Groupe de travail Réseau
Request for Comments : 1964
Catégorie : En cours de normalisation

J. Linn, OpenVision Technologies
juin 1996
Traduction Claude Brière de L'Isle

Mécanisme de GSS-API de Kerberos Version 5

Statut du présent mémoire
Le présent document spécifie un protocole Internet en cours de normalisation pour la communauté de l'Internet. Il appelle à la discussion et à des suggestions pour son amélioration. Prière de se référer à l'édition actuelle des "Normes officielles des protocoles de l'Internet" (STD 1) pour connaître l'état de normalisation et le statut de ce protocole. La distribution du présent mémoire n'est soumise à aucune restriction.

Résumé
La présente spécification définit les protocoles, procédures, et conventions que doivent employer les homologues qui mettent en œuvre l'interface de programme d'application de service générique de sécurité (GSS-API, *Generic Security Service Application Program Interface*) comme spécifié dans la [RFC1508] et la [RFC1509] lors de l'utilisation de la technologie de Kerberos version 5 comme spécifié dans la [RFC1510].

Remerciements
Beaucoup des matériaux du présent mémoire se fondent sur des documents de travail élaborés par John Wray de Digital Equipment Corporation et sur des discussions, des activités de mise en œuvre et d'essais d'interopérabilité dans lesquels se sont impliqués Marc Horowitz, Ted Ts'o, et John Wray. Des remerciements particuliers sont dus à chacun d'eux pour leurs contributions au développement et à la disponibilité de la prise en charge de GSS-API dans la base du code de Kerberos version 5.

Table des Matières

1. Formats de jeton

Cette section expose les caractéristiques visibles par le protocole du mécanisme de GSS-API à mettre en œuvre par dessus la technologie de sécurité de Kerberos v5 selon les [RFC1508] et [RFC1510] ; elle définit les éléments de protocole pour l'interopérabilité et est indépendante des liens de langage selon la [RFC1509].

Les jetons transférés entre les homologues de GSS-API (pour les besoins de la gestion du contexte de sécurité et la protection par message) sont définis. Les éléments de données échangés entre une mise en œuvre de point d'extrémité de GSS-API et le centre de distribution de clés Kerberos ne sont pas spécifiques de l'utilisation de GSS-API et sont donc définis dans la [RFC1510] plutôt que dans la présente spécification.

Pour la prise en charge de l'expérimentation, des essais, et de l'évolution en cours de la spécification, le mécanisme de GSS-API de Kerberos v5 tel que défini dans le présent mémoire et ses successeurs éventuels, sera identifié par l'identifiant d'objet suivant, tel que défini dans la [RFC1510], jusqu'à ce que la spécification soit avancée au niveau de RFC de proposition de norme :

{iso(1), org(3), dod(5), internet(1), security(5), kerberosv5(2)}

Lorsqu'il aura avancé jusqu'au niveau de proposition de norme, le mécanisme de GSS-API Kerberos v5 sera identifié par un identifiant d'objet ayant la valeur suivante :

{iso(1) member-body(2) United States(840) mit(113554) infosys(1) gssapi(2) krb5(2)}

1.1 Jetons d'établissement de contexte

Selon la [RFC1508], Appendice B, le jeton initial d'établissement de contexte sera inclus dans le tramage qui suit :

```
InitialContextToken ::= [APPLICATION 0] SEQUENCE IMPLICITE {
   thisMech            MechType                  -- MechType est un IDENTIFIANT D'OBJET représentant
                                                    "Kerberos v5"
   innerContextToken   TOUT DÉFINI PAR thisMech  -- contenu spécifique du mécanisme ; l'usage de l'ASN.1 au
                                                    sein de innerContextToken n'est pas exigé.
   }
```

Le innerContextToken du jeton de contexte initial va consister en un message Kerberos v5 KRB_AP_REQ, précédé par un champ Identifiant de jeton (token-id) (TOK_ID) de deux octets, qui devra contenir la valeur 01 00.

La trame de GSS-API ci-dessus devra être appliquée à tous les jetons émis par le mécanisme GSS-API Kerberos v5, y compris KRB_AP_REP, KRB_ERROR, de suppression de contexte, et les jetons par message, et non seulement le jeton initial dans une séquence d'établissement de contexte. Bien que non exigé par la [RFC1508], cela permet aux mises en œuvre d'effectuer une vérification d'erreurs améliorée. Le champ innerContextToken des jetons d'établissement de contexte pour le mécanisme de GSS-API de Kerberos v5 va contenir un message Kerberos (KRB_AP_REQ, KRB_AP_REP ou KRB_ERROR) précédé par un champ TOK_ID de deux octets contenant 01 00 pour les messages KRB_AP_REQ, 02 00 pour les messages KRB_AP_REP et 03 00 pour les messages KRB_ERROR.

1.1.1 Jeton initial

La syntaxe pertinente de KRB_AP_REQ (d'après la [RFC1510]) est la suivante :

```
AP-REQ ::= [APPLICATION 14] SEQUENCE {
      pvno [0]          ENTIER,                 -- indique la version 5
      msg-type [1]      ENTIER,                 -- indique KRB_AP_REQ
      ap-options[2]     APOptions,
      ticket[3]         Ticket,
      authenticator[4]  EncryptedData
}

APOptions ::= CHAINE BINAIRE {
      réservé           (0),
      use-session-key   (1),
      mutual-required   (2)
}

Ticket ::= [APPLICATION 1] SEQUENCE {
      tkt-vno [0]       ENTIER,                 -- indique la version 5
      realm [1]         Realm,
      sname [2]         PrincipalName,
      enc-part [3]      EncryptedData
}

-- Partie chiffrée de ticket
EncTicketPart ::= [APPLICATION 3] SEQUENCE {
      flags[0]              TicketFlags,
      key[1]                EncryptionKey,
      crealm[2]             Realm,
      cname[3]              PrincipalName,
      transited[4]          TransitedEncoding,
      authtime[5]           KerberosTime,
```

```
    starttime[6]            KerberosTime FACULTATIF,
    endtime[7]              KerberosTime,
    renew-till[8]           KerberosTime FACULTATIF,
    caddr[9]                HostAddresses FACULTATIF,
    authorization-data[10]  AuthorizationData FACULTATIF
}

-- authentifiant non chiffré
Authenticator ::= [APPLICATION 2] SEQUENCE  {
    authenticator-vno[0]    ENTIER,
    crealm[1]               Realm,
    cname[2]                PrincipalName,
    cksum[3]                Checksum FACULTATIF,
    cusec[4]                ENTIER,
    ctime[5]                KerberosTime,
    subkey[6]               EncryptionKey FACULTATIF,
    seq-number[7]           ENTIER FACULTATIF,
    authorization-data[8]   AuthorizationData FACULTATIF
}
```

