Groupe de travail Réseau
Request for Comments : 4256
Catégorie : Sur la voie de la normalisation
Traduction Claude Brière de L'Isle

F. Cusack, savecore.net
M. Forssen, AppGate Network Security AB

janvier 2006

Authentification d'échange de message générique pour le protocole 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 une méthode d'authentification générique pour le protocole SSH, qui convient pour les authentifications interactives où les données d'authentification devraient être entrées via un clavier (ou un appareil d'entrée alphanumérique équivalent). Le but principal de cette méthode est de permettre au client SSH de prendre en charge toute une classe de mécanismes d'authentification sans connaître les spécificités du ou des mécanismes d'authentification réels.

Table des matières

1. Introduction

Le protocole d'authentification SSH [RFC4252] est un protocole générique d'authentification d'utilisateur. Il est destiné à fonctionner sur le protocole de couche transport SSH [RFC4253]. Le protocole d'authentification suppose que les protocoles sous-jacents assurent la protection de l'intégrité et de la confidentialité.

Le présent document décrit une méthode d'authentification générique pour le protocole d'authentification SSH. Cette méthode convient pour les méthodes d'authentification interactive qui n'ont pas besoin de la prise en charge d'un logiciel spécial du côté client. Toutes les données d'authentification devraient plutôt être entrées via le clavier. Le but principal de cette méthode est de permettre au client SSH d'avoir une faible connaissance, ou d'ignorer complètement, les spécificités du ou des mécanismes d'authentification sous-jacents utilisés par le serveur SSH. Cela permet au serveur de choisir arbitrairement ou de changer le ou les mécanismes d'authentification sous-jacents sans avoir à mettre à jour le code du client.

Le nom de cette méthode d'authentification est "keyboard-interactive".

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 autre référence ou explication.

Le présent document décrit aussi les interactions du client avec l'utilisateur pour obtenir les informations d'authentification. Bien que cela sorte un peu du domaine d'application d'une spécification de protocole, on le décrit ici de toutes façons parce que certains aspects du protocole sont spécifiquement conçus sur la base des problèmes d'interface d'utilisateur, et omettre ces informations peut conduire à des mises en œuvre incompatibles ou bancales.

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].

2. Motifs

Les méthodes d'authentification actuellement définies pour SSH sont étroitement couplées avec le mécanisme d'authentification sous-jacent. Cela rend difficile l'ajout de nouveaux mécanismes d'authentification car tous les clients doivent être mis à jour pour prendre en charge le nouveau mécanisme. Avec la méthode générique définie ici, les clients n'auront pas besoin de changements de code pour prendre en charge de nouveaux mécanismes d'authentification, et si une couche d'authentification séparée est utilisés, comme avec [PAM], alors le serveur peut n'avoir pas non plus besoin de changement de code.

Cela présente un avantage significatif sur les autres méthodes, comme celle du "mot de passe" (définie dans la [RFC4252]) car de nouvelles méthodes (en principe plus fortes) peuvent être ajoutées "à volonté" et la sécurité du système peut être améliorée de façon transparente.

Les mécanismes de défi-réponse et de mot de passe à utilisation unique sont aussi facilement pris en charge par cette méthode d'authentification.

Cependant, cette méthode d'authentification se limite aux mécanismes d'authentification qui n'exigent pas de code particulier, comme les pilotes matériels ou l'essoreuse de mot de passe, chez le client.

3. Échanges du protocole

Le client initie l'authentification avec un message SSH_MSG_USERAUTH_REQUEST *(message SSH de demande d'authentification d'utilisateur)*. Le serveur demande alors les informations d'authentification au client avec un message SSH_MSG_USERAUTH_INFO_REQUEST *(message SSH de demande d'informations d'authentification d'utilisateur)*. Le client obtient les informations de l'utilisateur et répond ensuite avec un message SSM_MSG_USERAUTH_INFO_RESPONSE *(message SSH de réponse d'informations d'authentification d'utilisateur)*. Le serveur NE DOIT PAS envoyer d'autre SSH_MSG_USERAUTH_INFO_REQUEST avant d'avoir reçu la réponse du client.

