Groupe de travail Réseau
Request for Comments : 2412
Catégorie : Information
Traduction Claude Brière de L'Isle

H. Orman
Département d'informatique
Université de l'Arizona
novembre 1998

Protocole OAKLEY de détermination de clés

Statut de ce mémoire
Le présent mémoire fournit des informations pour la communauté de l'Internet. Il ne spécifie aucune sorte de norme de l'Internet. La distribution du présent mémoire n'est soumise à aucune restriction.

Résumé
Le présent document décrit un protocole, appelé OAKLEY, par lequel deux parties authentifiées peuvent se mettre d'accord sur du matériel de clés sûr et secret. Le mécanisme de base est l'algorithme d'échange de clés Diffie-Hellman.

Le protocole OAKLEY prend en charge le secret de transmission parfait (PFS, *Perfect Forward Secrecy*), la compatibilité avec le protocole ISAKMP *(Internet Security Association and Key Management Protocol,* protocole d'associations de sécurité et de gestion de clé Internet) pour la gestion des associations de sécurité, les structures de groupe abstrait définies par l'usager à utiliser avec l'algorithme Diffie-Hellman, les mises à jour de clés, et l'incorporation de clés réparties via des mécanisme hors bande.

Table des matières

1. Introduction

L'établissement des clés est le cœur de la protection des données qui s'appuie sur la cryptographie, et c'est un composant essentiel des mécanismes de protection de paquet décrits dans la [RFC2401], par exemple. Un mécanisme adaptable et sûr de distribution de clés pour l'Internet est une nécessité. Le but de ce protocole est de fournir ce mécanisme, couplé avec une bonne part de force cryptographique.

L'algorithme d'échange de clés Diffie-Hellman fournit un tel mécanisme. Il permet à deux parties de s'accorder sur une valeur partagée sans exiger de chiffrement. La valeur partagée est immédiatement disponible pour être utilisée au chiffrement de la suite de la conversation, par exemple, de la transmission de données et/ou de l'authentification. Le protocole STS [STS] donne la démonstration de la façon d'incorporer l'algorithme dans un protocole sûr, qui assure qu'en plus de partager un secret en toute sécurité, les deux parties peuvent être sûres de l'identité de l'autre, même lorsque il existe un attaquant actif.

Comme OAKLEY est un protocole générique d'échange de clés, comme les clés qu'il génère peuvent être utilisées pour chiffrer des données avec une longue durée de vie de confidentialité, de 20 ans ou plus, il est important que les algorithmes sous-jacents au protocole soient capables d'assurer la sécurité des clés pendant tout ce temps, sur la base des meilleures capacités de prévision disponibles dans l'avenir mathématique. Le protocole a donc deux options pour ajouter aux difficultés rencontrées par un attaquant qui a une grande quantité de trafic d'échange de clés enregistré à sa disposition (un attaquant passif). Ces options sont utiles pour déduire les clés qui seront utilisées pour le chiffrement.

Le protocole OAKLEY se rapporte à STS, partageant les similitudes d'authentification des exponentielles Diffie-Hellman et les utilisant pour déterminer une clé partagée, et aussi de réaliser le secret de transmission parfait pour la clé partagée, mais il diffère du protocole STS de plusieurs façons.

La première est l'ajout d'un mécanisme faible de validation d'adresse ("mouchards", décrits par Phil Karn dans le protocole d'échange de clés Photuris, travail en cours) pour aider à éviter les attaques de déni de service.

La seconde extension est pour permettre aux deux parties de choisir par accord mutuel des algorithmes pris en charge pour le protocole : la méthode de chiffrement, la méthode de déduction de clé, et la méthode d'authentification.

Enfin, l'authentification ne dépend pas du chiffrement utilisant les exponentielles Diffie-Hellman ; l'authentification valide plutôt le lien des exponentielles à l'identité des parties.

Le protocole n'exige pas que les deux parties calculent les exponentielles partagées avant l'authentification.

Ce protocole ajoute de la sécurité supplémentaire à la déduction des clés destinées à être utilisées avec le chiffrement (par opposition à l'authentification) en incluant une dépendance à un algorithme supplémentaire. La déduction des clés pour le chiffrement est faite pour ne pas dépendre que de l'algorithme Diffie-Hellman, mais aussi de la méthode de chiffrement utilisée pour authentifier en toute sécurité les parties communicantes l'une avec l'autre.

Finalement, ce protocole définit explicitement comment les deux parties peuvent choisir les structures mathématiques (représentation et fonctionnement de groupe) pour effectuer l'algorithme Diffie-Hellman ; elles peuvent utiliser les groupes standard ou définir les leurs propres. Les groupes définis par les utilisateurs fournissent un degré supplémentaire de sécurité à long terme.

OAKLEY a plusieurs options de distribution des clés. En plus de l'échange Diffie-Hellman classique, ce protocole peut être utilisé pour déduire une nouvelle clé d'une clé existante et pour distribuer une clé déduite en externe en la chiffrant.

Le protocole permet à deux parties d'utiliser tout ou partie des dispositifs d'anti-obstruction et de secret parfait de transmission. Il permet aussi l'utilisation de l'authentification fondée sur le chiffrement symétrique ou des algorithmes de non chiffrement. Cette souplesse est incluse pour permettre aux parties d'utiliser les caractéristiques qui conviennent le mieux à leurs exigences de sécurité et de performances.

Le présent document emprunte largement dans son esprit et son approche au travail en cours sur Photuris par Karn et Simpson (et de discussions avec les auteurs) et aux particularités du document ISAKMP de Schertler et autres. Le document sur le protocole ISAKMP a été aussi influencé par les articles de Paul van Oorschot et Hugo Krawcyzk.

2. Esquisse du protocole

2.1 Remarques générales

Le protocole OAKLEY est utilisé pour établir une clé partagée avec un identifiant alloué et des identités authentifiées associées pour les deux parties. Le nom de la clé peut être utilisé ensuite pour déduire des associations de sécurité pour les protocoles des RFC2402 et RFC2406 (AH et ESP) ou pour réaliser d'autres objectifs de sécurité du réseau.

Chaque clé est associée à des algorithmes qui sont utilisés pour l'authentification, la confidentialité et des fonctions unilatérales. Ce sont des algorithmes auxiliaires pour OAKLEY ; leur apparition dans les définitions ultérieures d'association de sécurité déduites avec d'autres protocoles n'est ni exigée ni interdite.

La spécification des détails de la façon d'appliquer un algorithme aux données est appelée une transformation. Le présent document ne fournit pas les définitions des transformations ; elles figureront dans des RFC distinctes.

Les jetons anti-obstruction, ou "mouchards", fournissent une forme faible d'identification d'adresse de source pour les deux parties ; l'échange de mouchards peut être achevé avant qu'ils effectuent la partie coûteuse en calcul du protocole (des exponentiations de grands entiers).

Il est important de noter que OAKLEY utilise les mouchards *(cookies)* à deux fins : anti-obstruction et nommage de clé. Les deux parties au protocole contribuent chacune pour un mouchard à l'initiation de l'établissement de clé ; la paire de mouchards devient l'identifiant de clé (KEYID) un nom réutilisable pour le matériel de clés. À cause de ce double rôle, on utilisera la notation pour l'enchaînement des mouchards ("COOKIE-I, COOKIE-R") de façon interchangeable avec le symbole "KEYID".

OAKLEY est conçu pour être un composant compatible du protocole ISAKMP [RFC2408], qui fonctionne sur le protocole UDP en utilisant un accès bien connu (voir la RFC sur l'allocation des numéros d'accès, STD02, RFC1700). La seule exigence technique pour l'environnement du protocole est que la pile de protocoles sous-jacente doit être capable de fournir l'adresse Internet de la partie distante pour chaque message. Donc, OAKLEY pourrait, en théorie, être utilisé directement sur le protocole IP ou sur UDP, si des allocations convenables de numéro de protocole ou d'accès étaient disponibles.

La machine qui fait fonctionner OAKLEY doit fournir un bon générateur de nombres aléatoires, comme décrit dans la [RFC1750], comme source de nombres aléatoires exigée par la présente description du protocole. Toute mention d'un "nom occasionnel" implique que la valeur du nom occasionnel soit générée par un tel outil. Il en est de même pour les valeurs "pseudo aléatoires".

2.2 Notation

Cette section décrit la notation utilisée dans le présent document pour les séquences et contenus de message.

2.2.1 Descriptions de message

Les échanges de protocole ci-dessous sont écrits en notation abrégée qui est destinée à porter les éléments essentiels de l'échange de manière claire. Un bref guide de la notation suit. Les formats détaillés et les valeurs allouées sont données dans les appendices.