Pour les besoins de la présente spécification, l'authentifiant devra inclure le numéro de séquence facultatif, et le champ Somme de contrôle *(checksum)* devra être utilisé pour convoyer le lien de canal, les fanions de service, et les informations facultatives de délégation. La somme de contrôle aura un type de 0x8003 (valeur qui est enregistrée dans la spécification du protocole Kerberos) et un champ Valeur d'au moins 24 octets. La longueur du champ Valeur est étendue au delà de 24 octets si et seulement si une facilité facultative de porter un message KRB_CRED, défini par Kerberos pour les besoins de délégation, est supporté par une mise en œuvre et est active sur un contexte. Lorsque la délégation est active, un ticket distributeur de tickets (TGT, *ticket-granting ticket*) avec son fanion TRANSMISSIBLE établi va être transféré au sein du message KRB_CRED.

Le format du champ Valeur de somme de contrôle est le suivant :

Octet	**Nom**	**Description**
0..3	Lgth	Nombre d'octets dans le champ Bnd ; contient actuellement hex 10 00 00 00 (16, représenté en format petit boutien)
4..19	Bnd	Hachage MD5 des liens de canaux, pris sur tous les composants non nuls des liens, dans l'ordre de déclaration. Les champs d'entiers au sein des liens de canaux sont représentés dans l'ordre petit boutien pour les besoins du calcul MD5.
20..23	Flags	Vecteur binaire des fanions d'établissement de contexte, avec des valeurs cohérentes avec la RFC1509, p. 41 : GSS_C_DELEG_FLAG : 1 GSS_C_MUTUAL_FLAG : 2 GSS_C_REPLAY_FLAG : 4 GSS_C_SEQUENCE_FLAG : 8 GSS_C_CONF_FLAG : 16 GSS_C_INTEG_FLAG : 32 Le vecteur binaire résultant est codé dans les octets 20..23 en format petit boutien.
24..25	DlgOpt	Identifiant de l'option Délégation (=1) [facultatif]
26..27	Dlgth	Longueur du champ Délégation. [facultatif]
28..n	Deleg	Message KRB_CRED (n = Dlgth + 29) [facultatif]

Pour le calcul du contenu du champ "Bnd", les points de détail suivants s'appliquent :

(1) Chaque champ d'entier doit être formaté sur quatre octets, en utilisant l'ordre d'octets petit boutien, pour les besoins du calcul du hachage MD5.

(2) Tous les champs de longueur d'entrées au sein des éléments gss_buffer_desc d'une gss_channel_bindings_struct, même ceux qui sont de valeur zéro, devront être inclus dans le calcul du hachage ; les éléments de valeur des éléments gss_buffer_desc devront être déréférencés, et les données résultantes devront être incluses dans le calcul du hachage, seulement pour le cas des éléments de gss_buffer_desc qui ont des spécifiants de longueur non zéro.

(3) Si l'appelant passe la valeur GSS_C_NO_BINDINGS au lieu d'une structure valide de liens de canaux, le champ Bnd devra être établi à 16 octets de valeur zéro.

Dans le jeton initial du mécanisme GSS-API de Kerberos v5 (le jeton KRB_AP_REQ) envoyé de l'initiateur à la cible, les

valeurs GSS_C_DELEG_FLAG, GSS_C_MUTUAL_FLAG, GSS_C_REPLAY_FLAG, et GSS_C_SEQUENCE_FLAG doivent chacune être réglées au ET logique du fanion de demande de l'initiateur correspondant au GSS_Init_sec_context() et un indicateur booléen de si ce service facultatif est disponible pour l'appelant du GSS_Init_sec_context(). GSS_C_CONF_FLAG et GSS_C_INTEG_FLAG, pour lesquels n'existe aucun fanion indicateur de niveau contexte correspondant à GSS_Init_sec_context() doivent chacun être réglés à indiquer si leurs services respectifs de protection par message sont disponibles pour être utilisés sur le contexte en cours d'établissement.

Lorsque des valeurs de lien de canal d'adresse de source d'entrée sont fournies par un appelant (c'est à dire, sauf si l'argument d'entrée est GSS_C_NO_BINDINGS ou si la valeur qui spécifie l'adresse de source dans la structure d'entrée est GSS_C_NULL_ADDRTYPE) et que le jeton correspondant reçu de l'homologue de contexte porte des restrictions d'adresse, il est recommandé qu'une mise en œuvre du mécanisme GSS-API de Kerberos v5 vérifie que l'adresse de source telle que fournie par l'appelant correspond à celle du jeton reçu, et elle devrait retourner la valeur d'état majeur GSS_S_BAD_BINDINGS si une discordance est détectée. Noter que la discussion se poursuit sur la force d'une recommandation dans ce domaine, et sur les circonstances dans lesquelles une recommandation devrait être applicable ; il est donc indiqué aux mises en œuvre que des changements pourraient être inclus dans des versions ultérieures de cette spécification sur ce sujet.

1.1.2 Jetons de réponse

Une séquence d'établissement de contexte fondé sur le mécanisme Kerberos v5 va effectuer une authentification unidirectionnelle (sans confirmation ou aucun jeton de retour de la cible à l'initiateur en réponse au KRB_AP_REQ de l'initiateur) si le bit mutual_req n'est pas établi dans l'appel de l'application à GSS_Init_sec_context(). Les applications qui exigent la confirmation que leur authentification a réussi devraient demander l'authentification mutuelle, résultant en une indication "mutual-required" au sein de l'option KRB_AP_REQ AP et l'établissement du bit mutual_req dans le champ Fanions de la somme de contrôle de l'authentifiant. En réponse à une telle demande, la cible du contexte va répondre à l'initiateur avec un jeton contenant soit une KRB_AP_REP soit une KRB_ERROR, achevant l'échange mutuel d'établissement de contexte.

