Groupe de travail Réseau
Request for Comments : 3645
RFC mise à jour : 2845
Catégorie : En cours de normalisation
Traduction Claude Brière de L'Isle

S. Kwan, P. Garg, J. Gilroy, L. Esibov, J. Westhead
Microsoft Corp.
R. Hall, Lucent Technologies
octobre 2003

Algorithme générique de service de sécurité pour l'authentification de transaction de clé secrète pour le DNS (GSS-TSIG)

Résumé

Le protocole d'authentification de transaction de clé secrète pour le DNS (TSIG) fournit l'authentification au niveau de la transaction pour le DNS. TSIG est extensible par la définition de nouveaux algorithmes. Le présent document spécifie un algorithme fondé sur l'Interface générique de programme d'application de service de sécurité (GSS-API) (RFC2743). Le présent document met à jour la RFC 2845.

Table des Matières

1. Introduction

Le protocole d'authentification de transaction de clé secrète pour le DNS (TSIG) [RFC2845] a été développé pour fournir une authentification et une protection de l'intégrité légères des messages entre deux entités du DNS, comme entre un client et un serveur ou entre deux serveurs. TSIG peut être utilisé pour protéger les messages de mise à jour dynamique, authentifier des

messages réguliers ou pour décharger d'un traitement DNSSEC compliqué [RFC2535] un client au profit d'un serveur et permettre quand même au client d'être assuré de l'intégrité des réponses.

Le protocole TSIG [RFC2845] est extensible par la définition de nouveaux algorithmes. Le présent document spécifie un algorithme fondé sur l'interface de programme d'application de service de sécurité générique (GSS-API, *Generic Security Service Application Program Interface*) [RFC2743]. GSS-API est un cadre qui fournit une sécurité abstraite au développeur de protocole d'application. Les services de sécurité offerts peuvent inclure l'authentification, l'intégrité, et la confidentialité.

Le cadre GSS-API présente plusieurs avantages :

* Indépendance du mécanisme et du protocole. Les mécanismes sous-jacents qui réalisent les services de sécurité peuvent être négociés au vol et peuvent varier au fil du temps. Par exemple, un client et un serveur PEUVENT utiliser Kerberos [RFC1964] pour une transaction, tandis que le même serveur PEUT utiliser SPKM [RFC2025] avec un autre client.

* Le développeur du protocole est déchargé de la responsabilité de la création et de la gestion d'une infrastructure de sécurité. Par exemple, le développeur n'a pas besoin de créer de nouveaux systèmes de distribution ou de gestion de clés. Le développeur s'appuie plutôt sur le mécanisme du service de sécurité pour gérer cela en son nom.

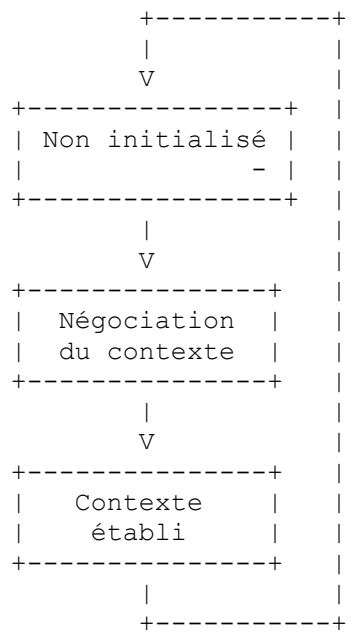
La portée du présent document se limite à la description d'un mécanisme d'authentification. Il ne discute ni ne propose un mécanisme d'autorisation. Les lecteurs qui ne seraient pas faniliarisés avec les concepts de GSS-API sont invités à lire la section 1 "Caractéristiques et concepts" de la [RFC2743] avant d'examiner en détail le présent protocole. On suppose aussi que le lecteur s'est familiarisé avec les [RFC2845], [RFC2930], [RFC1034] et [RFC1035].

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 la [RFC2119].

2. Vue d'ensemble de l'algorithme

Dans GSS, client et serveur interagissent pour créer un "contexte de sécurité". Le contexte de sécurité peut être utilisé pour créer et vérifier les signatures des transactions sur les messages entre les deux parties. Un contexte de sécurité unique est exigé pour chaque connexion unique entre client et serveur.

Créer un contexte de sécurité implique une négociation entre client et serveur. Une fois qu'un contexte a été établi, il a une durée de vie finie pendant laquelle il peut être utilisé pour sécuriser les messages. Il y a donc trois états d'un contexte qui sont associés à une connexion :



Chaque connexion commence dans l'état non initialisé.

2.1 Détails de GSS

Le client et le serveur DOIVENT être authentifiés localement et avoir acquis les accréditifs par défaut avant d'utiliser le présent protocole comme spécifié au paragraphe 1.1.1 "Accréditifs" de la [RFC2743].

L'algorithme GSS-TSIG comporte deux étapes :

I. Établir le contexte de sécurité. Le client et le serveur utilisent les API GSS_Init_sec_context et GSS_Accept_sec_context pour générer les jetons qu'ils se passent l'un l'autre en utilisant la [RFC2930] comme mécanisme de transport.

II. Une fois le contexte de sécurité établi, il est utilisé pour générer et vérifier les signatures en utilisant les API GSS_GetMIC et GSS_VerifyMIC. Ces signatures sont échangées par le client et le serveur au titre des enregistrements TSIG échangés dans les messages DNS envoyés entre le client et le serveur, comme décrit dans la [RFC2845].

2.2 Modifications au protocole TSIG (RFC2845)

Des modifications à la [RFC2845] permettent l'utilisation de TSIG en signant la réponse du serveur dans un endroit explicitement spécifié dans l'échange multi messages entre deux entités DNS même si la demande du client n'était pas signée.

Précisément, le paragraphe 4.2 de la RFC2845 DOIT être modifié comme suit :

Remplacer : "Le serveur NE DOIT PAS générer une réponse signée à une demande non signée."

Par : "Le serveur NE DOIT PAS générer une réponse signée à une demande non signée, sauf dans le cas d'une réponse à une interrogation TKEY non signée du client si une clé secrète est établie du côté serveur après que le serveur a traité l'interrogation du client. Signer les réponses aux interrogations TKEY non signées DOIT être spécifié explicitement dans la description d'un algorithme d'établissement de clé secrète individuelle."

3. Détails du protocole client

Un contexte unique est exigé pour chaque serveur auquel le client envoie des messages sûrs. Un contexte est identifié par une bride de contexte. Un client entretient une transposition entre les serveurs et les brides:

(target_name, key_name, context_handle) *(nom de cible, nom de clé, bride de contexte)*

La valeur key_name identifie aussi une bride de contexte. Le key_name est le nom du propriétaire des enregistrements TKEY et TSIG envoyés entre un client et un serveur pour s'indiquer l'un l'autre quel contexte DOIT être utilisé pour traiter la demande en cours.