3.1 Échange initial

L'authentification commence avec l'envoi par le client du paquet suivant :

octet : SSH_MSG_USERAUTH_REQUEST
chaîne : nom d'utilisateur (ISO-10646 UTF-8, comme défini dans la [RFC3629])
chaîne : nom de service (US-ASCII)
chaîne : "keyboard-interactive" (US-ASCII)
chaîne : étiquette de langue (comme défini dans la [RFC3066])
chaîne : sous méthodes (ISO-10646 UTF-8)

L'étiquette de langue est déconseillée et DEVRAIT être la chaîne vide. Elle pourrait être supprimée dans une future révision de cette spécification. Le serveur DEVRAIT plutôt choisir le langage à utiliser sur la base des étiquettes communiquées durant l'échange de clés [RFC4253].

Si l'étiquette de langue n'est pas la chaîne vide, le serveur DEVRAIT utiliser le langage spécifié pour tous les messages envoyés au client au titre de ce protocole. L'étiquette de langue NE DEVRAIT PAS être utilisée pour le choix du langage

pour des messages en dehors de ce protocole. Si le serveur ne prend pas en charge le langage demandé, le langage à utiliser dépend de la mise en œuvre.

Le champ "sous méthodes" est inclus afin que l'utilisateur puisse donner un conseil sur les méthodes qu'il veut utiliser réellement. C'est une liste séparée par des virgules des sous méthodes d'authentification (logicielles ou matérielles) que l'utilisateur préfère. Si le client a connaissance des sous méthodes préférées par l'utilisateur, vraisemblablement par un réglage de configuration, il PEUT utiliser le champ "sous méthodes" pour passer ces informations au serveur. Autrement, il DOIT envoyer la chaîne vide.

L'utilisateur et le serveur doivent se mettre d'accord sur les noms des sous méthodes.

L'interprétation par le serveur du champ "sous méthodes" dépend de la mise en œuvre.

Une stratégie possible de mise en œuvre du champ "sous méthodes" sur le serveur est que, sauf si l'utilisateur peut utiliser plusieurs sous méthodes différentes, le serveur ignore ce champ. Si l'utilisateur peut authentifier l'utilisation de l'une de ces diverses sous méthodes, le serveur devrait traiter le champ "sous méthodes" comme une indication de la sous méthode que l'utilisateur veut utiliser à ce moment.

Noter que quand ce message est envoyé au serveur, le client n'a pas encore invité l'utilisateur à fournir un mot de passe, et donc que ces informations NE SONT PAS incluses dans ce message initial (à la différence de la méthode "password").

Le serveur DOIT répondre par un message SSH_MSG_USERAUTH_SUCCESS *(réussite)*, SSH_MSG_USERAUTH_FAILURE *(échec)*, ou SSH_MSG_USERAUTH_INFO_REQUEST.

Le serveur NE DEVRAIT PAS répondre par le message SSH_MSG_USERAUTH_FAILURE si l'échec se fonde sur le nom d'utilisateur ou sur le nom de service ; il DEVRAIT plutôt envoyer un ou des messages SSH_MSG_USERAUTH_INFO_REQUEST, qui ressemblent tout à fait à ceux qui auraient été envoyés dans les cas où l'authentification devrait se poursuivre, et ensuite envoyer le message d'échec (après un délai convenable, comme décrit plus loin). L'objectif est de rendre impossible de trouver des noms d'utilisateur valides en comparant les résultats de l'authentification de différents utilisateurs.

Le serveur PEUT répondre avec un message SSH_MSG_USERAUTH_SUCCESS si aucune authentification n'est exigée pour l'utilisateur en question. Cependant, une meilleure approche, pour les raisons discutées ci-dessus, pourrait être de répondre avec un message SSH_MSG_USERAUTH_INFO_REQUEST et d'ignorer la réponse (ne pas la valider).

3.2 Demandes d'information

Les demandes sont générées à partir du serveur en utilisant le message SSH_MSG_USERAUTH_INFO_REQUEST.

Le serveur peut envoyer autant de demandes que nécessaire pour authentifier le client ; le client DOIT être prêt à traiter plusieurs échanges. Cependant, le serveur NE DOIT PAS avoir en cours plus d'un message SSH_MSG_USERAUTH_INFO_REQUEST. C'est-à-dire qu'il ne peut pas envoyer une autre demande avant que le client ait répondu.