La syntaxe pertinente pour KRB_AP_REP est la suivante :

```
AP-REP ::= [APPLICATION 15] SEQUENCE {
      pvno [0]          ENTIER,              -- représente Kerberos v5
      msg-type [1]      ENTIER,              -- représente KRB_AP_REP
      enc-part [2]      EncryptedData
}

EncAPRepPart ::= [APPLICATION 27] SEQUENCE {
      ctime [0]         KerberosTime,
      cusec [1]         ENTIER,
      subkey [2]        EncryptionKey FACULTATIF,
      seq-number [3]    ENTIER FACULTATIF
}
```

L'élément facultatif seq-number au sein de la EncAPRepPart de AP-REP devra être inclus.

La syntaxe de KRB_ERROR est la suivante :

```
KRB-ERROR ::=   [APPLICATION 30] SEQUENCE {
      pvno[0]           ENTIER,
      msg-type[1]       ENTIER,
      ctime[2]          KerberosTime FACULTATIF,
      cusec[3]          ENTIER FACULTATIF,
      stime[4]          KerberosTime,
      susec[5]          ENTIER,
      error-code[6]     ENTIER,
      crealm[7]         Realm FACULTATIF,
      cname[8]          PrincipalName FACULTATIF,
      realm[9]          Realm,                    -- Domaine correct
      sname[10]         PrincipalName,            -- Domaine correct
      e-text[11]        GeneralString FACULTATIF,
      e-data[12]        CHAINE D'OCTETS FACULTATIF
}
```

Les valeurs à transférer dans le champ Code d'erreur d'un message KRB-ERROR sont définies dans la [RFC1510], et ne sont pas dans la présente spécification.

1.2 Jetons par message et de suppression de contexte

Trois classes de jetons sont définies dans cette section : les jetons "MIC", émis par des invocations à GSS_GetMIC() (anciennement GSS_Sign()) et consommés par des invocations à GSS_VerifyMIC() (anciennement GSS_Verify()), les jetons "Wrap", émis par des invocations à GSS_Wrap() (anciennement GSS_Seal()) et consommés par des invocations à GSS_Unwrap() (anciennement GSS_Unseal()), et les jetons de suppression de contexte, émis par des invocations à GSS_Delete_sec_context() et consommés par les invocations à GSS_Process_context_token(). Noter que les références aux sous-programmes de GSS-API par message dans le reste de la présente spécification seront fondées sur les nouveaux noms recommandés de ces sous-programmes plutôt que sur leurs noms précédents.

Plusieurs variantes de clés de chiffrement sont utilisées dans la génération et le traitement des jetons par message :
(1) clé de contexte : utilise directement la clé de session Kerberos (ou la sous-clé, si elle est présente dans l'authentifiant émis par l'initiateur du contexte).
(2) clé de confidentialité : forme une variante de la clé de contexte par OU exclusif avec la constante hexadécimale f0f0f0f0f0f0f0f0.
(3) clé de germe MD2.5 : forme une variante de la clé de contexte en inversant les octets de la clé de contexte (c'est-à-dire, si la clé d'origine est la séquence de huit octets {aa, bb, cc, dd, ee, ff, gg, hh} la clé de germe sera {hh, gg, ff, ee, dd, cc, bb, aa}).

1.2.1 Jetons par message - MIC

L'utilisation de l'invocation de GSS_GetMIC() donne un jeton, séparé des données d'usager qu'on protège, qui peut être utilisé pour vérifier l'intégrité de ces données à réception. Le jeton a le format suivant :

N° d'octet	Nom	Description
0..1	TOK_ID	Champ Identification. Les jetons émis par GSS_GetMIC() contiennent la valeur hexadécimale 01 01 dans ce champ.
2..3	SGN_ALG	Indicateur d'algorithme d'intégrité. 00 00 - DES MAC MD5 01 00 - MD2.5 02 00 - DES MAC
4..7	Filler	Contient ff ff ff ff
8..15	SND_SEQ	Champ Numéro de séquence.
16..23	SGN_CKSUM	Somme de contrôle des "données à signer", calculée conformément à l'algorithme spécifié dans le champ SGN_ALG.

Les jetons GSS-API doivent être encapsulés au sein du protocole de niveau supérieur par l'application ; aucun champ de longueur incorporée n'est nécessaire.

1.2.1.1 Somme de contrôle

Procédure de calcul de somme de contrôle (commune à tous les algorithmes) : elles sont calculées sur le champ Données, logiquement précédé par les huit premiers octets de l'en-tête du paquet de texte en clair. La valeur résultante lie les données aux champs Type de paquet et Identifiant d'algorithme de signature.

Algorithme MD5 de MAC DES : la somme de contrôle est formée en calculant un hachage MD5 [RFC1321] sur les données de texte en clair, et ensuite en calculant un MAC DES-CBC sur les 16 octets du résultat MD5. Un MAC standard DES-CBC de 64 bits est calculé selon [FIPS-PUB-113], employant la clé de contexte et un vecteur d'initialisation (IV, *Initialisation Vector*) de zéro. Les 8 octets du résultat sont mémorisés dans le champ SGN_CKSUM.

Algorithme MD2.5 : la somme de contrôle est formée par le premier DES-CBC chiffrant un bloc zéro de 16 octets, en utilisant un IV de zéro et une clé formée en inversant les octets de la clé de contexte (c'est-à-dire, si la clé d'origine est la séquence de 8 octets {aa, bb, cc, dd, ee, ff, gg, hh}, la clé de la somme de contrôle sera {hh, gg, ff, ee, dd, cc, bb, aa}). La valeur résultante de 16 octets est logiquement ajoutée devant les données à signer. Une somme de contrôle MD5 standard est calculée sur les données combinées, et les huit premiers octets du résultat sont mémorisés dans le champ SGN_CKSUM.

Note 1 : on se réfère à cet algorithme de façon informelle comme à "MD2.5" pour noter le fait qu'il utilise la moitié des 128 bits générés par MD5 ; l'utilisation de seulement un sous-ensemble des bits MD5 est destinée à protéger contre

la perspective que les données soient ajoutées à un message existant avec les modifications correspondantes faites à la somme de contrôle.