Le client et le serveur DNS PEUVENT utiliser divers mécanismes de sécurité sous-jacents pour établir le contexte de sécurité comme décrit dans les sections 3 et 4. En même temps, pour garantir l'interopérabilité entre les clients et serveurs DNS qui prennent en charge GSS-TSIG, il est EXIGÉ que le mécanisme de sécurité utilisé par le client permette l'utilisation de Kerberos v5 (voir plus d'informations à la Section 9).

3.1 Négociation du contexte

Dans GSS, établir un contexte de sécurité implique de passer des jetons opaques entre le client et le serveur. Le client génère le jeton initial et l'envoie au serveur. Le serveur traite le jeton et si nécessaire, retourne un autre jeton au client. Le client traite ce jeton, et ainsi de suite, jusqu'à l'achèvement de la négociation. Le nombre de fois que le client et le serveur échangent des jetons ne dépend que du mécanisme de sécurité sous-jacent. L'achèvement d'une négociation résulte en une bride de contexte.

L'enregistrement de ressource TKEY [RFC2930] est utilisé comme véhicule de transfert des jetons entre un client et un serveur. L'enregistrement TKEY est un mécanisme général pour établir des clés secrètes à utiliser avec TSIG. Pour plus d'informations, voir la [RFC2930].

3.1.1 Invocation de GSS_Init_sec_context

Pour obtenir le premier jeton à envoyer à un serveur, un client DOIT invoque l'API GSS_Init_sec_context.

Les paramètres d'entrée suivants DOIVENT être utilisés. Le résultat de l'invocation est indiqué avec les valeurs de résultat suivantes. Consulter au paragraphe 2.2.1, "Invocation de GSS_Init_sec_context" de la [RFC2743] les définitions syntaxiques.

ENTRÉES
BRIDE D'ACCRÉDITIF claimant_cred_handle = NUL (NUL spécifie "utiliser par défaut"). Le client PEUT à la place spécifier une autre bride valide pour ses accréditifs.
BRIDE DE CONTEXTE input_context_handle = 0
NOM INTERNE targ_name = "DNS@<target_server_name>"
IDENTIFIANT D'OBJET mech_type = Mécanisme de sécurité sous-jacent choisi par la mise en œuvre. Pour garantir l'interopérabilité des mises en œuvre du mécanisme GSS-TSIG le client DOIT spécifier un mécanisme de sécurité sous-jacent valide qui permette l'utilisation de Kerberos v5 (voir plus d'informations à la Section 9).
CHAINE D'OCTETS input_token = NUL
BOOLÉEN replay_det_req_flag = VRAI
BOOLÉEN mutual_req_flag = VRAI
BOOLÉEN deleg_req_flag = VRAI
BOOLÉEN sequence_req_flag = VRAI
BOOLÉEN anon_req_flag = FAUX
BOOLÉEN integ_req_flag = VRAI
ENTIER lifetime_req = 0 (0 demande une valeur par défaut). Le client PEUT à la place spécifier une autre limite supérieure pour la durée de vie du contexte à établir, en secondes.
OCTET STRING chan_bindings = Tout lien de canal valide spécifié au paragraphe 1.1.6 "Liens de canal" de la [RFC2743]

RÉSULTATS
ENTIER major_status
BRIDE DE CONTEXTE output_context_handle
CHAINE D'OCTETS output_token
BOOLÉEN replay_det_state
BOOLÉEN mutual_state
ENTIER minor_status
IDENTIFIANT D'OBJET mech_type
BOOLÉEN deleg_state
BOOLÉEN sequence_state
BOOLÉEN anon_state
BOOLÉEN trans_state
BOOLÉEN prot_ready_state
BOOLÉEN conf_avail
BOOLÉEN integ_avail
ENTIER lifetime_rec

Si le major_status retourné est réglé à une des erreurs suivantes :

GSS_S_DEFECTIVE_TOKEN	*(jeton GSS défectueux)*
GSS_S_DEFECTIVE_CREDENTIAL	*(accréditifGSS défectueux)*
GSS_S_BAD_SIG (GSS_S_BAD_MIC)	*(mauvaise signature GSS)*
GSS_S_NO_CRED	*(pas d'accréditif GSS)*
GSS_S_CREDENTIALS_EXPIRED	*(accréditifsGSS arrivés à expiration)*
GSS_S_BAD_BINDINGS	*(mauvais lien GSS)*
GSS_S_OLD_TOKEN	*(jeton GSS périmé)*
GSS_S_DUPLICATE_TOKEN	*(jeton GSS dupliqué)*
GSS_S_NO_CONTEXT	*(pas de contexte GSS)*
GSS_S_BAD_NAMETYPE	*(mauvais nom de type GSS)*
GSS_S_BAD_NAME	*(mauvais nom GSS)*
GSS_S_BAD_MECH	*(mauvais mécanisme GSS)*
GSS_S_FAILURE	*(échec GSS)*

le client DOIT alors abandonner l'algorithme et NE DOIT PAS utiliser l'algorithme GSS-TSIG pour établir ce contexte de sécurité. Le présent document ne prescrit pas quel autre mécanisme pourrait être utilisé pour établir un contexte de sécurité. La prochaine fois que ce client a besoin d'établir un contexte de sécurité, il PEUT utiliser l'algorithme GSS-TSIG.

Les valeurs de réussite d'un major_status sont GSS_S_CONTINUE_NEEDED *(il est nécessaire de continuer GSS)* et GSS_S_COMPLETE *(GSS terminé)*. Le code de succès exact est important durant le traitement ultérieur.

Les valeurs de replay_det_state et de mutual_state indiquent respectivement si le paquetage de sécurité assure la détection de

répétition et l'authentification mutuelle. Si le major_status retourné est GSS_S_COMPLETE ET si une de ces valeurs ou les deux est FAUX, le client DOIT abandonner cet algorithme.

Le comportement du client PEUT dépendre d'autres paramètres de résultat selon la politique locale du client.

La bride output_context_handle est unique pour cette négociation et est mémorisée dans le tableau de transpositions du client comme la bride de contexte qui se transpose en nom de cible.

3.1.2 Envoi de l'interrogation TKEY au serveur

Un output_token *(jeton de résultat)* opaque retourné par GSS_Init_sec_context est transmis au serveur dans une demande d'interrogation avec QTYPE=TKEY. Le jeton lui-même sera placé dans un champ Key Data du champ RDATA dans l'enregistrement de ressource TKEY dans la section Enregistrement supplémentaire de l'interrogation. Le nom de propriétaire de l'ensemble d'enregistrement de ressource TKEY objet de l'interrogation et le nom de propriétaire de l'enregistrement de ressource TKEY fourni dans la section Enregistrement supplémentaire DOIT être le même. Ce nom identifie de façon univoque le contexte de sécurité pour le client et le serveur, et donc le client DEVRAIT utiliser une valeur unique au monde, comme décrit dans la [RFC2930]. Pour réaliser l'unicité mondiale, le nom PEUT contenir un UUID/GUID [ISO11578].