Le message SSH_MSG_USERAUTH_INFO_REQUEST est défini comme suit :

octet : SSH_MSG_USERAUTH_INFO_REQUEST
chaîne : nom (ISO-10646 UTF-8)
chaîne : instruction (ISO-10646 UTF-8)
chaîne : étiquette de langue (comme défini dans la [RFC3066])
entier : numéros d'invite
chaîne : invite[1] (ISO-10646 UTF-8)
booléen : écho[1]
...
chaîne : invite[numéro d'invites] (ISO-10646 UTF-8)
booléen : écho[numéro d'invite]

L'étiquette de langue est déconseillée et DEVRAIT être la chaîne vide. Elle pourrait être supprimée dans une future révision de cette spécification. Le serveur DEVRAIT plutôt choisir le langage utilisé sur la base des étiquettes communiquées durant l'échange de clé [RFC4253].

Si l'étiquette de langue n'est pas la chaîne vide, le serveur DEVRAIT utiliser le langage spécifié pour tous les messages envoyés au client au titre de ce protocole. L'étiquette de langue NE DEVRAIT PAS être utilisée pour le choix de langue pour les messages hors de ce protocole. Si le serveur ne prend pas en charge le langage demandé, le langage à utiliser dépend de la mise en œuvre.

Le serveur DEVRAIT prendre en considération que certains clients peuvent n'être pas capables d'afficher correctement un nom ou champ d'invite long (voir au paragraphe suivant) et limiter si possible la longueur de ces champs. Par exemple, au lieu d'un champ d'instruction de "Entrer le mot de passe" et un champ d'invite de "Mot de passe pour user23@host.domain:", un meilleur choix pourrait être un champ d'instruction de "Authentification de mot de passe pour user23@host.domain" et un champ d'invite de "Mot de passe: ". On s'attend à ce que cette méthode d'authentification soit normalement reprise par [PAM] et que de tels choix ne seraient pas autrement possibles.

Les champs de nom et instruction PEUVENT être des chaînes vides ; le client DOIT être prêt à traiter cela correctement. Le ou les champs d'invite NE DOIVENT PAS être des chaînes vides.

Le champ numéro d'invite peut être `0', et dans ce cas, il n'y aura pas de champ invite/écho dans le message, mais le client DEVRAIT quand même afficher les champs de nom et instruction (comme décrit ci-dessous).

3.3 Interface d'utilisateur

À réception d'un message de demande, le client DEVRAIT inviter l'utilisateur comme suit :

Un client d'interface de ligne de commande (CLI, *command line interface*) DEVRAIT imprimer le nom et l'instruction (si ils ne sont pas vides) en ajoutant de nouvelles lignes. Ensuite, pour chaque invite tour à tour, le client DEVRAIT afficher l'invite et lire l'entrée de l'utilisateur.

Un client d'interface d'utilisateur graphique (GUI, *graphical user interface*) a de nombreux choix sur la façon d'inviter l'utilisateur. Une possibilité est d'utiliser le champ "nom" (éventuellement précédé du nom de l'application) comme titre de la fenêtre de dialogue dans laquelle les invites sont présentées. Dans cette fenêtre de dialogue, le champ "instruction" sera un message de texte, et les invites seront des étiquettes pour les champs d'entrée de texte. Tous les champs DEVRAIENT être présentés à l'utilisateur. Par exemple, une mise en œuvre NE DEVRAIT PAS éliminer le champ "nom" parce que sa fenêtre n'a pas de titres ; elle DEVRAIT plutôt trouver un autre moyen d'afficher ces informations. Si les invites sont présentées dans une fenêtre de dialogue, le client NE DEVRAIT alors PAS présenter chaque invite dans une fenêtre séparée.

Tous les clients DOIVENT traiter de façon appropriée un champ "instruction" avec les nouvelles lignes incorporées. Ils DEVRAIENT aussi être capables d'afficher au moins 30 caractères pour le nom et les invites. Si le serveur présente des noms ou invites de plus de 30 caractères, le client PEUT tronquer ces champs à la longueur qu'il peut afficher. Si le client tronque des champs, il DOIT y avoir une indication évidente qu'une telle troncature s'est produite. Le champ "instruction" NE DEVRAIT PAS être tronqué.