Note 2 : cet algorithme est très nouveau et a subi une évaluation plus limitée que celle des autres algorithmes d'intégrité. Une évaluation initiale limitée indique qu'il peut être significativement plus faible que le MD5 MAC de DES.

Algorithme DES-MAC : un MAC DES-CBC standard de 64 bits est calculé sur les données de texte en clair selon [FIPS-PUB-113], employant la clé de contexte et un IV de zéro. Les procédures de bourrage pour accommoder les longueurs de données de texte en clair qui ne peuvent pas être des multiples entiers de 8 octets sont définies dans [FIPS-PUB-113]. Le résultat est une valeur de 8 octets, qui est mémorisée dans le champ SGN_CKSUM. La prise en charge de cet algorithme peut n'être pas présente dans toutes les mises en œuvre.

1.2.1.2 Numéro de séquence

Champ Numéro de séquence : le champ de 8 octets du numéro de séquence de texte en clair est formé à partir du numéro de séquence de quatre octets de l'envoyeur de la façon suivante. Si les quatre octets du numéro de séquence de l'envoyeur sont nommés s0, s1, s2 et s3 (de celui de moindre poids à celui de poids fort) le champ Numéro de séquence de texte en clair est la séquence de 8 octets : (s0, s1, s2, s3, di, di, di, di), où 'di' est l'indicateur de direction (Hex 0 – l'envoyeur est l'initiateur du contexte, Hex FF – l'envoyeur est celui qui accepte le contexte). Le champ est alors chiffré en DES-CBC en utilisant la clé de contexte et un IV formé à partir des huit premiers octets du précédent champ SGN_CKSUM calculé. Après l'envoi d'un jeton GSS_GetMIC() ou GSS_Wrap(), le numéro de séquence de l'envoyeur est incrémenté de un.

Le receveur du jeton va d'abord vérifier le champ SGN_CKSUM. Si il est valide, le champ numéro de séquence peut être déchiffré et comparé au numéro de séquence attendu. La répétition de l'indicateur de direction (effectivement 1 bit) au sein du champ numéro de séquence fournit une redondance afin que le receveur puisse vérifier que le déchiffrement a réussi.

Comme le calcul de somme de contrôle est utilisé comme un IV pour le déchiffrement du numéro de séquence, les tentatives de coller une somme de contrôle et un numéro de séquence provenant de différents messages sera détectée. L'indicateur de direction va détecter les paquets qui ont été malicieusement réfléchis.

Le numéro de séquence donne une base pour la détection des jetons répétés. La détection de répétition peut être effectuée en utilisant les informations d'état conservées sur les numéros de séquence reçus, interprétés en conjonction avec le contexte de sécurité sur lequel ils arrivent.

La fourniture des services de détection de la répétition par message et des déclassés est facultative pour les mises en œuvre du mécanisme de GSS-API de Kerberos v5. De plus, il est recommandé que les mises en œuvre du mécanisme de GSS-API de Kerberos v5 qui offrent ces services honorent la demande de l'appelant que les services soient désactivés sur un contexte. Précisément, si le fanion replay_det_req_flag est entré avec FAUX, replay_det_state devrait être retourné FAUX et les états GSS_DUPLICATE_TOKEN et GSS_OLD_TOKEN ne devraient pas être indiqués comme résultat de la détection de dupliqués lorsque les jetons sont traités ; si sequence_req_flag est entré avec FAUX, sequence_state devrait être retourné FAUX et les états GSS_DUPLICATE_TOKEN, GSS_OLD_TOKEN, et GSS_UNSEQ_TOKEN ne devraient pas être indiqués comme résultat de détection de hors séquence lorsque les jetons sont traités.

1.2.2 Jetons par message - Wrap

L'utilisation de l'invocation de GSS_Wrap() donne un jeton qui encapsule les entrées de données d'usager (facultativement chiffrées) avec les quantités associées de vérification d'intégrité. Le jeton émis par GSS_Wrap() consiste en un en-tête d'intégrité dont le format est identique à celui émis par GSS_GetMIC() (sauf que le champ TOK_ID contient la valeur 02 01), suivie par une portion de corps qui contient soit les données de texte en clair (si SEAL_ALG = ff ff) soit les données chiffrées pour toute autre valeur de SEAL_ALG prise en charge. Actuellement, seul SEAL_ALG = 00 00 est pris en charge, et signifie que le chiffrement DES-CBC est utilisé pour protéger les données.

Le format du jeton GSS_Wrap() est le suivant :

N° d'octet	Nom	Description
0..1	TOK_ID	Champ Identification. Les jetons émis par GSS_Wrap() contiennent la valeur hex 02 01 dans ce champ.
2..3	SGN_ALG	Indicateur d'algorithme de somme de contrôle. 00 00 - DES MAC MD5 01 00 - MD2.5 02 00 - DES MAC
4..5	SEAL_ALG	ff ff – aucun ; 00 00 - DES
6..7	Filler	Contient ff ff
8..15	SND_SEQ	Champ Numéro de séquence chiffré.

16..23	SGN_CKSUM	Somme de contrôle des données de texte en clair avec bourrage, calculée selon l'algorithme spécifié dans le champ SGN_ALG.
24..dernier	Data	Données chiffrées ou en texte en clair avec bourrage.

Le jeton GSS-API doit être encapsulé au sein du protocole de niveau supérieur par l'application ; aucun champ de longueur incorporé n'est nécessaire.

1.2.2.1 Somme de contrôle

Procédure de calcul de somme de contrôle (commune à tous les algorithmes) : elles sont calculées sur le champ des données de texte en clair avec bourrage, logiquement précédé par les huit premiers octets de l'en-tête du paquet de texte en clair. La signature résultante lie les données aux champs de type de paquet, version du protocole, et identifiant d'algorithme de signature.

Algorithme DES MAC MD5 : la somme de contrôle est formée en calculant un hachage MD5 sur les données de texte en clair avec bourrage, puis en calculant un MAC DES-CBC sur les 16 octets du résultat MD5. Un MAC standard DES-CBC de 64 bits est calculé selon [FIPS-PUB-113], employant la clé de contexte et un IV de zéro. Le résultat de 8 octets est mémorisé dans le champ SGN_CKSUM.