Enregistrement TKEY
NOM = chaîne de nom de domaine unique au monde générée par le client (comme décrit dans la [RFC2930])
RDATA
  Nom d'algorithme = gss-tsig
  Mode = 3 (négociation GSS-API – selon la [RFC2930])
  Taille de clé = taille de output_token en octets
  Données de clé = output_token

Les champs restants dans les RDATA TKEY, c'est-à-dire, les champs Inception, Expiration, Error, Autres taille et Données, DOIVENT être réglés conformément à la [RFC2930].

L'interrogation est transmise au serveur.

Note : Si l'invocation originale du client à GSS_Init_sec_context a retourné un major_status autre que GSS_S_CONTINUE_NEEDED ou GSS_S_COMPLETE, le client NE DOIT PAS envoyer alors d'interrogation TKEY. Le comportement du client dans ce cas est décrit au paragraphe 3.1.1 ci-dessus.

3.1.3 Réception de la réponse à l'interrogation TKEY du serveur

À réception de l'interrogation TKEY, le serveur DNS DOIT répondre conformément à la description de la Section 4. Cette section spécifie le comportement du client après réception de la réponse correspondant à son interrogation.

La prochaine étape de traitement dépend de la valeur du major_status provenant de la plus récente invocation effectuée par ce client à GSS_Init_sec_context : soit GSS_S_COMPLETE, soit GSS_S_CONTINUE.

3.1.3.1 Valeur de major_status == GSS_S_COMPLETE

Si la dernière invocation de GSS_Init_sec_context a donnée une valeur de major_status de GSS_S_COMPLETE et si un output_token non NUL a été envoyé au serveur, le composant côté client de la négociation est alors achevée et le client attend la confirmation du serveur.

La confirmation est sous la forme d'une réponse d'interrogation avec RCODE=NOERROR et avec le dernier enregistrement TKEY fourni par le client dans la section réponse de l'interrogation. La réponse DOIT être signée avec un enregistrement TSIG. Noter qu'il est permis au serveur de signer une réponse à une interrogation non signée du client du fait de la modification à la RFC2845 spécifiée au paragraphe 2.2 ci-dessus. La signature dans l'enregistrement TSIG DOIT être vérifiée en utilisant la procédure détaillée à la Section 5, "Envoi et vérification des messages signés". Si la réponse n'est pas signée, OU si la réponse est signée mais la signature est invalide, c'est qu'un attaquant a altéré le message en transit ou a tenté d'envoyer une fausse réponse au client. Dans ce cas, le client PEUT continuer d'attendre une réponse à sa dernière interrogation TKEY jusqu'à ce qu'expire le délai d'envoi de la dernière interrogation TKEY du client. Un tel délai est spécifié par la politique locale du client. C'est une nouvelle option qui permet au client DNS d'accepter des réponses multiples pour un identifiant d'interrogation et d'en choisir une (pas nécessairement la première) sur la base de certains critères.

Si la signature est vérifiée, l'état du contexte est avancé à Contexte établi. On passe au paragraphe 3.2 pour l'usage du contexte de sécurité.

3.1.3.2 Valeur de major_status == GSS_S_CONTINUE_NEEDED

Si la dernière invocation de GSS_Init_sec_context a donné une valeur de major_status de GSS_S_CONTINUE_NEEDED, la négociation n'est alors pas achevée. Le serveur va retourner au client une réponse d'interrogation avec un enregistrement TKEY dans la section Réponse. Si le message d'erreur DNS n'est pas NO_ERROR ou si le champ d'erreur dans l'enregistrement TKEY n'est pas 0 (c'est-à-dire, pas d'erreur) le client DOIT alors abandonner cette séquence de négociation. Le client DOIT supprimer le contexte actif en invoquant GSS_Delete_sec_context en fournissant la context_handle associée. Le client PEUT répéter la séquence de négociation en commençant par l'état non initialisé comme décrit au paragraphe 3.1. Pour empêcher une boucle sans fin, le nombre de tentatives d'établir un contexte de sécurité DOIT être limité à dix ou moins.

Si le message d'erreur DNS est NO_ERROR et si le champ Erreur dans l'enregistrement TKEY est 0 (c'est-à-dire, pas d'erreur) le client DOIT alors passer un jeton spécifié dans le champ Key Data dans l'enregistrement de ressource TKEY à GSS_Init_sec_context en utilisant les mêmes valeurs de paramètres que dans l'invocation précédente sauf les valeurs pour BRIDE DE CONTEXTE input_context_handle et CHAINE D'OCTETS input_token comme décrit ci-dessous.

ENTRÉES
BRIDE DE CONTEXTE input_context_handle = context_handle (c'est la context_handle qui correspond au key_name qui est le nom du propriétaire de l'enregistrement TKEY dans la section Réponse dans la réponse d'interrogation TKEY)

CHAINE D'OCTETS input_token = jeton provenant du champ Key de l'enregistrement TKEY

Selon les valeurs suivantes de RÉSULTAT de GSS_Init_sec_context
ENTIER major_status
CHAINE D'OCTETS output_token
le client DOIT entreprendre une des actions suivantes :

Si le RÉSULTAT major_status est réglé à une des valeurs suivantes :

GSS_S_DEFECTIVE_TOKEN	*(jeton GSS défectueux)*
GSS_S_DEFECTIVE_CREDENTIAL	*(accréditif GSS défectueux)*
GSS_S_BAD_SIG (GSS_S_BAD_MIC)	*(mauvaise signature GSS)*
GSS_S_NO_CRED	*(pas d'accréditif GSS)*
GSS_S_CREDENTIALS_EXPIRED	*(accréditifs GSS expirés)*
GSS_S_BAD_BINDINGS	*(mauvais lien GSS)*
GSS_S_OLD_TOKEN	*(jeton GSS périmé)*
GSS_S_DUPLICATE_TOKEN	*(jeton GSS dupliqué)*
GSS_S_NO_CONTEXT	*(pas de contexte GSS)*
GSS_S_BAD_NAMETYPE	*(mauvais type de nom GSS)*
GSS_S_BAD_NAME	*(mauvais nom GSS)*
GSS_S_BAD_MECH	*(mauvais mécanisme GSS)*
GSS_S_FAILURE	*(échec GSS)*

le client DOIT abandonner cette séquence de négociation. Cela signifie que le client DOIT supprimer un contexte actif en invoquant GSS_Delete_sec_context en fournissant la context_handle associée. Le client PEUT répéter la séquence de négociation en commençant par l'état non initialisé comme décrit au paragraphe 3.1. Pour empêcher une boucle sans fin, le nombre de tentatives d'établir un contexte de sécurité DOIT être limité à dix ou moins.