Les clients DEVRAIENT utiliser le filtrage des caractères de contrôle, comme discuté dans la [RFC4251], pour éviter des attaques par inclusion de caractères de contrôle de terminal dans les champs à afficher.

Pour chaque invite, le champ d'écho correspondant indique si l'entrée d'utilisateur devrait avoir un écho lorsque les caractères sont frappés. Les clients DEVRAIENT correctement faire écho/masquer les entrées d'utilisateur pour chaque invite indépendamment des autres invites dans le message de demande. Si un client n'honore pas le champ "écho" pour une raison quelconque, le client DOIT alors s'égarer du côté du masquage d'entrée. Un client GUI peut souhaiter avoir une boîte de vérification pour basculer l'écho/masque. Les clients NE DEVRAIENT PAS ajouter de caractères supplémentaires à l'invite, comme des ": " (deux points espace) ; le serveur est chargé de fournir tout le texte à afficher à l'utilisateur. Les clients DOIVENT aussi accepter des réponses vides de l'utilisateur et les passer comme des chaînes vides.

3.4 Réponses d'information

Après avoir obtenu les informations demandées de l'utilisateur, le client DOIT répondre avec un message SSH_MSG_USERAUTH_INFO_RESPONSE.

Le format du message SSH_MSG_USERAUTH_INFO_RESPONSE est le suivant :

octet : SSH_MSG_USERAUTH_INFO_RESPONSE
entier : numéro de réponse

chaîne : réponse[1] (ISO-10646 UTF-8)
...
chaîne : réponse[numéro de réponse] (ISO-10646 UTF-8)

Noter que les réponses sont codées en ISO-10646 UTF-8. Il appartient au serveur d'interpréter les réponses et de les valider. Cependant, si le client lit les réponses dans un autre codage (par exemple, ISO 8859-1) il DOIT convertir les réponses en ISO-10646 UTF-8 avant de les transmettre.

Du point de vue de l'internationalisation, il est souhaité que si un utilisateur entre des réponses, le processus d'authentification fonctionnera sans considération du système d'exploitation et du logiciel client qu'il utilise. Cela exige une normalisation. Les systèmes qui prennent en charge des mots de passe non ASCII DEVRAIENT toujours normaliser les mots de passe et les noms d'utilisateur chaque fois qu'ils les ajoutent à la base de données, ou les comparent (avec ou sans hachage) aux entrées existantes de 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.

Si le champ "num-responses" ne correspond pas au champ "num-prompts" dans le message de demande, le serveur DOIT envoyer un message d'échec.

Dans le cas où le serveur envoie un champ "num-prompts" à zéro dans le message de demande, le client DOIT envoyer un message de réponse avec un champ "num-responses" de zéro pour achever l'échange.

Les réponses DOIVENT être dans l'ordre des invites. C'est-à-dire que réponse[n] DOIT être la réponse à invite[n].

Après avoir reçu la réponse, le serveur DOIT envoyer un message SSH_MSG_USERAUTH_SUCCESS, SSH_MSG_USERAUTH_FAILURE, ou un autre message SSH_MSG_USERAUTH_INFO_REQUEST.

Si le serveur échoue à authentifier l'utilisateur (par le mécanisme d'authentification sous-jacent) il NE DEVRAIT PAS envoyer un autre message de demande pour tenter d'obtenir de nouvelles données d'authentification ; il DEVRAIT plutôt envoyer un message d'échec. La seule fois où le serveur devrait envoyer plusieurs messages de demande est si des données d'authentification supplémentaires sont nécessaires (c'est-à-dire, parce qu'il y a plusieurs mécanismes d'authentification sous-jacents qui doivent être utilisés pour authentifier l'utilisateur).

Si le serveur entend répondre avec un message d'échec, il PEUT retarder d'un délai qui dépend de la mise en œuvre l'envoi au client. On soupçonne que les mises en œuvre vont probablement rendre ce délai configurable ; une valeur par défaut suggérée est de 2 secondes.

4. Exemples d'authentification

Voici deux exemples d'échanges entre un client et un serveur. Le premier est un exemple de défi/réponse avec un jeton manuel. C'est une authentification qui n'est pas possible avec les autres méthodes d'authentification.