Afin de représenter succinctement les échanges de messages, le présent document utilise une notation abrégée qui décrit chaque message en termes de source et destination et des champs pertinents.

Une flèche ("->") indique si le message est envoyé de l'initiateur au répondant, ou vice versa ("<-").

Les champs dans le message sont nommés et séparés par une virgule. Le protocole utilise la convention le premier de plusieurs champs constitue un format d'en-tête fixe pour tous les messages.

Par exemple, considérons un échange de message HYPOTHÉTIQUE qui implique un message de format fixe, les quatre champs fixes étant deux "mouchards", le troisième champ étant un nom de type de message, le quatrième champ étant un entier multi-précision représentant une puissance d'un nombre :

```
    Initiateur                                          Répondant
            ->    Cookie-I, 0, OK_KEYX, g^x                 ->
            <-    Cookie-R, Cookie-I, OK_KEYX, g^y          <-
```

La notation décrit une séquence de deux messages. L'initiateur commence par envoyer un message avec 4 champs au répondant ; le premier champ a la valeur non spécifiée "Cookie-I", le second champ a la valeur numérique 0, le troisième champ indique que le type de message est OK_KEYX, la quatrième valeur est un élément de groupe abstrait g à la puissance x.

La seconde ligne indique que le répondant réplique par la valeur "Cookie-R" dans le premier champ, une copie de la valeur "Cookie-I" dans le second champ, type de message OK_KEYX, et le nombre g élevé à la puissance y.

La valeur OK_KEYX est en majuscules pour indiquer que c'est une constante unique (les constantes sont définies dans les appendices).

Les entiers de précision variable avec une longueur de zéro sont des valeurs nulles pour le protocole.

Parfois, le protocole va indiquer qu'une charge utile entière (usuellement, la charge utile d'échange de clé) a des valeurs nulles. La charge utile est quand même présente dans le message, afin de simplifier l'analyse.

2.2.2 Guide des symboles

Cookie-I et Cookie-R (ou CKY-I et CKY-R) sont des nombres pseudo-aléatoires de 64 bits. Leur méthode de génération doit assurer avec une forte probabilité que les nombres utilisés pour chaque adresse IP distante sont uniques sur une certaine période de temps, comme une heure.

KEYID est l'enchaînement des mouchards de l'initiateur et du répondant et du domaine d'interprétation ; c'est le nom du matériel de clé.

sKEYID est utilisé pour noter le matériel de clé nommé par le KEYID. Il n'est jamais transmis, mais il est utilisé dans divers calculs effectués par les deux parties.

OK_KEYX et OK_NEWGRP sont des types de message distincts.

IDP est un bit qui indique si le matériel après les limites du chiffrement (voir l'appendice B) est ou non chiffré. NIDP signifie non chiffré.

g^x et g^y sont les codages d'éléments de groupe, où g est un élément de groupe spécial indiqué dans la description de groupe (voir l'Appendice A) et g^x indique cet élément élevé à la puissance x. Le type du codage est soit un entier à précision variable, soit une paire de tels entiers, comme indiqué dans l'opération de groupe dans la description de groupe. Noter qu'on écrira g^xy comme raccourci pour g^(xy). Voir à l'Appendice F les références qui décrivent la mise en œuvre de calculs de grands entiers et les relations entre les définitions de groupe diverses et les opérations arithmétiques de base.

EHAO est une liste de choix de chiffrement/hachage/authentification. Chaque élément est une paire de valeurs : un nom de classe et un nom d'algorithme.

EHAS est un ensemble de trois éléments choisis dans la liste EHAO, un de chacune des classes pour le chiffrement, le hachage, l'authentification.

GRP est un nom (valeur de 32 bits) pour le groupe et ses paramètres pertinents : la taille des entiers, l'opération arithmétique, et l'élément générateur. Il y a quelques GRP prédéfinis (pour les groupes d'exponentiation modulaire à 768 bits, modesp à 1024 bits, modexp à 2048 bits, les courbes elliptiques à 155 bits et 210 bits, voir l'Appendice E) mais les participants peuvent partager d'autres descriptions de groupe dans une étape ultérieure du protocole (voir le paragraphe 2.8.1 Définition d'un nouveau groupe). Il est important de distinguer la notion de GRP de celle de descripteur de groupe (Appendice A) ; le premier est un nom pour ce dernier.

Le symbole barre verticale "|" est utilisé pour noter l'enchaînement d'une chaîne binaire. Les champs sont enchaînés en utilisant leur forme codée comme elle apparaît dans leur charge utile.

Ni et Nr sont des noms occasionnels choisis respectivement par l'initiateur et le répondant.

ID(I) et ID(R) sont les identités à utiliser dans l'authentification respectivement de l'initiateur et du répondant.

E{x}Ki indique le chiffrement de x en utilisant la clé publique de l'initiateur. Le chiffrement est fait en utilisant l'algorithme associé à la méthode d'authentification ; normalement, ce sera RSA.

S{x}Ki indique la signature sur x en utilisant la clé privée (clé de signature) de l'initiateur. La signature est faite en utilisant l'algorithme associé à la méthode d'authentification ; ce sera normalement RSA ou DSS.

prf(a, b) note le résultat de l'application d'une fonction pseudo-aléatoire "a" aux données "b". On peut penser "a" comme une clé ou une valeur qui caractérise la fonction prf ; dans le dernier cas, c'est l'indice dans une famille de fonctions. Chaque fonction dans la famille fournit un "hachage" ou mélange unidirectionnel de l'entrée.

prf(0, b) note l'application d'une fonction unidirectionnelle aux données "b".
La ressemblance avec la notation précédente est délibérée et indique qu'un seul algorithme, par exemple, MD5, pourrait être utilisé à ces deux fins. Dans le premier cas, une transformation MD5 "chiffrée" serait utilisée avec la clé "a" ; dans le second cas, la transformation aurait la valeur de clé fixe de zéro, résultant en une fonction unidirectionnelle.

Le terme "transformation" est utilisé pour se référer aux fonctions définies dans des RFC auxiliaires. Les RFC de transformations seront tirées de celles définies pour IPsec AH et ESP (voir al RFC 2401 sur l'architecture globale qui met en application ces protocoles).

2.3 Généralités sur le message d'échange de clés

Le but du traitement de l'échange de clés est l'établissement sûr de l'état d'informations de matériel de clés commun entre les deux parties. Ces informations d'état sont un nom de clé, le matériel de clé secret, l'identification des deux parties, et trois algorithmes à utiliser durant l'authentification : le chiffrement (pour la confidentialité des identités des deux parties) le hachage (une fonction pseudo-aléatoire pour protéger l'intégrité du messages et pour authentifier les champs du message) et l'authentification (l'algorithme sur lequel se fonde l'authentification mutuelle des deux parties). Les codages et significations de ces choix sont présentés à l'Appendice B.

Le mode principal d'échange a cinq caractéristiques facultatives : échange de mouchards sans état, secret parfait vers l'avant pour le matériel de clé, secret des identités, secret parfait vers l'avant pour le secret des identités, utilisation de signatures (pour la non répudiation). Les deux parties peuvent utiliser toute combinaison de ces caractéristiques.

Les lignes générales du traitement sont que l'initiateur de l'échange commence par spécifier autant d'informations qu'il souhaite dans son premier message. Le répondant réplique, fournissant autant d'informations qu'il le souhaite. Les deux côtés échangent des messages, fournissant plus d'informations à chaque fois, jusqn'à ce que leurs exigences soient satisfaites.

Le choix de la quantité d'informations à inclure dans chaque message dépend des options qui sont désirées. Par exemple, si les mouchards sans état ne sont pas une exigence, et si le secret de l'identité et le secret parfait vers l'avant du matériel de clé ne sont pas exigés, et si des signatures non révocables sont acceptables, l'échange peut alors être achevé en trois messages.

Des caractéristiques supplémentaires peuvent augmenter le nombre d'allers-retours nécessaires pour la détermination du matériel de clés.

ISAKMP fournit des champs pour spécifier les paramètres d'association de sécurité à utiliser avec les protocoles AH et ESP. Ces types de charge utile d'association de sécurité sont spécifiés dans le mémorandum sur ISAKMP ; les types de charge utile peuvent être traités avec le matériel et les algorithmes de clé de OAKLEY, mais le présent document ne discute pas de leur utilisation.