Si le RÉSULTAT de major_status est GSS_S_CONTINUE_NEEDED OU GSS_S_COMPLETE, le client DOIT alors agir comme décrit ci-dessous.

Si la réponse du serveur était signée, et si le RÉSULTAT major_status est GSS_S_COMPLETE, alors la signature dans l'enregistrement TSIG DOIT être vérifiée en utilisant la procédure détaillée à la Section 5. Si la signature est invalide, le client DOIT alors abandonner cette séquence de négociation. Cela signifie que le client DOIT supprimer un contexte actif en invoquant GSS_Delete_sec_context en fournissant la bride de contexte associée. Le client PEUT répéter la séquence de négociation en commençant à l'état non initialisé comme décrit au paragraphe 3.1. Pour empêcher une boucle sans fin, le nombre de tentatives d'établissement d'un contexte de sécurité DOIT être limité à dix ou moins.

Si major_status est GSS_S_CONTINUE_NEEDED, la négociation n'est pas finie. Le jeton output_token DOIT être passé au serveur dans un enregistrement TKEY en répétant la séquence de négociation commençant au paragraphe 3.1.2. Le client DOIT fixer une limite au nombre de continuations dans une négociation de contexte pour empêcher des boucles sans fin. Une telle limite NE DEVRAIT PAS excéder une valeur de 10.

Si major_status est GSS_S_COMPLETE et si output_token est non NUL, le composant côté client de la négociation est achevé

mais le jeton output_token DOIT être passé au serveur en répétant la séquence de négociation qui commence au paragraphe 3.1.2.

Si major_status est GSS_S_COMPLETE et si le output_token est NUL, la négociation du contexte est terminée. L'état du contexte est avancé à Contexte établi. On passe au paragraphe 3.2 pour l'usage du contexte de sécurité.

3.2 Contexte établi

Lorsque la négociation du contexte est achevée, la bride context_handle DOIT être utilisée pour la génération et la vérification des signatures de transaction.

Les procédures pour l'envoi et la réception des messages signés sont décrites dans la Section 5, "Envoi et vérification des messages signés".

3.2.1 Terminaison d'un contexte

Lorsque le client n'a pas l'intention de continuer à utiliser le contexte de sécurité établi, le client DEVRAIT supprimer un contexte actif en invoquant GSS_Delete_sec_context en fournissant la bride de contexte associée, ET le client DEVRAIT supprimer le contexte établi sur le serveur DNS en utilisant l'enregistrement de ressource TKEY avec le champ Mode réglé à 5, c'est-à-dire, "suppression de clé" [RFC2930].

4. Détails du protocole du serveur

Comme sur le côté client, le résultat d'une négociation de contexte réussie est une bride de contexte utilisée dans la future génération et vérification des signatures de transaction.

Un serveur PEUT gérer plusieurs contextes avec plusieurs clients. Les clients identifient leurs contextes en fournissant un nom de clé dans leur demande. Le serveur conserve une transposition des noms de clé en brides :

(nom_de_clé, bride_de_contexte)

4.1 Négociation du contexte

Un serveur DOIT reconnaître les interrogations TKEY comme des messages de négociation de contexte de sécurité.

4.1.1 Réception de l'interrogation TKEY du client

À réception d'une interrogation avec QTYPE = TKEY, le serveur DOIT examiner si les champs de Mode et de Nom d'algorithme de l'enregistrement TKEY dans la section des enregistrements supplémentaires du message contiennent les valeurs respectivement de 3 et de gss-tsig. Si c'est le cas, le tableau de transposition (nom_de_clé, bride_de_contexte) est examiné à la recherche du nom_de_clé correspondant au nom du propriétaire du nom de l'enregistrement TKEY dans la section d'enregistrements supplémentaires de l'interrogation. Si le nom est trouvé dans le tableau et si le contexte de sécurité pour ce nom est établi et n'est pas arrivé à expiration, le serveur DOIT alors répondre à l'interrogation avec une erreur BADNAME dans le champ d'erreur de TKEY. Si le nom est trouvé dans le tableau et si le contexte de sécurité n'est pas établi, la bride de contexte correspondante est utilisée dans les opérations GSS ultérieures. Si le nom est trouvé mais si le contexte de sécurité est arrivé à expiration, le serveur supprime alors ce contexte de sécurité, comme décrit au paragraphe 4.2.1, et interprète cette interrogation comme le début d'une nouvelle négociation de contexte de sécurité et effectue les opérations décrites aux paragraphes 4.1.2 et 4.1.3. Si le nom n'est pas trouvé, le serveur interprète alors cette interrogation comme le début d'une nouvelle négociation de contexte de sécurité et effectue les opérations décrites aux paragraphes 4.1.2 et 4.1.3.

4.1.2 Invocation de GSS_Accept_sec_context

Le serveur effectue son côté d'une négociation de contexte en invoquant GSS_Accept_sec_context. Les paramètres d'entrée suivants DOIVENT être utilisés. Le résultat de l'invocation est indiqué par les valeurs de résultat ci-dessous. Consulter le paragraphe 2.2.2 "Invocation de GSS_Accept_sec_context" de la [RFC2743] pour les définitions syntaxiques.

ENTRÉES
BRIDE DE CONTEXTE input_context_handle = 0 si c'est une nouvelle négociation, context_handle correspondant à

key_name si c'est une négociation en cours
CHAINE D'OCTETS input_token = jeton spécifié dans le champ Key du RR TKEY (de la section Enregistrements supplémentaires de l'interrogation du client)
BRIDE D'ACCREDITIF acceptor_cred_handle = NUL (NUL spécifie "utiliser la valeur par défaut"). Le serveur PEUT à la place spécifier une autre bride valide vers ses accréditifs.
CHAINE D'OCTETS chan_bindings = Tout lien de canal valide comme spécifié au paragraphe 1.1.6 "Liens de canal" de la [RFC2743]

RÉSULTATS
ENTIER major_status
BRIDE_DE_COBTEXTE output_context_handle
CHAINE D'OCTETS output_token
ENTIER minor_status
NOM INTERNE src_name
IDENTIFIANT d'OBJET mech_type
BOOLÉEN deleg_state
BOOLÉEN mutual_state
BOOLÉEN replay_det_state
BOOLÉEN sequence_state
BOOLÉEN anon_state
BOOLÉEN trans_state
BOOLÉEN prot_ready_state
BOOLÉEN conf_avail
BOOLÉEN integ_avail
ENTIER lifetime_rec
BRIDE_DE_CONTEXTE delegated_cred_handle

Si c'est la première invocation de GSS_Accept_sec_context dans une nouvelle négociation, output_context_handle est alors mémorisé dans le tableau de transposition de clé du serveur comme la context_handle qui se transpose en le nom de l'enregistrement TKEY.