Algorithme MD2.5 : la somme de contrôle est formée d'abord en chiffrant en DES-CBC un bloc de zéros de 16 octets, en utilisant un IV de zéro et une clé formée en inversant les octets de la clé de contexte (c'est-à-dire, si la clé d'origine est la séquence de huit octets {aa, bb, cc, dd, ee, ff, gg, hh}, la clé de somme de contrôle sera {hh, gg, ff, ee, dd, cc, bb, aa}). La valeur de 16 octets résultante est logiquement précédée des "données à signer". Une somme de contrôle MD5 standard est calculée sur les données combinées, et les huit premiers octets du résultat sont mémorisés dans le champ SGN_CKSUM.

Algorithme DES-MAC : un MAC DES-CBC standard de 64 bits est calculé sur les données de texte en clair avec bourrage selon [FIPS-PUB-113], en employant la clé de contexte et un IV de zéro. Les données de texte en clair avec bourrage sont déjà assurées d'être un multiple entier de 8 octets ; aucun bourrage additionnel n'est requis ou appliqué afin d'accomplir le calcul du MAC. Le résultat est une valeur de 8 octets, qui est mémorisée dans le champ SGN_CKSUM. La prise en charge de cet algorithme peut n'être pas présente dans toutes les mises en œuvre.

1.2.2.2 Numéro de séquence

Champ Numéro de séquence : le champ Numéro de séquence de 8 octets de texte en clair est formé à partir du numéro de séquence de quatre octets de l'envoyeur comme suit. Si les quatre octets du numéro de séquence de l'envoyeur sont nommés s0, s1, s2 et s3 (de celui de moindre poids à celui de poids fort) le champ Numéro de séquence de texte en clair est la séquence de 8 octets (s0, s1, s2, s3, di, di, di, di) où 'di' est l'indicateur de direction (Hex 0 – l'envoyeur est l'initiateur du contexte, Hex FF – l'envoyeur est celui qui accepte le contexte).

Le champ est ensuite chiffré en DES-CBC en utilisant la clé de contexte et un IV formé à partir des huit premiers octets du champ SEAL_CKSUM.

Après l'envoi d'un jeton GSS_GetMIC() ou GSS_Wrap(), le numéro de séquence de l'envoyeur est incrémenté de un.

1.2.2.3 Bourrage

Bourrage des données : avant le chiffrement et/ou le calcul de signature, les données de texte en clair sont bourrées jusqu'au prochain multiple supérieur de 8 octets, en ajoutant entre 1 et 8 octets, la valeur de chacun de ces octets étant le nombre total d'octets de bourrage. Par exemple, soit des données d'une longueur de 20 octets, quatre octets de bourrage seront ajoutés, et chaque octet va contenir la valeur (hex) 04. Un confondeur aléatoire de 8 octets est ajouté devant les données, et les signatures sont calculées sur le texte en clair bourré résultant.

Après le bourrage, les données sont chiffrées selon l'algorithme spécifié dans le champ SEAL_ALG. Pour SEAL_ALG=DES (le seul algorithme non nul actuellement pris en charge, les données sont chiffrées en utilisant DES-CBC, avec un IV de zéro. La clé utilisée est déduite de la clé de contexte établie en appliquant l'opérateur OU-X à la clé de contexte avec la constante hexadécimale f0f0f0f0f0f0f0f0.

1.2.3 Jeton de suppression de contexte

Le jeton émis par GSS_Delete_sec_context() se fonde sur le format de paquet pour les jetons émis par GSS_GetMIC(). Le jeton de suppression de contexte a le format suivant :

N° d'octet	Nom	Description
0..1	TOK_ID	Champ Identification. Les jetons émis par GSS_Delete_sec_context() contiennent la valeur hexadécimale 01 02 dans ce champ.
2..3	SGN_ALG	Indicateur d'algorithme d'intégrité. 00 00 - DES MAC MD5 01 00 - MD2.5 02 00 - DES MAC
4..7	Filler	Contient ff ff ff ff
8..15	SND_SEQ	Champ Numéro de séquence.
16..23	SGN_CKSUM	Somme de contrôle des "données à signer", calculée conformément à l'algorithme spécifié dans le champ SGN_ALG field.

SGN_ALG et SND_SEQ seront calculés comme pour les jetons émis par GSS_GetMIC(). La SGN_CKSUM sera calculée comme pour les jetons émis par GSS_GetMIC(), sauf que le composant données d'usager des "données à signer" sera une chaîne de longueur zéro.

2. Types de nom et identifiants d'objet

Cette section discute des types de noms qui peuvent être passés en entrée de l'invocation de GSS_Import_name() du mécanisme de GSS-API de Kerberos v5, et de leurs valeurs d'identifiant associées. Elle définit les éléments d'interface pour la prise en charge de la portabilité, et suppose l'utilisation des liens du langage C selon la [RFC1509]. En plus de la spécification des valeurs d'OID pour les identifiants de type de nom, des noms symboliques sont inclus et recommandés aux mises en œuvre de GSS-API dans l'intérêt des appelants. On comprend que toutes les mises en œuvre du mécanisme de GSS-API Kerberos v5 n'ont pas besoin de prendre en charge tous les types de nom de cette liste, et que des formes supplémentaires de noms seront vraisemblablement ajoutées à cette liste au fil du temps. De plus, les définitions de certains types de noms, ou de tous, peuvent ultérieurement migrer vers d'autres spécifications, indépendantes du mécanisme. L'occurrence d'un type de nom dans cette spécification est précisément non destinée à suggérer que ce type pourrait n'être pris en charge que par une mise en œuvre du mécanisme Kerberos v5. En particulier, l'occurrence de la chaîne "_KRB5_" dans les chaînes de noms symboliques constitue un moyen d'enregistrer de façon non ambiguë les chaînes de nom, évitant les collisions avec d'autres documents ; elle n'est pas destinée à limiter l'usage ou l'applicabilité des types de noms.

Pour que ce soit clair pour les mises en œuvre de GSS-API, la discussion dans la présente section de certaines formes de noms décrit les moyens par lesquels ces formes peuvent être prises en charge par la technologie Kerberos existante. Ces discussions ne sont pas destinées à empêcher d'autres stratégies de mise en œuvre pour la prise en charge des formes de noms au sein des mécanismes Kerberos ou de mécanismes fondés sur d'autres technologies. Pour améliorer la portabilité des applications, les mises en œuvre des mécanismes sont invitées à prendre en charge les formes de noms telles que définies dans cette section, même si leurs mécanismes sont indépendants de Kerberos v5.