2.3.1 Champs essentiels du message d'échange de clés

Il y a 12 champs dans un message d'échange de clé OAKLEY. Tous les champs ne sont pas pertinents dans tous les messages ; si un champ n'est pas pertinent il peut avoir une valeur nulle ou n'être pas présent (pas de charge utile).

CKY-I	mouchard de l'origine.
CKY-R	mouchard du répondant.
MSGTYPE	pour l'échange de clé, sera ISA_KE&AUTH_REQ ou ISA_KE&AUTH_REP ; pour les définitions de nouveaux groupes, sera ISA_NEW_GROUP_REQ ou ISA_NEW_GROUP_REP.
GRP	nom du groupe Diffie-Hellman utilisé pour l'échange.
g^x (ou g^y)	entier de longueur variable représentant une puissance du générateur de groupe.
EHAO ou EHAS	fonctions de chiffrement, hachage, authentification, respectivement offertes et choisies.
IDP	indique si le chiffrement ave g^xy suit ou non (secret parfait vers l'avant pour les identifiants).
ID(I)	identité pour l'initiateur.
ID(R)	identité pour le répondant.
Ni	nom occasionnel fourni par l'initiateur.
Nr	nom occasionnel fournit pas le répondant.

La construction des mouchards dépend de la mise en œuvre. Phil Karn a recommandé d'en faire le résultat d'une fonction unidirectionnelle appliquée à une valeur secrète (changée périodiquement) aux adresses IP locale et distante, et aux accès UDP local et distant. De cette façon, les mouchards restent sans état et arrivent périodiquement à expiration. Noter qu'avec OAKLEY, cela provoquerait aussi l'expiration de l'identifiant de clé KEYID déduit de la valeur secrète, nécessitant le retrait de toute information d'état associée à cette valeur.

Afin de prendre en charge des clés prédistribuées, on recommande que les mises en œuvre réservent une certaine portion de leur espace de mouchards à des clés permanentes. Leur codage ne dépend que de la mise en œuvre locale.

Les fonctions de chiffrement utilisées avec OAKLEY doivent être des transformations cryptographiques qui garantissent la confidentialité et l'intégrité des données du message. On ne peut pas admettre d'utiliser simplement DES en mode CBC. Les transformations OBLIGATOIRES et FACULTATIVES vont inclure toutes celles qui satisfont ces critères et dont l'utilisation est définie avec la RFC2406 (ESP).

Les fonctions unidirectionnelles (de hachage) utilisées avec OAKLEY doivent être des transformations cryptographiques qui peuvent être utilisées soit comme transformations de hachage à clé (pseudo-aléatoire) soit comme transformations non chiffrées. Les transformations OBLIGATOIRES et FACULTATIVES vont inclure toutes celles dont l'utilisation est définie avec la RFC2406 (AH).

Lorsque des noms occasionnels sont indiqués, ils seront des entiers à précision variable avec une valeur d'entropie qui correspond à l'attribut "force" du GRP utilisé dans l'échange. Si aucun GRP n'est indiqué, les noms occasionnels doivent être longs d'au moins 90 bits. Le générateur pseudo-aléatoire pour le matériel de nom occasionnel devrait commencer avec des données initiales qui aient au moins 90 bits d'entropie ; voir la RFC1750.

2.3.1.1 Annonce d'exposant

Idéalement, les exposants vont avoir au moins 180 bits d'entropie pour chaque échange de clé. Cela assure une complète indépendance du matériel de clé entre deux échanges (noter que ceci s'applique si seulement une des parties choisit un exposant aléatoire). En pratique, les mises en œuvre peuvent souhaiter fonder plusieurs échanges de clé sur une seule valeur de base avec 180 bits d'entropie et utiliser des fonctions de hachage unidirectionnelles pour garantir que l'exposition d'une clé ne va pas compromettre les autres. Dans ce cas, une bonne recommandation est de garder les valeurs de base pour les noms occasionnels et les mouchards séparées de la valeur de base pour les exposants, et de remplacer la valeur de base par 180 bits d'entropie complets aussi fréquemment que possible.

Les valeurs 0 et p-1 ne devraient pas être utilisées comme valeurs d'exposant ; les mises en œuvre devraient être sûres de vérifier ces valeurs, et elles devraient aussi refuser d'accepter les valeurs 1 et p-1 des parties distantes (où p est le nombre premier utilisé pour définir un groupe d'exponentiation modulaire).

2.3.2 Transposition en structures de message ISAKMP

Tous les champs de message OAKLEY correspondent aux charges utiles ou aux composants de charge utile de message ISAKMP. Les champs de charge utile pertinents sont la charge utile SA, la charge utile AUTH, la charge utile Certificate, la charge utile Échange de clé. Le cadre du protocole ISAKMP est un travail en cours pour l'instant, et la transposition exacte des champs du message Oakley en charges utiles ISAKMP est aussi en cours (on l'appelle le document Resolution).

Certains des champs d'en-tête et de charge utile ISAKMP vont avoir des valeurs constantes lorsque utilisés avec OAKLEY. Les valeurs exactes à utiliser seront publiées dans un document de domaine d'interprétation qui accompagnera le document Resolution.

Dans ce qui suit, on indique où apparaît chaque champ OAKLEY dans la structure du message ISAKMP. Cela est seulement recommandé ; le document Resolution sera l'autorité finale sur cette transposition.

CKY-I	en-tête ISAKMP
CKY-R	en-tête ISAKMP
MSGTYPE	type de message dans l'en-tête ISAKMP
GRP	charge utile SA, section Proposal
g^x (or g^y)	charge utile d'échange de clé, codé comme entier à précision variable
EHAO et EHAS	charge utile SA, section Proposal
IDP	un bit dans le champ Réservé dans l'en-tête AUTH
ID(I	charge utile AUTH, champ Identity
ID(R)	charge utile AUTH, champ Identity
Ni	charge utile AUTH, champ Nom occasionnel
Nr	charge utile AUTH, champ Nom occasionnel
S{...}Kx	charge utile AUTH, champ Données
prf{K,...}	charge utile AUTH, champ Données

2.4 Protocole d'échange de clés

Le nombre exact et le contenu des messages échangés durant un échange de clé OAKLEY dépend des options que l'initiateur et le répondant veulent utiliser. Un échange de clé peut être achevé avec trois messages ou plus, selon ces options.

Les trois composants du protocole de détermination de clé sont
1. l'échange de mouchards (facultativement sans état)
2. l'échange de demi-clé Diffie-Hellman (facultatif, mais essentiel pour le secret parfait vers l'avant)
3. l'authentification (options : confidentialité des identifiants, confidentialité des identifiants avec PFS, non révocable)

L'initiateur peut fournir aussi peu d'informations qu'une demande d'échange nue, ne portant aucune information supplémentaire. D'un autre côté, l'initiateur peut commencer en fournissant toutes les informations nécessaires pour que le répondant authentifie la demande et achève rapidement la détermination de clé, si le répondant choisit d'accepter cette méthode. Sinon, le répondant peut répondre par une quantité minimale d'informations (au minimum, un mouchard).

La méthode d'authentification peut être avec des signatures numériques, le chiffrement par clé publique, ou une clé symétrique hors bande. Les trois différentes méthodes conduisent à de légères variations dans les messages, et les variantes sont illustrées par les exemples de cette section.

L'initiateur est chargé de retransmettre les messages si le protocole ne se termine pas à temps. Le répondant doit donc éviter d'éliminer les informations des réponses jusqu'à ce qu'il en soit accusé réception par l'initiateur dans le cours de la continuation du protocole.

Le reste de cette section contient des exemples qui montrent comment utiliser les options OAKLEY.

2.4.1 Exemple agressif

L'exemple suivant indique comment deux parties peuvent achever un échange de clé en trois messages. Les identités ne sont pas secrètes, le matériel de clé déduit est protégé par PFS.

En utilisant des signatures numériques, les deux parties auront une preuve de communication qui peut être enregistrée et présentée ultérieurement à un tiers.

Le matériel de clé impliqué par les exponentielles de groupe n'est pas nécessaire pour achever l'échange. Si il est désirable de différer le calcul, la mise en œuvre peut sauvegarder les valeurs "x" et "g^y" et marquer le matériel de clé comme "non calculé". Il peut être calculé ultérieurement à partir de ces informations.