4.1.3 Envoi de la réponse à l'interrogation TKEY au client

Le serveur DOIT répondre au client par une réponse d'interrogation TKEY avec un RCODE = NOERROR, qui contient un enregistrement TKEY dans la section Réponse.

Si le RÉSULTAT major_status est une des erreurs suivantes, le champ Erreur dans l'enregistrement TKEY est réglé à BADKEY.

GSS_S_DEFECTIVE_TOKEN	*(jeton GSS défectueux)*
GSS_S_DEFECTIVE_CREDENTIAL	*(accréditif GSS défectueux)*
GSS_S_BAD_SIG (GSS_S_BAD_MIC)	*(mauvaise signature GSS)*
GSS_S_DUPLICATE_TOKEN	*(jeton GSS dupliqué)*
GSS_S_OLD_TOKEN	*(jeton GSS périmé)*
GSS_S_NO_CRED	*(pas d'accréditif GSS)*
GSS_S_CREDENTIALS_EXPIRED	*(accréditifs GSS expirés)*
GSS_S_BAD_BINDINGS	*(mauvais liens GSS)*
GSS_S_NO_CONTEXT	*(pas de contexte GSS)*
GSS_S_BAD_MECH	*(mauvais mécanisme GSS)*
GSS_S_FAILURE	*(échec GSS)*

Si le RÉSULTAT major_status est réglé à GSS_S_COMPLETE ou à GSS_S_CONTINUE_NEEDED, le serveur DOIT alors agir comme décrit ci-dessous.

Si major_status est GSS_S_COMPLETE, la composante serveur de la négociation est terminée. Si le output_token est non NUL, il DOIT alors être retourné au client dans un champ Key Data des RDATA dans TKEY. Le champ Erreur dans l'enregistrement TKEY est réglé à NOERROR. Le message DOIT être signé avec un enregistrement TSIG comme décrit à la Section 5. Noter qu'il est permis au serveur de signer une réponse à l'interrogation non signée du client grâce à la modification de la RFC2845 spécifiée au paragraphe 2.2. L'état du contexte est avancé à Contexte établi. Le paragraphe 4.2 expose l'usage du contexte de sécurité.

Si major_status est GSS_S_COMPLETE et si output_token est NUL, l'enregistrement TKEY reçu du client DOIT alors être

retourné dans la section Réponse de la réponse. Le message DOIT être signé avec un enregistrement TSIG comme décrit à la Section 5. Noter qu'il est permis au serveur de signer une réponse à l'interrogation non signée du client grâce à la modification à la RFC2845 spécifiée au paragraphe 2.2. L'état du contexte est avancé à Contexte établi. Le paragraphe 4.2 expose l'usage du contexte de sécurité.

Si major_status est GSS_S_CONTINUE_NEEDED, la composante serveur de la négociation n'est pas encore terminée. Le serveur répond à l'interrogation TKEY par une réponse à interrogation standard, en plaçant dans la section Réponse un enregistrement TKEY contenant le output_token dans le champ Key Data de RDATA. Le champ Erreur dans l'enregistrement TKEY est réglé à NOERROR. Le serveur DOIT limiter le nombre de fois qu'il est permis à un certain contexte de se répéter, pour empêcher des boucles sans fin. Une telle limite NE DEVRAIT PAS excéder 10.

Dans tous les cas, sauf si major_status est GSS_S_COMPLETE et output_token est NUL, les autres champs de l'enregistrement TKEY DOIVENT contenir les valeurs suivantes :

NOM = key_name
RDATA
Nom d'algorithme = gss-tsig
Mode = 3 (négociation GSS-API – selon la [RFC2930])
Taille de clé = taille de output_token en octets

Les champs restants dans le RDATA de TKEY, c'est-à-dire les champs Inception, Expiration, Erreur, Autre taille et Données, DOIVENT être réglés conformément à la [RFC2930].

4.2 Contexte établi

Lorsque la négociation du contexte est achevée, la bride context_handle est utilisée pour la génération et la vérification des signatures de transactions. La bride est valide pour une durée limitée déterminée par le mécanisme de sécurité sous-jacent. Un serveur PEUT terminer unilatéralement un contexte à tout moment (voir au paragraphe 4.2.1).

Un serveur DEVRAIT limiter la quantité de mémoire utilisée pour mettre en antémémoire les contextes établis.

Les procédures d'envoi et de réception des messages signés figurent à la Section 5 "Envoi et vérification des messages signés".

4.2.1 Terminaison d'un contexte

Un serveur peut terminer tout contexte établi à tout moment. Le serveur PEUT indiquer au client que le contexte est en cours de suppression en incluant un RR TKEY dans une réponse avec le champ Mode réglé à 5, c'est-à-dire, "suppression de clé" [RFC2930]. Un contexte actif est supprimé par l'invocation de GSS_Delete_sec_context en fournissant la bride de contexte associée.

5. Envoi et vérification des messages signés

5.1 Envoi d'un message signé – invocation de GSS_GetMIC

La procédure pour l'envoi d'un message protégé par une signature est spécifiée dans la [RFC2845]. Les données à passer au sous-programme de signature incluent la totalité du message DNS avec les variables spécifiques de TSIG ajoutées. Pour le format exact, voir la [RFC2845]. Pour le présent protocole, on utilise les valeurs de variable TSIG suivantes :

Enregistrement TSIG
    NOM = le key_name qui identifie ce contexte
RDATA
    Nom d'algorithme = gss-tsig

On alloue aux champs restants dans les RDATA de TSIG les valeurs appropriées décrites dans la [RFC2845].

La signature est générée en invoquant GSS_GetMIC. Les paramètres d'entrée suivants DOIVENT être utilisés. Le résultat de l'invocation est indiqué avec les valeurs de résultat spécifiées ci-dessous. Consulter le paragraphe 2.3.1 "Invocation de GSS_GetMIC" de la [RFC2743] pour les définitions syntaxiques.

ENTRÉES
BRIDE DE CONTEXTE context_handle = context_handle pour key_name
CHAINE D'OCTETS message = message sortant plus variables TSIG (selon la [RFC2845])
ENTIER qop_req = 0 (0 demande une valeur par défaut). L'invocateur PEUT à la place spécifier une autre valeur valide (pour les détails, voir le paragraphe 1.2.4 de la [RFC2743])

RÉSULTATS
ENTIER major_status
ENTIER minor_status
CHAINE D'OCTETS per_msg_token

Si major_status est GSS_S_COMPLETE, la génération de signature a alors réussi. La signature dans per_msg_token est insérée dans le champ Signature de l'enregistrement TSIG et le message est transmis.