C : octet SSH_MSG_USERAUTH_REQUEST
C : chaîne "user23"
C : chaîne "ssh-userauth"
C : chaîne "keyboard-interactive"
C : chaîne ""
C : chaîne ""

S : octet SSH_MSG_USERAUTH_INFO_REQUEST
S : chaîne "CRYPTOCard Authentication"
S : chaîne "Le défi est '14315716'"
S : chaîne "fr"
S : entier 1
S : chaîne "Réponse: "
S : booléen VRAI

[Le client invite l'utilisateur à entrer le mot de passe]

C : octet SSH_MSG_USERAUTH_INFO_RESPONSE
C : entier 1
C : chaîne "6d757575"

S : octet SSH_MSG_USERAUTH_SUCCESS

Le second exemple est une authentification par mot de passe standard ; dans ce cas, le mot de passe de l'utilisateur est arrivé à expiration.

C : octet SSH_MSG_USERAUTH_REQUEST
C : chaîne "user23"
C : chaîne "ssh-userauth"
C : chaîne "keyboard-interactive"
C : chaîne "fr"
C : chaîne ""

S : octet SSH_MSG_USERAUTH_INFO_REQUEST
S : chaîne "Authentification par mot de passe"
S : chaîne ""
S : chaîne "fr"
S : entier 1
S : chaîne "Mot de passs: "
S : booléen FAUX

[Le client invite l'utilisateur à entrer le mot de passe]

C : octet SSH_MSG_USERAUTH_INFO_RESPONSE
C : entier 1
C : chaîne "mot de passe"

S : octet SSH_MSG_USERAUTH_INFO_REQUEST
S : chaîne "Mot de passe expiré"
S : chaîne "Votre mot de passe est expiré."
S : chaîne "fr"
S : entier 2
S : chaîne "Entrer un nouveau mot de passe: "
S: booléen FAUX
S: chaîne "Entrer le à nouveau: "
S: booléen FAUX

[Le client invite l'utilisateur à entrer un nouveau mot de passe]

C : octet SSH_MSG_USERAUTH_INFO_RESPONSE
C : entier 2
C : chaîne "newpass"
C : chaîne "newpass"

S : octet SSH_MSG_USERAUTH_INFO_REQUEST
S : chaîne "Mot de passe changé"
S : chaîne "Mot de passe change avec succès pour user23."
S : chaîne "fr"
S : entier 0

[Le client affiche le message à l'utilisateur]

C : octet SSH_MSG_USERAUTH_INFO_RESPONSE
C : entier 0

S : octet SSH_MSG_USERAUTH_SUCCESS

5. Considérations relatives à l'IANA

Le type de userauth "keyboard-interactive" est utilisé pour cette méthode d'authentification.

Les constantes spécifiques de la méthode suivantes sont utilisées avec cette méthode d'authentification:

SSH_MSG_USERAUTH_INFO_REQUEST 60
SSH_MSG_USERAUTH_INFO_RESPONSE 61

6. Considérations sur la sécurité

Le protocole d'authentification et cette méthode d'authentification dépendent de la sécurité de la couche de transport SSH sous-jacente. Sans la confidentialité qui y est fournie, toutes les données d'authentification passées avec cette méthode sont sujettes à interception.

Le nombre d'échanges client-serveur exigés pour achever une authentification utilisant cette méthode peut être variable. Il est possible qu'un observateur puisse obtenir des informations précieuses simplement en comptant ce nombre. Par exemple, un observateur peut deviner qu'un mot de passe d'utilisateur est arrivé à expiration, et avec un peu plus d'observations peut être capable de déterminer la durée de vie de mot de passe imposée par la politique d'expiration de mot de passe d'un site.

7. Références

7.1 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)

[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.

[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)

7.2 Référence pour information

[PAM] Samar, V., Schemers, R., "Unified Login With Pluggable Authentication Modules (PAM)", OSF RFC 86.0, octobre 1995.

Adresse des auteurs

Frank Cusack
savecore.net
mél : frank@savecore.net

Martin Forssen
AppGate Network Security AB
Otterhallegatan 2
SE-411 18 Gothenburg
SWEDEN
mél : maf@appgate.com

Déclaration complète de droits de reproduction