Initiateur **Répondant**

CKY-I, 0, OK_KEYX, GRP, g^x, EHAO, NIDP, ID(I), ID(R), Ni, 0, S{ID(I) | ID(R) | Ni | 0 | GRP | g^x | 0 | EHAO}Ki--->
<----- CKY-R, CKY-I, OK_KEYX, GRP, g^y, EHAS, NIDP, ID(R), ID(I), Nr, Ni, S{ID(R) | ID(I) | Nr | Ni | GRP | g^y | g^x | EHAS}Kr
CKY-I, CKY-R, OK_KEYX, GRP, g^x, EHAS, NIDP, ID(I), ID(R), Ni, Nr, S{ID(I) | ID(R) | Ni | Nr | GRP | g^x | g^y | EHAS}Ki ---------->

"NIDP" signifie que les options PFS pour cacher les identités ne sont pas utilisées. c'est-à-dire, les identités ne sont pas chiffrées en utilisant une clé fondée sur g^xy.

Les champs sont montrés séparés par des virgules dans ce document ; ils sont enchaînés dans les messages de protocole réels en utilisant leur forme codée, comme spécifié dans le document Resolution ISAKMP/Oakley.

Le résultat de cet échange est une clé avec KEYID = CKY-I|CKY-R,
et la valeur sKEYID = prf(Ni | Nr, g^xy | CKY-I | CKY-R).

Les grandes lignes du traitement pour cet échange sont les suivantes :

Initialisation
L'initiateur génère un mouchard unique et l'associe à l'adresse IP attendue du répondant, et les informations d'état qu'il a choisies : GRP (l'identifiant de groupe) un exposant pseudo-aléatoire x qu'il a choisi, g^x, la liste EHAO, le nom occasionnel, les identités. Le premier choix d'authentification dans la liste EHAO est un algorithme qui accepte les signatures numériques, et cela est utilisé pour signer les identifiants, le nom occasionnel et l'identifiant de groupe. L'initiateur note de plus que la clé est dans l'état initial "non authentifié", et établit un temporisateur pour une éventuelle retransmission et/ou terminaison de la demande.

Lorsque le répondant reçoit le message, il peut choisir d'ignorer toutes les informations et les traiter comme simplement une demande de mouchard, ne créant pas d'état. Si CKY-I n'est pas déjà utilisé par l'adresse de source dans l'en-tête IP, le répondant génère un mouchard unique, CKY-R. L'étape suivante dépend des préférences du répondant. La réponse minimale requise est de répondre avec le premier champ de mouchard réglé à zéro et CKY-R dans le second champ. Pour cet exemple, on va supposer que le répondant est plus agressif (pour les autres solutions, voir la section 6) et accepte ce qui suit :

groupe avec GRP identifiant,
premier choix d'authentification (qui doit être la méthode de signature numérique utilisée pour signer le message de l'initiateur),
absence de secret parfait vers l'avant pour protéger les identités,
identité ID(I) et identité ID(R)

Dans cet exemple, le répondant décide d'accepter toutes les informations offertes par l'initiateur. Il valide la signature sur la portion signée du message, et associe la paire (CKY-I, CKY-R) aux informations d'état suivantes :

les adresses réseau de source et destination du message,
l'état de clé "non authentifié",
le premier algorithme de l'offre d'authentification,
le GRP de groupe, une valeur d'exposant "y" dans le GRP de groupe, et g^x provenant du message,
le nom occasionnel Ni et une valeur Nr choisie de façon pseudo-aléatoire,
un temporisateur pour une éventuelle destruction de l'état.

Le répondant calcule g^y, forme le message de réponse, et ensuite signe les informations d'identifiant et de nom occasionnel avec la clé privée de ID(R) et l'envoie à l'initiateur. Dans tous les échanges, chaque partie devrait s'assurer que ni l'un ni l'autre n'offre ni n'accepte 1 ou g^(p-1) comme exponentiel.

Dans cet exemple, pour accélérer le protocole, le répondant accepte implicitement le premier algorithme dans la classe Authentication de la liste EHAO. Cela parce que il ne peut pas valider la signature de l'initiateur sans accepter l'algorithme pour faire la signature. La liste EHAS du répondant va aussi refléter son acceptation.

L'initiateur reçoit le message de réponse et

il valide que CKY-I est une association valide pour l'adresse réseau du message entrant,
il ajoute la valeur CKY-R à l'état pour la paire (CKY-I, adresse réseau) et associe toutes les informations d'état à la paire (CKY-I, CKY-R),

il valide la signature du répondant sur les informations d'état (si la validation devait échouer, le message serait éliminé)
il ajoute g^y à ses informations d'état,
il sauvegarde les choix de EHA dans l'état,
facultativement, il calcule (g^y)^x (= g^xy) (cela peut être différé jusqu'après l'envoi du message de réponse),
il envoie le message de réponse, signé avec la clé publique de ID(I),
il marque le KEYID (CKY-I|CKY-R) comme authentifié, et compose le message de réponse et la signature.

Lorsque le répondant reçoit le message de l'initiateur, et si la signature est valide, il marque la clé comme étant dans l'état authentifié. Il devrait calculer g^xy et l'associer au KEYID.

Noter que bien que PFS ne soit pas utilisé pour la protection de l'identité, PFS pour le matériel de clé déduit est toujours présent parce que les demi clés Diffie-Hellman g^x et g^y sont échangées.

Même si le répondant n'accepte que certaines des informations de l'initiateur, l'initiateur va considérer que le protocole fonctionne. L'initiateur devrait supposer que les champs qui n'ont pas été acceptés par le répondant n'ont pas été enregistrés par le répondant .

Si le répondant n'accepte pas l'échange agressif et choisit un autre algorithme pour la fonction A, alors le protocole ne va pas continuer à utiliser l'algorithme de signature ou la valeur de signature provenant du premier message.

2.4.1.1 Champs non présents

Si le répondant n'accepte pas tous les champs offerts par l'initiateur, il devrait inclure des valeurs nulles pour ces champs dans sa réponse. La Section 6 contient des lignes directrices sur la façon de choisir les champs de "gauche à droite". Si un champ n'est pas accepté, il doit avoir des valeurs nulles ainsi que tous les champs suivants.

Le répondant ne devrait pas enregistrer d'informations qu'il n'accepte pas. Si les identifiants et noms occasionnels ont des valeurs nulles, il n'y aura pas de signature sur ces valeur nulles.

2.4.1.2 Signature via des fonctions pseudo-aléatoires

L'exemple agressif est écrit pour suggérer que la technologie de clé publique est utilisée pour les signatures. Cependant, une fonction pseudo-aléatoire peut être utilisée si les parties s'étaient préalablement mises d'accord sur un tel schéma et ont une clé partagée.

Si la première proposition dans la liste EHAO est une méthode "clé existante", alors le KEYID désigné dans cette proposition fournira le matériel de clé pour la "signature" qui est calculée en utilisant l'algorithme "H" associé au KEYID.

Supposons que la première proposition dans EHAO est EXISTING-KEY, 32 et que l'algorithme "H" pour KEYID 32 est MD5-HMAC, par négociation préalable. Le matériel de clé est une chaîne de bits, qu'on appellera sK32. Alors, dans le premier message de l'échange agressif, où la signature S{ID(I), ID(R), Ni, 0, GRP, g^x, EHAO}Ki est indiquée, le calcul de signature sera effectué par MD5-HMAC_func(KEY=sK32, DATA = ID(I) | ID(R) | Ni | 0 | GRP | g^x | g^y | EHAO) (La définition exacte de l'algorithme correspondant à "MD5-HMAC- func" apparaîtra dans la RFC qui définit cette transformation).

Le résultat de ce calcul apparaît dans la charge utile Authentification.

2.4.2 Exemple agressif avec identités cachées

L'exemple suivant indique comment deux parties peuvent achever un échange de clé sans utiliser de signature numérique. La cryptographie à clé publique cache les identités durant l'authentification. Les exponentielles de groupe sont échangées et authentifiées, mais le matériel de chiffrement impliqué (g^xy) n'est pas nécessaire durant l'échange.

Cet échange a une importante différence avec le précédent schéma de signature --- dans le premier message, une identité pour le répondant est indiquée en clair : ID(R'). Cependant, l'identité cachée avec la cryptographie de clé publique est différente : ID(R). Cela arrive parce que l'initiateur doit d'une certaine façon dire au répondant quelle paire de clé publique/privée utiliser pour le déchiffrement, mais en même temps, l'identité est cachée par chiffrement avec cette clé publique.

L'initiateur peut choisir de renoncer au secret de l'identité du répondant, mais ceci n'est pas souhaitable. À la place, si il y a une identité bien connue pour le nœud répondant, la clé publique pour cette identité peut être utilisée pour chiffrer l'identité réelle du répondant.