Si major_status est GSS_S_CONTEXT_EXPIRED, GSS_S_CREDENTIALS_EXPIRED ou GSS_S_FAILURE, l'invocateur DOIT supprimer le contexte de sécurité, retourner à l'état non initialisé et DEVRAIT négocier un nouveau contexte de sécurité, comme décrit au paragraphe 3.1

Si major_status est GSS_S_NO_CONTEXT, l'invocateur DOIT supprimer l'entrée pour key_name du tableau de transposition (target_name, key_name, context_handle) retourner à l'état non initialisé et DEVRAIT négocier un nouveau contexte de sécurité, comme décrit au paragraphe 3.1

Si major_status est GSS_S_BAD_QOP, l'invocateur DEVRAIT répéter l'invocation GSS_GetMIC avec une valeur de QOP permise. Le nombre de telles répétitions DOIT être limité pour empêcher des boucles sans fin.

5.2 Vérification du message signé – invocation de GSS_VerifyMIC

La procédure pour vérifier un message protégé par une signature est spécifiée dans la [RFC2845].

Le NOM de l'enregistrement TSIG détermine quelle context_handle se transpose en le contexte qui DOIT être utilisé pour vérifier la signature. Si le NOM ne se transpose pas en un contexte établi, le serveur DOIT envoyer une réponse d'erreur standard TSIG au client indiquant BADKEY dans le champ TSIG Erreur (comme décrit dans la [RFC2845]).

Pour l'algorithme GSS, une signature est vérifiée en utilisant GSS_VerifyMIC :

ENTRÉES
BRIDE DE CONTEXTE context_handle = context_handle pour key_name
CHAINE D'OCTETS message = message entrant plus variables TSIG (selon la [RFC2845])
CHAINE D'OCTETS per_msg_token = Champ Signature du RR TSIG

RÉSULTATS
ENTIER major_status
ENTIER minor_status
ENTIER qop_state

Si major_status est GSS_S_COMPLETE, la signature est authentique et le message a été livré intact. Selon la [RFC2845], les valeurs de temporisateur de l'enregistrement TSIG DOIVENT aussi être valides avant de considérer le message comme authentique. L'invocateur NE DOIT PAS agir sur la demande ou la réponse dans le message tant que ces vérifications ne sont pas faites.

Lorsque un serveur traite une demande d'un client, le serveur DOIT envoyer une réponse d'erreur standard TSIG au client, indiquant BADKEY dans le champ Erreur TSIG comme décrit dans la [RFC2845], si major_status est réglé à une des valeurs suivantes :

GSS_S_DEFECTIVE_TOKEN	*(jeton GSS défectueux)*
GSS_S_BAD_SIG (GSS_S_BAD_MIC)	*(mauvaise signature GSS)*
GSS_S_DUPLICATE_TOKEN	*(jeton GSS dupliqué)*
GSS_S_OLD_TOKEN	*(jeton GSS périmé)*
GSS_S_UNSEQ_TOKEN	*(jeton GSS pas en séquence)*
GSS_S_GAP_TOKEN	*(trou dans les jetons GSS)*
GSS_S_CONTEXT_EXPIRED	*(contexte GSS expiré)*
GSS_S_NO_CONTEXT	*(pas de contexte GSS)*

GSS_S_FAILURE *(échec GSS)*

Si les valeurs de temporisateur de l'enregistrement TSIG sont invalides, le message NE DOIT PAS être considéré comme authentique. Si cette vérification d'erreur échoue lorsque un serveur traite la demande d'un client, la réponse d'erreur appropriée DOIT être envoyée au client conformément à la [RFC2845].

6. Exemple d'utilisation de l'algorithme GSS-TSIG

Cette Section donne un exemple où un client, client.example.com, et un serveur, server.example.com, établissent un contexte de sécurité conformément à l'algorithme décrit ci-dessus.

I. Le client initialise la négociation de contexte de sécurité
Pour établir un contexte de sécurité avec un serveur, server.example.com, le client invoque GSS_Init_sec_context avec les paramètres suivants. (Noter que certains paramètres ENTRÉE et RÉSULTAT ne sont pas critiques pour cet algorithme et ne sont pas décrits dans cet exemple.)

BRIDE DE CONTEXTE input_context_handle = 0
NOM INTERNE targ_name = "DNS@server.example.com"
CHAINE D'OCTETS input_token = NUL
BOOLÉEN replay_det_req_flag = VRAI
BOOLÉEN mutual_req_flag = VRAI

Les paramètres de RÉSULTATS retournés par GSS_Init_sec_context incluent :
ENTIER major_status = GSS_S_CONTINUE_NEEDED
BRIDE DE CONTEXTE output_context_handle context_handle
CHAINE D'OCTETS output_token output_token
BOOLÉEN replay_det_state = VRAI
BOOLÉEN mutual_state = VRAI

Le client vérifie que les valeurs replay_det_state et mutual_state sont VRAI. Comme le major_status est GSS_S_CONTINUE_NEEDED, qui est une valeur de succès de RÉSULTAT de major_status, le client mémorise la context_handle qui se transpose en "DNS@server.example.com" et passe à l'étape suivante.

II. Le client envoie une interrogation avec QTYPE = TKEY au serveur
Le client envoie une interrogation avec QTYPE = TKEY pour une chaîne de nom de domaine unique au monde générée par le client, 789.client.example.com.server.example.com. L'interrogation contient un enregistrement TKEY dans sa section Enregistrements supplémentaires avec les champs suivants. (Noter que certains champs non spécifiques de cet algorithme ne sont pas spécifiés.)

NOM = 789.client.example.com.server.example.com.
RDATA
Nom d'algorithme = gss-tsig
Mode = 3 (négociation GSS-API – selon la [RFC2930])
Taille de clé = Taille de output_token en octets
Données de clé = output_token

Après que le key_name 789.client.example.com.server.example.com. a été généré, il est mémorisé dans le tableau de transposition (target_name, key_name, context_handle) du client.

III. Le serveur reçoit une interrogation avec QTYPE = TKEY
Lorsque le serveur reçoit une interrogation avec QTYPE = TKEY, il vérifie que les champs Mode et Algorithme dans l'enregistrement TKEY dans la section Enregistrements supplémentaires de l'interrogation sont réglés à 3 et "gss-tsig" respectivement. Il trouve que le key_name 789.client.example.com.server.example.com. ne figure pas dans son tableau de transposition (key_name, context_handle).

