats.

La valeur de 'signature' est une signature avec la clé privée d'hôte des données qui suivent, dans l'ordre indiqué :
chaîne : identifiant de session
octet : SSH_MSG_USERAUTH_REQUEST
chaîne : nom d'utilisateur
chaîne : nom de service
chaîne : "hostbased"
chaîne : algorithme de clé publique pour clé d'hôte
chaîne : clé publique d'hôte et certificats pour hôte client
chaîne : nom d'hôte client exprimé comme FQDN en US-ASCII
chaîne : nom d'utilisateur sur l'hôte client codé en ISO-10646 UTF-8 [RFC3629]

Le serveur DOIT vérifier que la clé d'hôte appartient bien à l'hôte client nommé dans le message, que l'utilisateur sur cet hôte est autorisé à se connecter, et que la valeur de 'signature' est une signature valide sur la valeur appropriée par la clé d'hôte donnée. Le serveur PEUT ignorer le 'nom d'utilisateur' du client si il veut n'authentifier que l'hôte client.

Chaque fois que possible, il est RECOMMANDÉ que le serveur effectue des vérifications supplémentaires pour vérifier que l'adresse réseau obtenue du réseau (qui n'est pas de confiance) correspond au nom donné pour l'hôte client. Cela rend plus difficile l'exploitation de clés d'hôte compromises. Noter que cela peut exiger un traitement particulier pour les connexions passant à travers un pare-feu.

10. Considérations relatives à l'IANA

Le présent document fait partie d'un ensemble. Les considérations relatives à l'IANA pour le protocole SSH, tel que défini dans les [RFC4251], [RFC4253], [RFC4254], et le présent document, sont détaillées dans la [RFC4250].

11. Considérations sur la sécurité

L'objet de ce protocole est d'effectuer l'authentification de l'utilisateur du client. On suppose qu'il fonctionne sur un protocole de couche transport sûre, qui a déjà authentifié la machine serveur, établi un canal de communications chiffré, et

a calculé un identifiant de session univoque pour cette session. La couche transport assure le secret de transmission pour l'authentification par mot de passe et les autres méthodes qui s'appuient sur des données secrètes.

Les considérations de sécurité complètes pour ce protocole figurent dans la [RFC4251].

12. Références

12.1 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)

[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)

[RFC3066] H. Alvestrand, "Étiquettes pour l'identification des langues", BCP 47, janvier 2001. *(Obsolète, voir la* RFC4646.)

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

[RFC4013] K. Zeilenga, "SASLprep : Profil Stringprep pour les noms d'utilisateur et mots de passe", février 2005.

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

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

[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)

12.2 Référence pour information

[ssh-1.2.30] Ylonen, T., "ssh-1.2.30/RFC", Filchier compressé en tarball : ftp://ftp.funet.fi/pub/unix/security/login/ssh/ssh-1.2.30.tar.gz, novembre 1995.

Adresse des auteurs

Tatu Ylonen
SSH Communications Security Corp
Valimotie 17
00380 Helsinki
Finland
mél : ylo@ssh.com

Chris Lonvick (editor)
Cisco Systems, Inc.
12515 Research Blvd.
Austin 78759
USA
mél : clonvick@cisco.com

Notice de marque commerciale

"ssh" est une marque commerciale déposée au États Unis d'Amérique et/ou dans d'autres pays.

Déclaration complète de droits de reproduction

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