Initiateur **Répondant**

----------> CKY-I, 0, OK_KEYX, GRP, g^x, EHAO, NIDP, ID(R'), E{ID(I), ID(R), E{Ni}Kr}Kr' ---------->
<-------- CKY-R, CKY-I, OK_KEYX, GRP, g^y, EHAS, NIDP, E{ID(R), ID(I), Nr}Ki, prf(Kir, ID(R) | ID(I) | GRP | g^y | g^x | EHAS) <-
--------> CKY-I, CKY-R, OK_KEYX, GRP, 0, 0, NIDP, prf(Kir, ID(I) | ID(R) | GRP | g^x | g^y | EHAS) ------->

Kir = prf(0, Ni | Nr)

Note : "NIDP" signifie que l'option PFS pour cacher les identités n'est pas utilisée.
Note : La valeur ID(R') est incluse dans la charge utile Authentification comme décrit à l'Appendice B.

Le résultat de cet échange est une clé avec KEYID = CKY-I|CKY-R et la valeur sKEYID = prf(Ni | Nr, g^xy | CKY-I | CKY-R).

Le traitement pour cet échange est comme suit :

Initiation
L'initiateur génère un mouchard unique et lui associe l'adresse IP escomptée du répondant, et ses informations d'état choisies : GRP, g^x, liste EHAO. Le premier choix d'authentification dans la liste EHAO est un algorithme qui prend en charge le chiffrement par clé publique. L'initiateur désigne aussi deux identités à utiliser pour la connexion et les entre dans l'état. Une identité bien connue pour la machine répondante est aussi choisie, et la clé publique pour cette identité est utilisée pour chiffrer le nom occasionnel Ni et les deux identités de connexion. L'initiateur note de plus que la clé est dans l'état initial de "non authentifié", et lance un temporisateur pour une éventuelle retransmission et/ou terminaison de la demande.

Lorsque le répondant reçoit le message, il peut choisir d'ignorer toutes les informations et le traiter comme une simple demande de mouchard, ne créant pas d'état.

Si CKY-I n'est pas déjà utilisé par l'adresse de source dans l'en-tête IP, le répondant génère un mouchard unique, CKY-R. Comme précédemment, l'étape suivante dépend des préférences du répondant. La réponse minimale requise est un message avec le premier champ de mouchard réglé à zéro et CKY-R dans le second champ. Pour cet exemple, on supposera que le répondant est plus agressif et accepte ce qui suit :

> GRP de groupe, premier choix d'authentification (qui doit être l'algorithme de chiffrement à clé publique utilisé pour chiffrer la charge utile) absence de secret parfait vers l'avant pour protéger les identités, identité ID(I), identité ID(R)

Le répondant doit déchiffrer les informations d'identifiant et de nom occasionnel, en utilisant la clé privée pour le R' ID. Après cela, la clé privée pour le R ID sera utilisée pour déchiffrer le champ Nom occasionnel.

Le répondant associe maintenant la paire (CKY-I, CKY-R) aux informations d'état suivantes :
- les adresses réseau de source et destination du message,
- l'état de clé de "non authentifié",
- le premier algorithme de chaque classe dans la liste EHAO (offres d'algorithmes de chiffrement-hachage-authentification),
- le GRP de groupe et une valeur y et g^y dans le GRP de groupe,
- le nom occasionnel Ni et une valeur Nr choisie de façon pseudo-aléatoire,
- un temporisateur pour une éventuelle destruction de l'état.

Le répondant chiffre alors les informations d'état avec la clé publique de ID(I), forme la valeur prf, et l'envoie à l'initiateur.

L'initiateur reçoit le message de réponse et :
- valide que CKY-I est une association valide pour l'adresse réseau du message entrant,
- ajoute la valeur CKY-R à l'état pour la paire (CKY-I, adresse réseau) et associe toutes les informations d'état à la paire (CKY-I, CKY-R),
- déchiffre les informations d'identifiant et de nom occasionnel,
- vérifie le calcul de prf (si cette vérification échoue, le message est éliminé)
- ajoute g^y à ses informations d'état,
- sauvegarde les choix de EHA dans l'état,
- calcule facultativement (g^x)^y (= g^xy) (cela peut être différé),
- envoie le message de réponse, chiffré avec la clé publique de ID(R),
- et marque le KEYID (CKY-I|CKY-R) comme authentifié.

Lorsque le répondant reçoit ce message, il marque la clé comme étant dans l'état authentifié. Si il ne l'a pas déjà fait, il

devrait calculer g^xy et l'associer au KEYID.

Le matériel de clé secret sKEYID = prf(Ni | Nr, g^xy | CKY-I | CKY-R)

Noter que bien que PFS ne soit pas utilisé pour la protection de l'identité, PFS est quand même présent pour le matériel de clé déduit parce que les demi-clés Diffie-Hellman g^x et g^y sont échangées.

2.4.3 Exemple agressif avec identités privées et sans Diffie-Hellman

Une dépense de calcul considérable peut être évitée si le secret parfait vers l'avant n'est pas une exigence pour la déduction de clé de session. Les deux parties peuvent échanger les noms occasionnels et les parties de clé secrète pour réaliser l'authentification et déduire le matériel de clé. La confidentialité à long terme des données protégées avec du matériel de clé déduit dépend des clés privées de chacune des parties.

Dans cet échange, le GRP a la valeur 0 et le champ pour l'exponentielle de groupe est utilisé pour garder à la place une valeur de nom occasionnel.

Comme au paragraphe précédent, le premier algorithme proposé doit être un système de chiffrement à clé publique ; en répondant par un mouchard et un champ d'exponentielle non zéro, le répondant accepte implicitement la première proposition et l'absence de secret parfait vers l'avant pour les identités et le matériel de clé dérivé.

Initiateur **Répondant**

-------> CKY-I, 0, OK_KEYX, 0, 0, EHAO, NIDP, ID(R'), E{ID(I), ID(R), sKi}Kr', Ni -------->
<-- CKY-R, CKY-I, OK_KEYX, 0, 0, EHAS, NIDP, E{ID(R), ID(I), sKr}Ki, Nr, prf(Kir, ID(R) | ID(I) | Nr | Ni | EHAS) <-----------
------> CKY-I, CKY-R, OK_KEYX, EHAS, NIDP, prf(Kir, ID(I) | ID(R) | Ni | Nr | EHAS) --------->

Kir = prf(0, sKi | sKr)

Note Les valeurs sKi et sKr vont dans les champs Nom occasionnel. Le changement de notation est destiné à souligner que l'entropie est critique pour le réglage du matériel de clé.

Note "NIDP" signifie que l'option PFS pour cacher les identités n'est pas utilisée.

Le résultat de cet échange est une clé avec KEYID = CKY-I|CKY-R et la valeur sKEYID = prf(Kir, CKY-I | CKY-R).

2.4.4 Exemple prudent

Dans cet exemple les deux parties sont minimalement agressives ; elles utilisent l'échange de mouchards pour retarder la création d'état, et elles utilisent le secret parfait vers l'avant pour protéger les identités. Pour cet exemple, elles utilisent le chiffrement par clé publique pour l'authentification ; les signatures numériques ou des clés prépartagées peuvent aussi être utilisées, comme illustré précédemment. Ici, l'exemple prudent ne change pas l'utilisation des nom occasionnels, des prf, ect., mais il change la quantité d'informations transmises dans chaque message.

Le répondant considère la capacité de l'initiateur à répéter CKY-R comme évidence faible que le message provient d'un correspondant "vivant" sur le réseau et le correspondant est associé à l'adresse réseau de l'initiateur. L'initiateur fait des hypothèses similaires lorsque CKY-I est répété chez l'initiateur.

Tous les messages doivent avoir des mouchards valides ou au moins un mouchard à zéro. Si les deux mouchards sont à zéro, cela indique une demande de mouchard ; si seul le mouchard de l'initiateur est zéro, c'est une réponse à une demande de mouchard.

Les informations dans les messages qui violent les règles des mouchards ne peuvent être utilisées pour aucune des opérations OAKLEY.

Noter que l'initiateur et le répondant doivent se mettre d'accord sur un ensemble d'algorithmes EHA ; il n'y a pas un ensemble pour le répondant et un autre pour l'initiateur. L'initiateur doit inclure au moins MD5 et DES dans l'offre initiale.

Les champs non indiqués ont des valeurs nulles.