IV. Le serveur invoque GSS_Accept_sec_context
Pour continuer la négociation de contexte de sécurité, le serveur invoque GSS_Accept_sec_context avec les paramètres suivants. (Noter que certains paramètres ENTRÉE et RÉSULTAT qui ne sont pas critiques pour cet algorithme ne sont pas décrits dans l'exemple.)

ENTRÉES

BRIDE DE CONTEXTE input_context_handle = 0
CHAINE D'OCTETS input_token = jeton spécifié dans le champ Key provenant du RR TKEY (de la section Enregistrements supplémentaires de l'interrogation du client)

Les paramètres de RÉSULTATS retournés par le GSS_Accept_sec_context incluent :
ENTIER major_status = GSS_S_CONTINUE_NEEDED
BRIDE DE CONTEXTE output_context_handle context_handle
CHAINE D'OCTETS output_token output_token

Le serveur mémorise la transposition de 789.client.example.com.server.example.com. en RÉSULTAT context_handle dans son tableau de transposition (key_name, context_handle).

V. Le serveur répond à l'interrogation TKEY
Depuis le major_status = GSS_S_CONTINUE_NEEDED dans la dernière invocation du serveur à GSS_Accept_sec_context, le serveur répond à l'interrogation TKEY en plaçant dans la section Réponse un enregistrement TKEY qui contient un output_token dans le champ Key Data de RDATA. Le champ Erreur dans l'enregistrement TKEY est réglé à 0. Le RCODE dans la réponse à l'interrogation est réglé à NOERROR.

VI. Le client traite le jeton retourné par le serveur
Lorsque le client reçoit la réponse à l'interrogation TKEY du serveur, il invoque GSS_Init_sec_context avec les paramètres suivants. (Noter que certains paramètres ENTRÉE et RÉSULTAT qui ne sont pas critiques pour cet algorithme ne sont pas décrits dans l'exemple.)

BRIDE DE CONTEXTE input_context_handle = la context_handle mémorisée dans l'entrée du tableau de transposition du client (DNS@server.example.com., 789.client.example.com.server.example.com., context_handle)
NOM NTERNE targ_name = "DNS@server.example.com"
CHAINE D'OCTETS input_token = jeton du champ Key de l'enregistrement TKEY provenant de la section Réponse de la réponse du serveur
BOOLÉEN replay_det_req_flag = VRAI
BOOLÉEN mutual_req_flag = VRAI

Les paramètres de RÉSULTAT retournés par GSS_Init_sec_context incluent :
ENTIER major_status = GSS_S_COMPLETE
BRIDE DE CONTEXTE output_context_handle = context_handle
CHAINE D'OCTETS output_token = output_token
BOOLÉEN replay_det_state = VRAI
BOOLÉEN mutual_state = VRAI

Comme le major_status est réglé à GSS_S_COMPLETE, le contexte de sécurité côté client est établi, mais comme le output_token n'est pas NUL, le client DOIT envoyer une interrogation TKEY au serveur comme décrit ci-dessous.

VII. Le client envoie une interrogation avec QTYPE = TKEY au serveur
Le client envoie au serveur une interrogation TKEY pour le nom 789.client.example.com.server.example.com. L'interrogation contient un enregistrement TKEY dans la section Enregistrements supplémentaires avec les paramètres suivants. (Noter que certains paramètres ENTRÉE et RÉSULTAT qui ne sont pas critiques pour cet algorithme ne sont pas décrits dans l'exemple.)

NOM = 789.client.example.com.server.example.com.
RDATA
Nom d'algorithme = gss-tsig
Mode = 3 (négociation GSS-API – selon la [RFC2930])
Taille de clé = Taille de output_token en octets
Données de clé = output_token

VIII. Le serveur reçoit une interrogation TKEY
Lorsque le serveur reçoit une interrogation TKEY, il vérifie que les champs Mode et Algorithme dans l'enregistrement TKEY dans la section Enregistrements supplémentaires de l'interrogation sont réglés respectivement à 3 et gss-tsig. Il trouve que le key_name 789.client.example.com.server.example.com. figure dans son tableau de transposition (key_name, context_handle).

IX. Le serveur invoque GSS_Accept_sec_context
Pour continuer la négociation de contexte de sécurité, le serveur invoque GSS_Accept_sec_context avec les paramètres suivants. (Noter que certains paramètres ENTRÉE et RÉSULTAT qui ne sont pas critiques pour cet algorithme ne sont pas décrits dans l'exemple)

ENTRÉES
BRIDE DE CONTEXTE input_context_handle = context_handle de l'entrée (789.client.example.com.server.example.com., context_handle) dans le tableau de transposition du serveur
CHAINE D'OCTETS input_token = jeton spécifié dans le champ Key du RR TKEY (provenant de la section Enregistrements supplémentaires de l'interrogation du client)

Les paramètres de RÉSULTAT retournés par GSS_Accept_sec_context incluent :
ENTIER major_status = GSS_S_COMPLETE
BRIDE DE CONTEXTE output_context_handle = context_handle
CHAINE D'OCTETS output_token = NUL

Comme major_status = GSS_S_COMPLETE, le contexte de sécurité sur le côté serveur est établi, mais le serveur doit encore répondre à l'interrogation TKEY du client, comme décrit ci-dessous. L'état du contexte de sécurité est avancé à Contexte établi.

X. Le serveur répond à l'interrogation TKEY
Comme le major_status = GSS_S_COMPLETE dans la dernière invocation du serveur à GSS_Accept_sec_context et le output_token est NUL, le serveur répond à l'interrogation TKEY en plaçant dans la section Réponse un enregistrement TKEY qui a été envoyé par le client dans la section Enregistrements supplémentaires de la dernière interrogation TKEY du client. De plus, ce serveur place un enregistrement TSIG dans la section Enregistrements supplémentaires de sa réponse. Le serveur invoque GSS_GetMIC pour générer une signature à inclure dans l'enregistrement TSIG. Le serveur spécifie les paramètres GSS_GetMIC d'ENTRÉE suivants :

BRIDE DE CONTEXTE context_handle = context_handle provenant de l'entrée (789.client.example.com.server.example.com., context_handle) du tableau de transposition du serveur
CHAINE D'OCTETS message = message sortant plus variables TSIG (comme décrites dans la [RFC2845])

Les paramètres de RÉSULTAT retournés par GSS_GetMIC incluent :
ENTIER major_status = GSS_S_COMPLETE
CHAINE D'OCTETS per_msg_token

Le champ Signature dans l'enregistrement TSIG est réglé à per_msg_token.