2.1 Formes de nom obligatoires

Ce paragraphe expose les formes de noms qui doivent être prises en charge par toutes les mises en œuvre conformes du mécanisme de GSS-API Kerberos v5.

2.1.1 Forme de nom de principal Kerberos

Cette forme de nom devra être représentée par l'identifiant d'objet {iso(1) member-body(2) United States(840) mit(113554) infosys(1) gssapi(2) krb5(2) krb5_name(1)}. Le nom symbolique recommandé pour ce type est "GSS_KRB5_NT_PRINCIPAL_NAME".

Ce type de nom correspond à la représentation en une seule chaîne d'un nom Kerberos. (Dans la mise en œuvre de Kerberos v5 du MIT, de tels noms sont analysables avec la fonction krb5_parse_name().) Les éléments inclus dans cette représentation de nom sont les suivantes, en commençant par le début de la chaîne :

(1) Un ou plusieurs composants de nom principal ; si plus d'un composant de nom principal est inclus, les composants sont séparés par `/`. Des octets arbitraires peuvent être inclus dans les composants de nom principal, avec les contraintes et considérations particulières suivantes :
   (1a) Toute occurrence des caractères `@` ou `/`dans un composant de nom doit être immédiatement précédée par le caractère de citation `\`, pour empêcher l'interprétation comme un séparateur de composant ou de domaine.
   (1b) Les caractères ASCII de saut à la ligne, tabulation, espace arrière, et nul peuvent intervenir directement dans le composant ou peuvent être représentés, respectivement, par `\n`, `\t`, `\b`, ou `\0`.

(1c) Si le caractère de citation `\` survient en dehors des contextes décrits en (1a) et (1b) ci-dessus, le caractère suivant est interprété littéralement. Comme cas particulier, cela permet à la représentation redoublée `\\` de représenter une seule occurrence du caractère de citation.
(1d) Une occurrence du caractère de citation `\` comme dernier caractère d'un composant est illégale.

(2) Facultativement, un caractère `@`, signifie qu'un nom de domaine suit immédiatement. Si aucun élément de nom de domaine n'est inclus, on suppose le nom de domaine local. Les caractères `/` , `:`, et nul ne peuvent pas apparaître dans un nom de domaine ; les caractères `@`, saut à la ligne, tabulation, et espace arrière peuvent être inclus en utilisant les conventions de citation décrites en (1a), (1b), et (1c) ci-dessus.

2.1.2 Forme de nom de service fondé sur l'hôte

Cette forme de nom a été incorporée au niveau de GSS-API indépendante du mécanisme comme GSS-API, Version 2. Ce paragraphe conserve l'allocation d'identifiant d'objet et de nom symbolique faite précédemment au niveau de mécanisme de GSS-AOI Kerberos v5, et adopte la définition comme promue au niveau indépendant du mécanisme.

Cette forme de nom devra être représentée par l'identifiant d'objet {iso(1) member-body(2) United States(840) mit(113554) infosys(1) gssapi(2) generic(1) service_name(4)}. Le nom symbolique précédemment recommandé pour ce type est "GSS_KRB5_NT_HOSTBASED_SERVICE_NAME". Le nom symbolique actuellement préféré pour ce type est "GSS_C_NT_HOSTBASED_SERVICE".

Ce type de nom est utilisé pour représenter des services associés à des ordinateurs hôtes. Cette forme de nom est construite en utilisant deux éléments, "service" et "hostname", comme suit :

service@hostname

Lorsque une référence à un nom de ce type est résolue, le "hostname" est canonisé en tentant une recherche DNS et en utilisant le nom de domaine pleinement qualifié qui est retourné, ou en utilisant le "hostname" comme il est fourni si la recherche DNS échoue. L'opération de canonisation transpose aussi le nom de l'hôte en caractères minuscules.

L'élément "hostname" peut être omis. Si aucun séparateur "@" n'est inclus, le nom entier est interprété comme le spécificateur du service, avec le "hostname" pris par défaut comme nom canonisé de l'hôte local.

Les valeurs pour l'élément "service" seront enregistrées auprès de l'IANA.

2.1.3 Forme d'objet Nom exporté pour le mécanisme Kerberos v5

La prise en charge de cette forme de nom n'est pas exigée des mises en œuvre de GSS-V1, mais sera exigée pour une utilisation en conjonction avec l'invocation de GSS_Export_name() prévue pour GSS-API version 2. L'utilisation de cette forme de nom sera signifiée par une valeur d'OID de "GSS-API Exported Name Object" qui sera définie au niveau indépendant du mécanisme pour GSS-API version 2.

Ce type de nom représente un objet auto descriptif, dont la structure de tramage sera définie au niveau indépendant du mécanisme pour GSS-API version 2. Lorsque il est généré par le mécanisme Kerberos v5, l'OID de mécanisme au sein du nom exportable sera celui du mécanisme Kerberos v5. Le composant de nom au sein du nom exportable sera une chaîne contiguë avec une structure comme définie pour la forme de nom principal de Kerberos.

Afin de réaliser un codage distinctif pour les besoins de comparaison, les contraintes supplémentaires suivantes sont imposées à l'opération d'exportation :
(1) toutes les occurrences des caractères `@`, `/`, et `\` dans des composants principaux ou noms de domaines devront être citées avec un caractère `\` précédant immédiatement ;
(2) toutes les occurrences des caractères nul, espace arrière, tabulation, ou saut à la ligne dans des composants principaux ou des noms de domaine seront représentées, respectivement, par `\0`, `\b`, `\t`, ou `\n` ;
(3) le caractère de citation `\` ne devra pas être émis dans un nom exporté sauf pour accommoder les cas (1) et (2).