Initiateur **Répondant**

------> 0, 0, OK_KEYX ------------>
<------ 0, CKY-R, OK_KEYX <------------
------> CKY-I, CKY-R, OK_KEYX, GRP, g^x, EHAO ------------>
<------ CKY-R, CKY-I, OK_KEYX, GRP, g^y, EHAS <------------
------> CKY-I, CKY-R, OK_KEYX, GRP, g^x, IDP*, ID(I), ID(R), E{Ni}Kr, ------------>
<------ CKY-R, CKY-I, OK_KEYX, GRP, 0, 0, IDP, E{Nr, Ni}Ki, ID(R), ID(I), prf(Kir, ID(R) | ID(I) | GRP | g^y | g^x | EHAS) <------------
------> CKY-I, CKY-R, OK_KEYX, GRP, 0 , 0, IDP, prf(Kir, ID(I) | ID(R) | GRP | g^x | g^y | EHAS) ------>

Kir = prf(0, Ni | Nr)

* Lorsque IDP est activé, les charges utiles d'authentification sont chiffrées avec l'algorithme de chiffrement choisi en utilisant le matériel de clé prf(0, g^xy). (La transformation qui définit l'algorithme de chiffrement définira comment choisir les bits de clé à partir du matériel de clé.) Ce chiffrement est en plus et vient après tout chiffrement de clé publique. Voir l'Appendice B.

Noter que dans les premiers messages, plusieurs champs sont omis de la description. Ces champs sont présents comme des valeurs nulles.

Le premier échange permet au répondant d'utiliser des mouchards sans état ; si le répondant génère des mouchards d'une manière qui lui permet de les valider sans les sauvegarder, comme dans Photuris, c'est alors possible. Même si l'initiateur inclut un mouchard dans sa demande initiale, le répondant peut encore utiliser des mouchards sans état en omettant simplement le CKY-I de sa réponse et en refusant d'enregistrer le mouchard de l'initiateur jusqu'à ce qu'il apparaisse dans un message ultérieur.

Après l'achèvement de l'échange, les deux parties calculent le matériel de clé partagé sKEYID comme prf(Ni | Nr, g^xy | CKY-I | CKY-R) où "prf" est la fonction pseudo-aléatoire dans la classe "hachage" choisie dans la liste EHA.

Comme avec les mouchards, chaque partie considère la capacité du côté distant à répéter la valeur Ni ou Nr comme preuve que Ka, la clé publique de la partie a, parle pour la partie distante et établit son identité.

En analysant cet échange, il est important de noter que bien que l'option IDP assure que les identités sont protégées par une clé éphémère g^xy, l'authentification elle-même ne dépend pas de g^xy. Il est essentiel que les étapes d'authentification valident les valeurs g^x et g^y, et il est donc impératif que l'authentification n'implique pas une dépendance circulaire à ces valeurs. Un tiers pourrait intervenir dans un schéma "d'interposition" pour convaincre l'initiateur et le répondant d'utiliser des valeurs g^xy différentes ; bien qu'une telle attaque puisse avoir pour résultat de révéler les identités aux espions, l'authentification va échouer.

2.4.5 Force supplémentaire pour la protection des clés de chiffrement

Les noms occasionnels Ni et Nr sont utilisés pour donner une dimension supplémentaire au secret en déduisant les clés de session. Cela fait dépendre le secret de la clé de deux problèmes différents : le problème du logarithme discret dans le groupe G, et le problème de casser le schéma de chiffrement du nom occasionnel. Si le chiffrement RSA est utilisé, ce second problème est alors en gros équivalent à factoriser les clés publiques RSA de l'initiateur et du répondant.

Pour l'authentification, le type de clé, la méthode de validation, et l'exigence de certification doivent être indiqués.

2.5 Identité et authentification

2.5.1 Identité

Dans les échanges OAKLEY, l'initiateur offre les identifiants d'initiateur et de répondant -- le premier est l'identité revendiquée par l'initiateur, et le second est l'identifiant demandé pour le répondant .

Si aucun des deux identifiants n'est spécifié, les ID sont tirés des adresses de source et destination de l'en-tête IP.

Si l'initiateur ne fournit pas un identifiant de répondant, le répondant peut répondre en désignant toute identité que permet la politique locale. L'initiateur peut refuser de l'accepter en terminant l'échange.

Le répondant peut aussi répondre avec un identifiant différent de celui qu'a suggéré l'initiateur ; l'initiateur peut accepter implicitement cela en continuant l'échange ou le refuser en terminant (en ne répondant pas).

2.5.2 Authentification

L'authentification mutuelle des principaux est au cœur de tout schéma d'échange de clés. La communauté de l'Internet doit décider d'un standard adaptable pour résoudre ce problème, et OAKLEY doit utiliser ce standard. Au moment de la rédaction de ce texte, un tel standard n'existe pas, bien que plusieurs soient émergeants. Le présent document tente de décrire comment une poignée de normes pourrait être incorporée dans OAKLEY, sans tenter de faire des choix entre elles.

Les méthodes suivantes peuvent apparaître dans l'offre d'OAKLEY :

a. Clés pré-partagées
   Lorsque deux parties se sont mises d'accord sur une méthode de confiance pour la distribution des clés secrètes pour leur authentification mutuelle, elles peuvent être utilisées pour l'authentification. Cela pose des problèmes d'adaptation évidents pour les grands systèmes, mais c'est une solution intérimaire acceptable dans certaines situations. La prise en charge des clé pré-partagées est EXIGÉE.

   L'algorithme de chiffrement, hachage, et authentification à utiliser avec une clé pré-partagée doit faire partie des informations d'état distribuées avec la clé elle-même.

   Les clés pré-partagées ont un KEYID et le matériel de clé sKEYID ; le KEYID est utilisé dans une offre d'option d'authentification de clé pré-partagée. Il peut y avoir plus d'une offre de clé pré-partagée dans une liste.

   Comme le KEYID persiste sur différentes invocations de OAKLEY (après un crash, etc.) il doit occuper une partie réservée de l'espace de KEYID pour les deux parties. Quelques bits peuvent être mis de côté dans "l'espace de mouchards" de chaque partie pour s'accommoder de cela.

   Il n'y a pas d'autorité de certification pour les clés pré-spartagées. Lorsque une clé pré-partagée est utilisée pour générer une charge utile d'authentification, l'autorité de certification est "Aucune", le type d'authentification est "Prépartagé", et la charge utile contient le KEYID, codé par deux quantités de 64 bits, et le résultat de l'application de la fonction de hachage pseudo-aléatoire au corps de message avec le sKEYID formant la clé pour la fonction.

b. Clé publiques DNS
   Les extensions de sécurité au protocole DNS [RFC2065] fournissent un moyen commode pour accéder aux informations de clé publique, en particulier pour les clés publiques associées aux hôtes. Les clés RSA sont une exigence pour les mises en œuvre sûres du DNS ; des extensions pour permettre des clés DSS facultatives sont une possibilité à court terme.

   Les enregistrements DNS KEY ont des enregistrements SIG associés qui sont signés par une autorité de zone, et une hiérarchie de signatures jusqu'au serveur racine établissent un fondement pour la confiance. Les enregistrements SIG indiquent l'algorithme utilisé pour former la signature.

   Les mises en œuvre OAKLEY doivent prendre en charge l'utilisation des enregistrements DNS KEY et SIG pour l'authentification par rapport aux adresses IPv4 et IPv6 et les noms de domaine pleinement qualifiés. Cependant, il n'est pas exigé des mises en œuvre qu'elles prennent en charge un algorithme particulier (RSA, DSS, etc.).

c. Les clés publiques RSA w/o signature d'autorité de certification
   PGP [PGP] utilise des clés publiques avec une méthode informelle pour établir la confiance. Le format des clés publiques et des méthodes de dénomination de PGP sera décrit dans une RFC séparée. L'algorithme RSA peut être utilisé avec les clés PGP soit pour signer soit pour chiffrer ; l'option d'authentification devrait indiquer respectivement soit RSA-SIG soit RSA-ENC. La prise en charge de ceci est FACULTATIVE.

d.1 Clé publiques RSA w/ certificats
   Il y a divers formats et conventions de dénomination pour les clés publiques qui sont signées par une ou plusieurs autorités de certification. Le protocole d'échange de clé publique (PKI, *Public Key Interchange*) expose les codages et la validation X.509. La prise en charge de ceci est FACULTATIVE.

d.2 Clés DSS w/ certificats
   Le codage pour la norme de signature numérique avec X.509 est décrite dans le draft-ietf-ipsec-dss-cert-00.txt. La prise en charge de ceci est FACULTATIVE ; un type d'authentification ISAKMP sera alloué.