XI. Le client traite le jeton retourné par le serveur
Le client reçoit la réponse à l'interrogation TKEY provenant du serveur. Comme le major_status était GSS_S_COMPLETE dans la dernière invocation du client à GSS_Init_sec_context, le client vérifie que la réponse du serveur est signée. Pour valider la signature, le client invoque GSS_VerifyMIC avec les paramètres suivants :

ENTRÉES
BRIDE DE CONTEXTE context_handle = context_handle for 789.client.example.com.server.example.com. key_name
CHAINE D'OCTETS message = message entrant plus variables TSIG (comme décrit dans la [RFC2845])
CHAINE D'OCTETS per_msg_token = Champ Signature provenant du RR TSIG inclus dans la réponse à l'interrogation du serveur

Comme le paramètre de RÉSULTAT major_status = GSS_S_COMPLETE, la signature est validée, la négociation de sécurité est achevée et l'état du contexte de sécurité est avancé à Contexte établi. Ce client et ce serveur vont utiliser le contexte de sécurité établi pour signer et valider les signatures lors de l'échange de paquets entre eux jusqu'à l'expiration du contexte.

7. Considérations sur la sécurité

Le présent document décrit un protocole pour la sécurité du DNS qui utilise GSS-API. La sécurité apportée par ce protocole n'est pas plus effective que la sécurité assurée par les mécanismes GSS sous-jacents.

Toutes les considérations sur la sécurité des [RFC2743], [RFC2845], et [RFC2930] s'appliquent au protocole décrit dans le présent document.

8. Considérations relatives à l'IANA

L'IANA a réservé le nom d'algorithme TSIG gss-tsig pour être utilisé dans le champs Algorithme des enregistrements de ressource TKEY et TSIG. Ce nom d'algorithme se réfère à l'algorithme décrit dans le présent document. L'exigence d'avoir ce

nom enregistré auprès de l'IANA est spécifiée dans la [RFC2845].

9. Conformité

La GSS API qui utilisé SPNEGO [RFC2478] donne le maximum de souplesse au choix des mécanismes de sécurité sous-jacents qui permettent la négociation du contexte de sécurité. La GSS API qui utilisé SPNEGO [RFC2478] permet au client et au serveur de négocier et choisir au vol de tels mécanismes de sécurité sous-jacents. Pour assurer une telle souplesse, les clients et serveurs du DNS DEVRAIENT spécifier SPNEGO mech_type dans leurs appels GSS API. En même temps, afin de garantir l'interopérabilité entre clients et serveurs DNS qui prennent en charge GSS-TSIG, il est exigé que :

- les serveurs DNS spécifient le mech_type SPNEGO
- les GSS API invoquées par le client DNS prennent en charge Kerberos v5
- les GSS API invoquées par le serveur DNS prennent en charge SPNEGO [RFC2478] et Kerberos v5.

En plus de cela, les GSS API utilisées par le client et le serveur DNS PEUVENT aussi prendre en charge d'autres mécanismes de sécurité sous-jacents.

10. Déclaration de propriété intellectuelle

L'IETF ne prend pas position sur la validité et la portée de tout droit de propriété intellectuelle ou autres droits qui pourrait être revendiqués au titre de la mise en œuvre ou l'utilisation de la technologie décrite dans le présent document ou sur la mesure dans laquelle toute licence sur de tels droits pourrait être ou n'être pas disponible ; pas plus qu'elle ne prétend avoir accompli aucun effort pour identifier de tels droits. Les informations sur les procédures de l'ISOC au sujet des droits dans les documents de l'ISOC figurent dans les BCP 78 et BCP 79.

Des copies des dépôts d'IPR faites au secrétariat de l'IETF et toutes assurances de disponibilité de licences, ou le résultat de tentatives faites pour obtenir une licence ou permission générale d'utilisation de tels droits de propriété par ceux qui mettent en œuvre ou utilisent la présente spécification peuvent être obtenues sur répertoire en ligne des IPR de l'IETF à http://www.ietf.org/ipr.

L'IETF invite toute partie intéressée à porter son attention sur tous copyrights, licences ou applications de licence, ou autres droits de propriété qui pourraient couvrir les technologies qui peuvent être nécessaires pour mettre en œuvre la présente norme. Prière d'adresser les informations à l'IETF à ietf-ipr@ietf.org.

11. Remerciements

Les auteurs du présent document tiennent à remercier les personnes suivantes de leur contribution à cette spécification : Chuck Chan, Mike Swift, Ram Viswanathan, Olafur Gudmundsson, Donald E. Eastlake 3rd, et Erik Nordmark.

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.

[RFC2478] E. Baize, D. Pinkas, "Mécanisme de négociation GSS-API simple et protégé", décembre 1998. (*Obsolète, voir* RFC4178) *(PS)*

[RFC2743] J. Linn, "Interface générique de programme d'application de service de sécurité, version 2, mise à jour 1", janvier 2000. (*MàJ par* RFC5554)

[RFC2845] P. Vixie et autres, "Authentification de transaction de clé secrète pour DNS (TSIG)", mai 20000 (*MàJ par* RFC3645) *(P.S.)*

[RFC2930] D. Eastlake 3rd, "Établissement de clés secrètes pour le DNS (TKEY RR)", septembre 2000. *(P.S.)*

12.2 Références informatives

[ISO11578] ISO/IEC 11578:1996 "Information technology, Open Systems Interconnection, Remote Procedure Call", http://www.iso.ch/cate/d2229.html .

[RFC1034] P. Mockapetris, "Noms de domaines - Concepts et facilités", STD 13, novembre 1987.

[RFC1035] P. Mockapetris, "Noms de domaines – Mise en œuvre et spécification", STD 13, novembre 1987.

[RFC1964] J. Linn, "Mécanisme GSS-API de Kerberos version 5", juin 1996. (*MàJ par* RFC4121 et RFC6649)

[RFC2025] C. Adams, "Mécanisme simple de GSS-API à clé publique (SPKM)", octobre 1996. *(P.S.)*

[RFC2137] D. Eastlake 3rd, "Mise à jour dynamique sécurisée du système de noms de domaines", avril 1997. (*Obsolète, voir* RFC3007) *(MàJ* RFC1035) *(P.S.)*

[RFC2535] D. Eastlake, 3rd, "Extensions de sécurité du système des noms de domaines", mars 1999. (*Obsolète, voir* RFC4033, RFC4034, RFC4035) *(P.S.)*

13. Adresse des auteurs

Stuart Kwan
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
USA
mél : skwan@microsoft.com

Praerit Garg
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
USA
mél : praeritg@microsoft.com

James Gilroy
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
USA
mél : jamesg@microsoft.com

Levon Esibov
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
USA
mél : levone@microsoft.com

Randy Hall
Lucent Technologies
400 Lapp Road
Malvern PA 19355
USA
mél : randyhall@lucent.com

Jeff Westhead
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
USA
mél : jwesth@microsoft.com

14. Déclaration complète de droits de reproduction

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