2.2 Formes de nom facultatives

Cette section discute des formes de nom supplémentaires qui peuvent facultativement être prises en charge par les mises en œuvre du mécanisme de GSS-API Kerberos v5. Il est reconnu que certaines des formes de nom citées ici sont dérivées de plates-formes de système d'exploitation UNIX(tm) ; certaines formes citées peuvent n'être pas pertinentes pour des plates-formes non UNIX, et la définition de formes supplémentaires correspondant à de telles plateformes peut aussi être

appropriée. On reconnaît aussi que des fonctions spécifiques du système d'exploitation en dehors de GSS-API existent probablement afin d'effectuer des traductions entre ces formes, et que les mises en œuvre de GSS-API qui prennent en charge ces formes peuvent elles-mêmes être mises en couches par dessus de telles fonctions spécifiques du système d'exploitation. L'inclusion de cette prise en charge au sein des mises en œuvre de GSS-API est un choix de l'application.

2.2.1 Forme de nom d'usager

Cette forme de nom devra être représentée par l'identifiant d'objet {iso(1) member-body(2) United States(840) mit(113554) infosys(1) gssapi(2) generic(1) user_name(1)}. Le nom symbolique recommandé pour ce type est "GSS_KRB5_NT_USER_NAME".

Ce type de nom est utilisé pour indiquer un utilisateur désigné sur un système local.

Son interprétation est spécifique du système d'exploitation. Cette forme de nom est construite comme :

username

En supposant que les noms principaux d'utilisateurs sont les mêmes que les noms de leurs systèmes d'exploitation locaux, une mise en œuvre de GSS_Import_name() fondée sur la technologie Kerberos v5 peut traiter les noms de cette forme en ajoutant un signe "@" et le nom du domaine local.

2.2.2 Forme d'UID de machine

Cette forme de nom devra être représentée par l'identifiant d'objet {iso(1) member-body(2) United States(840) mit(113554) infosys(1) gssapi(2) generic(1) machine_uid_name(2)}. Le nom symbolique recommandé pour ce type est "GSS_KRB5_NT_MACHINE_UID_NAME".

Ce type de nom est utilisé pour indiquer un identifiant d'usager numérique correspondant à un usager sur un système local. Son interprétation est spécifique du système d'exploitation. Le gss_buffer_desc qui représente un nom de ce type devrait contenir un uid_t à signification locale, représenté dans l'ordre des octets de l'hôte. L'opération GSS_Import_name() résout cet uid en un nom d'utilisateur, qui est alors traité comme la forme de nom d'utilisateur.

2.2.3 Forme d'UID de chaîne

Cette forme de nom devra être représentée par l'identifiant d'objet {iso(1) member-body(2) United States(840) mit(113554) infosys(1) gssapi(2) generic(1) string_uid_name(3)}. Le nom symbolique recommandé pour ce type est "GSS_KRB5_NT_STRING_UID_NAME".

Ce type de nom est utilisé pour indiquer une chaîne de chiffres représentant l'identifiant d'usager numérique d'un utilisateur sur un système local. Son interprétation est spécifique du système d'exploitation. Ce type de nom est similaire à la forme d'UID de machine, sauf que la mémoire tampon contient une chaîne représentant le uid_t.

3. Gestion des accréditifs

Le protocole Kerberos v5 utilise des accréditifs différents (au sens de GSS-API) pour initier et accepter les contextes de sécurité. Les clients normaux reçoivent un ticket distributeur de tickets (TGT, *ticket-granting ticket*) et une clé de session associée au moment de "l'amorçage" ; la paire constituée d'un TGT et de sa clé de session correspondante forme un accréditif qui convient pour initier des contextes de sécurité. Un ticket distributeur de tickets, sa clé de session, et toutes les autres paires (ticket, clé) obtenues par l'utilisation de ticket distributeur de tickets, sont normalement mémorisés dans une antémémoire d'accréditifs Kerberos v5, appelée parfois un fichier de tickets.

La clé de chiffrement utilisée par le serveur Kerberos pour sceller les tickets pour un service d'application particulier forme les accréditifs convenables pour accepter les contextes de sécurité. Ces clés de service sont normalement mémorisées dans un tableau de clés Kerberos v5, ou fichier srvtab. En plus de leur utilisation comme accréditifs d'acceptation, ces clés de service peuvent aussi être utilisées pour obtenir des accréditifs d'initiation pour leur principal de service.

Le dispositif d'accréditif du mécanisme Kerberos v5 peut contenir des références à l'un ou l'autre type d'accréditif. C'est une affaire locale de savoir comment la mise en œuvre du mécanisme Kerberos v5 trouve l'antémémoire d'accréditifs ou le tableau de clé approprié de Kerberos v5.

Cependant, lorsque le mécanisme Kerberos v5 tente d'obtenir des accréditifs d'initiation pour un principal de service, qu'ils ne sont pas disponibles dans une antémémoire d'accréditifs, et que la clé pour ce principal de service est disponible dans un tableau de clés Kerberos v5, le mécanisme devrait utiliser la clé de service pour obtenir les accréditifs d'initiation pour ce service. Cela devrait être accompli en demandant un ticket distributeur de tickets au centre de distribution de clés Kerberos, et en déchiffrant la réponse du KDC à l'aide de la clé de service.

4. Définitions des paramètres

Cette section définit les valeurs de paramètres utilisées par le mécanisme de GSS-API de Kerberos v5. Elle définit les éléments d'interface pour la prise en charge de la portabilité, et suppose l'utilisation des liens de langage C selon la [RFC1509].

4.1 Codes d'états mineurs

Ce paragraphe recommande les noms symboliques communs pour les valeurs d'état mineur à retourner par le mécanisme de GSS-API de Kerberos v5. L'utilisation de ces définitions permettra aux mises en œuvre indépendantes d'améliorer la portabilité des applications à travers différentes mises en œuvre du mécanisme défini dans cette spécification. (En tous cas, les mises en œuvre de GSS_Display_status() vont permettre aux appelants de convertir les indicateurs d'état mineur en représentations de texte.) Chaque mise en œuvre devrait rendre disponible, par l'inclusion de fichiers ou d'autres moyens, la facilité de traduire ces noms symboliques en les valeurs concrètes qu'utilise une mise en œuvre particulière de GSS-API pour représenter les valeurs d'état mineur spécifiées dans ce paragraphe.