2.5.3 Validation des clés d'authentification

La combinaison de l'algorithme d'authentification, de l'autorité d'authentification, du type d'authentification, et d'une clé (usuellement publique) définit comment valider les messages par rapport à l'identité revendiquée. Les informations de clé seront disponibles soit à partir d'une clé pré-partagée, soit à partir de quelque sorte d'autorité de certification.

Généralement, l'autorité de certification produit un certificat qui lie le nom de l'entité à une clé publique. Les mises en œuvre OAKLEY doivent être prêtes à aller chercher et valider les certificats avant d'utiliser la clé publique pour les besoins de l'authentification OAKLEY.

La charge utile Authentification ISAKMP définit le champ Autorité d'authentification pour spécifier l'autorité qui doit apparaître dans la hiérarchie de confiance pour l'authentification.

Une fois qu'un certificat approprié est obtenu (voir au paragraphe 2.4.3) la méthode de validation va dépendre du type d'authentification ; si c'est PGP, alors, les sous-programmes de validation de signature PGP peuvent être invoqués pour satisfaire aux prédicats de la toile de confiance locale ; si c'est RSA avec des certificats X.509, le certificat doit être examiné pour voir si la signature de l'autorité de certification peut être validée, et si la hiérarchie est reconnue par la politique locale.

2.5.4 Collecte des objets d'identité

En plus de l'interprétation de certificat ou d'autre structure de données qui contiennent une identité, les usagers de OAKLEY doivent faire face à la tâche de restitution des certificats qui lient une clé publique à un identifiant et aussi de restituer les certificats auxiliaires pour les autorités certificatrices ou cosignataires (comme dans la toile de confiance PGP).

La charge utile ISAKMP Accréditifs peut être utilisée pour rattacher d'utiles certificats aux messages OAKLEY. La charge utile Accréditifs est définie à l'Appendice B.

La prise en charge de l'accès et de la révocation des certificats de clé publique via le protocole DNS sécurisé de la [RFC2065] est OBLIGATOIRE pour les mises en œuvre OAKLEY. D'autres méthodes de restitution peuvent être utilisées lorsque la classe AUTH indique une préférence.

Le protocole d'échange de clé publique PKI expose un protocole complet qui pourrait être utilisé avec les certificats codés selon X.509.

2.6 Interface des transformations cryptographiques

Le matériel de clé calculé par l'échange de clés devrait avoir au moins 90 bits d'entropie, ce qui signifie qu'il doit avoir au moins 90 bits de long. Ce peut être plus ou moins que ce qui est requis pour le chiffrement et/ou les transformations de fonction pseudo-aléatoire.

Les transformations utilisées avec OAKLEY devraient avoir des algorithmes auxiliaires qui tirent parti de l'entier à précision variable et le transforment en matériel de clé de la longueur appropriée. Par exemple, un algorithme DES pourrait prendre les 56 bits de moindre poids, un algorithme triple DES pourrait utiliser ce qui suit :

K1 = 56 bits de moindre poids de md5(0|sKEYID)
K2 = 56 bits de moindre poids de md5(1|sKEYID)
K3 = 56 bits de moindre poids de md5(2|sKEYID)

Les transformations seront invoquées avec le matériel de clé codé comme entier à précision variable, la longueur des données et le bloc de mémoire avec les données. La conversion du matériel de clé en une clé de transformation est de la responsabilité de la transformation.

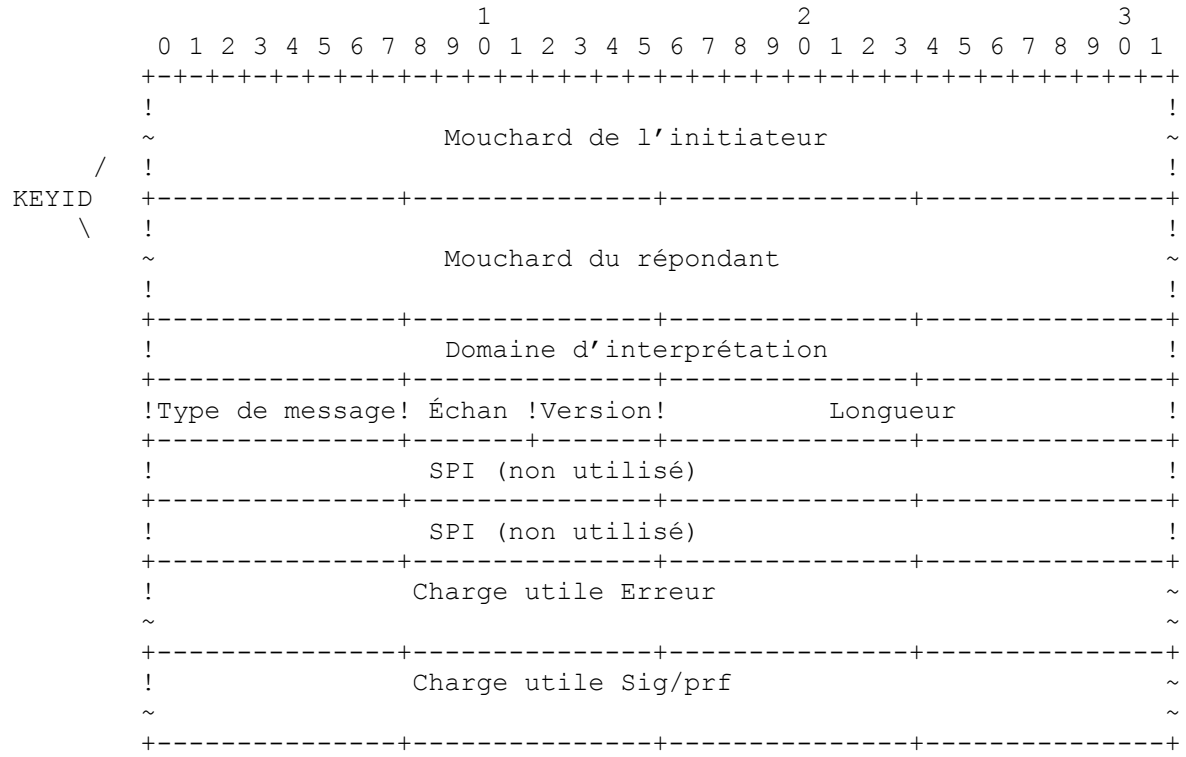
2.7 Retransmission, fins de temporisation, et messages d'erreur

Si une réponse du répondant n'est pas obtenue dans un délai approprié, le message devrait être retransmis par l'initiateur. Ces retransmissions doivent être traitées en douceur par les deux parties ; le répondant doit conserver les informations pour retransmission jusqu'à ce que l'initiateur passe au prochain message dans le protocole ou termine l'échange.

Les messages avec des erreurs d'informations posent un problème parce que ils ne peuvent pas être authentifiés en utilisant seulement les informations présentes dans un échange incomplet ; pour cette raison, les parties peuvent souhaiter établir une clé par défaut pour les messages d'erreur OAKLEY. Une méthode possible pour établir une telle clé est décrite dans l'Appendice B, sous l'utilisation des types de message ISA_INIT.

Dans ce qui suit, le type de message est Erreur OAKLEY, le KEYID fournit l'algorithme H et la clé pour authentifier le contenu du message ; cette valeur est portée dans la charge utile Sig/Prf.

La charge utile Erreur contient le code d'erreur et le contenu du message rejeté.



Le message d'erreur va contenir les mouchards tels que présentés dans le message erroné, le type de message OAKLEY_ERROR, et la cause de l'erreur, suivie par le message rejeté.

Les messages d'erreur ne sont que pour information, et la correction du protocole ne dépend pas d'eux.

Causes d'erreur :

TIMEOUT	l'échange a pris trop longtemps, l'état est détruit
AEH_ERROR	un algorithme inconnu apparaît dans une offre
GROUP_NOT_SUPPORTED	le GRP désigné n'est pas pris en charge
EXPONENTIAL_UNACCEPTABLE	l'exponentielle est trop grande/petite ou est +-1
SELECTION_NOT_OFFERED	le choix n'apparaît pas dans l'offre
NO_ACCEPTABLE_OFFERS	aucune offre ne répond aux exigences de l'hôte
AUTHENTICATION_FAILURE	échec de signature ou de fonction de hachage
RESOURCE_EXCEEDED	trop d'échanges ou trop d'informations d'état
NO_EXCHANGE_IN_PROGRESS	une réponse est reçue alors qu'aucune demande n'est en cours

2.8 Sécurité supplémentaire pour les clés de confidentialité : groupes privés

Si les deux parties ont besoin d'utiliser un schéma de détermination de clé Diffie-Hellman qui ne dépend pas de la définition de groupe standard, elles ont l'option d'établir un groupe privé. L'authentification n'a pas besoin d'être répétée, parce que cette étape du protocole sera protégée par une clé d'authentification préexistante. Comme mesure de sécurité supplémentaire, les deux parties vont établir un nom privé pour le matériel de clé partagé, de sorte que même si ils utilisent exactement le même groupe pour communiquer avec d'autres parties, la réutilisation ne sera pas apparente à un attaquant passif.