Cette liste pourra grossir au fil du temps, et il est possible que se fasse sentir le besoin de codes d'états mineurs supplémentaires spécifiques de mises en œuvre particulières. Il est cependant recommandé que les mises en œuvre retournent une valeur d'état mineur comme défini sur la base d'un mécanisme dans ce paragraphe lorsque ce code représente avec précision un état rapportable plutôt que d'utiliser un code distinct, défini par la mise en œuvre.

4.1.1 Codes spécifiques non Kerberos

```
GSS_KRB5_S_G_BAD_SERVICE_NAME   /* "pas de @ dans la chaîne de nom SERVICE-NAME" */
GSS_KRB5_S_G_BAD_STRING_UID     /* "STRING-UID-NAME contient des non chiffres" */
GSS_KRB5_S_G_NOUSER             /* "l'UID ne se résout pas en username" */
GSS_KRB5_S_G_VALIDATE_FAILED    /* "erreur de validation" */
GSS_KRB5_S_G_BUFFER_ALLOC       /* "n'a pas pu allouer les données de gss_buffer_t" */
GSS_KRB5_S_G_BAD_MSG_CTX        /* "contexte de message invalide" */
GSS_KRB5_S_G_WRONG_SIZE         /* "la mémoire tampon n'a pas la bonne taille" */
GSS_KRB5_S_G_BAD_USAGE          /* "le type d'usage de l'accréditif est inconnu" */
GSS_KRB5_S_G_UNKNOWN_QOP        /* "la qualité de protection spécifiée est inconnue" */
```

4.1.2 Codes spécifiques de Kerberos

```
GSS_KRB5_S_KG_CCACHE_NOMATCH    /* "le principal dans l'antémémoire d'accréditif ne correspond pas au
                                   nom désiré" */
GSS_KRB5_S_KG_KEYTAB_NOMATCH    /* "aucun principal du tableau de clé ne correspond au nom désiré" */
GSS_KRB5_S_KG_TGT_MISSING       /* "l'antémémoire d'accréditifs n'a pas de TGT" */
GSS_KRB5_S_KG_NO_SUBKEY         /* "l'authentifiant n'a pas de sous-clé" */
GSS_KRB5_S_KG_CONTEXT_ESTABLISHED /* "le contexte est déjà pleinement établi" */
GSS_KRB5_S_KG_BAD_SIGN_TYPE     /* "le jeton a un type de signature inconnu" */
GSS_KRB5_S_KG_BAD_LENGTH        /* "le jeton a une longueur de champ invalide" */
GSS_KRB5_S_KG_CTX_INCOMPLETE    /* "tentative d'utiliser un contexte de sécurité incomplet" */
```

4.2 Qualité des valeurs de protection

Ce paragraphe définit les valeurs de qualité de protection (QOP) à utiliser avec le mécanisme de GSS-API de Kerberos v5 comme entrées aux sous-programmes GSS_Wrap() et GSS_GetMIC() afin de choisir parmi les différents algorithme de protection de l'intégrité et de confidentialité. Des valeurs supplémentaires de QOP pourront être ajoutées dans de futures versions de cette spécification. Des positions de bit qui ne se chevauchent pas sont et seront employées afin que la qualité de protection de l'intégrité aussi bien que de la confidentialité puisse être choisie avec un seul paramètre, via l'opérateur OU inclusif des valeurs spécifiées d'intégrité et de confidentialité.

4.2.1 Algorithmes d'intégrité

Les valeurs suivantes de qualité de protection sont actuellement définies pour le mécanisme de GSS-API de Kerberos v5, et sont utilisées pour choisir entre les algorithmes de vérification d'intégrité.

GSS_KRB5_INTEG_C_QOP_MD5	(valeur numérique : 1)	/* Intégrité utilisant le MD5 partiel ("MD2.5") de texte en clair */
GSS_KRB5_INTEG_C_QOP_DES_MD5	(valeur numérique : 2)	/* Intégrité utilisant le MAC DES de MD5 de texte en clair */
GSS_KRB5_INTEG_C_QOP_DES_MAC	(valeur numérique : 3)	/* Intégrité utilisant le MAC DES de texte en clair */

4.2.2 Algorithmes de confidentialité

Une seule valeur de qualité de protection de la confidentialité est actuellement définie pour le mécanisme de GSS-API de Kerberos v5 :

GSS_KRB5_CONF_C_QOP_DES	(valeur numérique : 0)	/* Confidentialité avec DES */

Note : la qualité de protection de la confidentialité devrait n'être indiquée que par les appels de GSS-API capables de fournir des services de confidentialité. Si des valeurs de qualité de protection de la confidentialité différentes de zéro sont définies à l'avenir pour représenter différents algorithmes, les positions de bit qui contiennent donc ces valeurs devraient être supprimées avant d'être retournées par les mises en œuvre de GSS_GetMIC() et GSS_VerifyMIC().

4.3 Tailles de mémoire tampon

Toutes les mises en œuvre de la présente spécification devront être capables d'accepter des mémoires tampon d'au moins 16 koctets en entrée à GSS_GetMIC(), GSS_VerifyMIC(), et GSS_Wrap(), et devront être capables d'accepter le jeton de sortie généré par GSS_Wrap() pour une mémoire tampon de 16 koctets comme entrée à GSS_Unwrap(). La prise en charge de plus grandes tailles de mémoire tampon est facultative mais recommandée.

5. Considérations pour la sécurité

Les questions de sécurité sont discutées tout au long du présent mémoire.

6. Références

[RFC1321] R. Rivest, "Algorithme de résumé de message MD5", avril 1992. *(Information)*

[RFC1508] J. Linn, "Interface générique de programme de service de sécurité", septembre 1993. *(P.S.) (remplacée par* RFC2078, *elle-même remplacée par* RFC2743*)*

[RFC1509] J. Wray, "API de service générique de sécurité : liens C", septembre 1993. *(remplacée par* RFC2744*)*

[RFC1510] J. Kohl et C. Neuman, "Service Kerberos d'authentification de réseau (v5)", septembre 1993. *(Obsolète, voir* RFC6649*)*

[FIPS-PUB-113] National Bureau of Standards, Federal Information Processing Standard 113, "Computer Data Authentication", mai 1985.

Adresse de l'auteur

John Linn
OpenVision Technologies
One Main St.
Cambridge, MA 02142
SA
téléphone : +1 617.374.2245
mél : John.Linn@ov.com