Les groupe privés présente l'avantage de rendre une attaque passive largement répandue plus difficile en augmentant le nombre de groupes qui devraient être analysés de façon exhaustive afin de récupérer un grand nombre de clés de session. Cela se distingue du cas où seulement un ou deux groupes sont utilisés ; dans ce cas, on s'attendrait à ce que des années et des années de clés de session soient compromises.

Il y a deux défis techniques à relever : comment un utilisateur particulier crée un groupe unique et approprié, et comment une seconde partie s'assure que le groupe proposé est raisonnablement sûr ?

La sécurité d'un groupe d'exponentiation modulaire dépend du plus grand facteur premier de la taille du groupe. Afin de maximiser cette taille, on peut choisir des nombres premiers "forts" ou Sophie Germaine, P = 2Q + 1, où P et Q sont premiers. Cependant, si P = kQ + 1, où k est petit, alors la force du groupe est encore considérable. Ces groupes sont connus sous le nom de sous groupes de Schnorr, et ils peuvent être trouvés avec beaucoup moins d'efforts de calcul que les premiers Sophie-Germaine.

Les sous-groupes de Schnorr peuvent aussi être validés efficacement en utilisant des essais de premiers probables.

Il est aussi assez facile de trouver P, k, et Q tels que le plus grand facteur premier puisse être facilement prouvé comme étant Q.

On estime qu'il prendrait environ 10 minutes pour trouver un nouveau groupe d'environ 2^1024 éléments, et cela pourrait être fait une fois par jour par un processus programmé ; valider un groupe proposé par une partie distante prendrait peut-être une minute sur une machine RISC à 25 MHz ou une machine CISC à 66 MHz.

On note que la validation est seulement faite entre des parties mutuellement authentifiées préalablement, et qu'une nouvelle définition de groupe suit toujours et est protégée par une clé établie en utilisant un groupe bien connu. Il y a cinq points à garder en mémoire :

a. La description et l"identifiant public pour le nouveau groupe sont protégés par le groupe bien connu.

b. Le répondant peut rejeter la tentative d'établissement du nouveau groupe, soit parce qu'il est trop occupé, soit parce qu'il ne peut pas valider le plus grand facteur premier comme étant suffisamment grand.

c. Les nouveaux module et générateur peuvent être mis en antémémoire pour de longues périodes ; ils ne sont pas critiques pour la sécurité et n'ont pas besoin d'être associés à l'activité en cours.

d. Générer périodiquement une nouvelle valeur g^x va être plus coûteux si il y a de nombreux groupes en antémémoire ; cependant, l'importance de générer fréquemment de nouvelles valeurs de g^x est réduite, de sorte que la période peut être allongée en conséquence.

e. Tous les groupes d'exponentiation modulaires ont des sous-groupes qui sont plus faibles que le groupe principal. Pour les nombres premiers Sophie-Germaine, si le générateur est un carré, il y a alors seulement deux éléments dans le sous-groupe : 1 et g^(-1) (le même que g^(p-1)) que nous avons déjà recommandé d'éviter. Pour les sous-groupes de Schnorr avec k non égal à 2, le sous-groupe peut être évité en vérifiant que l'exponentielle n'est pas une racine $k^{ème}$ de 1 (e^k != 1 mod p).

2.8.1 Définition d'un nouveau groupe

Ce paragraphe décrit comment définir un nouveau groupe. La description du groupe est cachée aux espions, et l'identifiant alloué au groupe est univoque pour les deux parties. L'utilisation du nouveau groupe pour des échanges de clé Diffie-Hellman est décrite au paragraphe suivant.

Le secret de la description et l'identifiant augmentent la difficulté d'une attaque passive, parce que si le descripteur de groupe n'est pas connu de l'attaquant, il n'y a pas de moyen direct et efficace pour obtenir des informations sur les clés calculées en utilisant le groupe.

Seule la description du nouveau groupe a besoin d'être chiffrée dans cet échange. L'algorithme de hachage est impliqué par la session OAKLEY désignée par le groupe. Le chiffrement est la fonction de chiffrement de la session OAKLEY.

Le descripteur du nouveau groupe est codé dans la charge utile du nouveau groupe. Les noms occasionnels sont codés dans la charge utile Authentification.

Les données au delà de la limite du chiffrement sont chiffrées en utilisant la transformation désignée par le KEYID.

Les messages suivants utilisent l'identifiant d'échange de clé ISAKMP Nouveau groupe OAKLEY.

Pour définir un nouveau groupe d'exponentiation modulaire :

Initiateur **Répondant**

-------> KEYID, INEWGRP, Desc(Nouveau groupe), Na prf(sKEYID, Desc(New Group) | Na) ---------->
<------- KEYID, INEWGRPRS, Na, Nb prf(sKEYID, Na | Nb | Desc(New Group)) <----------
--------> KEYID, INEWGRPACK, prf(sKEYID, Nb | Na | Desc(New Group)) ---------->

Ces messages sont chiffrés à la limite de chiffrement en utilisant la clé indiquée. La valeur de hachage est placée dans le champ "signature numérique" (voir l'Appendice B).

Identifiant de nouveau GRP = trunc16(Na) | trunc16(Nb)

(trunc16 indique la troncature à 16 bits ; l'initiateur et le répondant doivent utiliser des noms occasionnels qui ont des bits de poids fort distincts de tous ceux utilisés pour les GRPID en cours.)

Desc(G) est le codage du descripteur pour le descripteur de groupe (voir à l'Appendice A le format d'un descripteur de groupe).

Les deux parties doivent mémoriser la transposition entre l'identifiant du nouveau groupe GRP et le descripteur de groupe Desc(Nouveau groupe). Ils doivent aussi noter les identités utilisées pour le KEYID et les copier dans l'état pour le nouveau groupe.

Noter qu'on pourrait avoir le même descripteur de groupe associé à plusieurs KEYID. Le pré-calcul des valeurs de g^x peut être fait sur la seule base du descripteur de groupe, et non sur le nom du groupe privé.

2.8.2 Déduction d'une clé en utilisant un groupe privé

Une fois qu'un groupe privé a été établi, son identifiant de groupe peut être utilisé dans les messages d'échange de clé dans la position du GRP. Aucun changement du protocole n'est requis.

2.9 Mode rapide : nouvelles clés à partir des vieilles

Lorsque un KEYID authentifié et le matériel de clé associé sKEYID existe déjà, il est facile de déduire des KEYID supplémentaires et des clés partageant des attributs similaires (GRP, EHA, etc.) en utilisant seulement des fonctions de hachage. Le KEYID pourrait être un de ceux qui ont été déduits en mode principal, par exemple.

D'un autre côté, la clé authentifiée peut être une clé distribuée manuellement, qui est partagée par l'initiateur et le répondant via des moyens extérieurs à OAKLEY. Si la méthode de distribution a formé le KEYID en utilisant des valeurs uniques appropriées pour les deux moitiés (CKY-I et CKY-R) alors, cette méthode est applicable.

Dans ce qui suit, l'identifiant d'échange de clé est OAKLEY Quick Mode. Les noms occasionnels sont portés dans la charge utile Authentification, et la valeur de la prf est portée dans la charge utile Authentification ; l'autorité d'authentification est "Aucune" et le type est "Pré-partagé".

Le protocole est :

Initiateur **Répondant**

----------> KEYID, INEWKRQ, Ni, prf(sKEYID, Ni) ---------->
<---------- KEYID, INEWKRS, Nr, prf(sKEYID, 1 | Nr | Ni) <----------
----------> KEYID, INEWKRP, 0, prf(sKEYID, 0 | Ni | Nr) ---------->

Le nouveau KEYID, NKEYID, est Ni | Nr

sNKEYID = prf(sKEYID, Ni | Nr)

Les identités et valeurs de EHA associées au NKEYID sont les mêmes que celles associées au KEYID.

Chaque partie doit valider les valeurs de hachage avant d'utiliser la nouvelle clé pour quelque fin que ce soit.

2.10 Définition et utilisation de clés prédistribuées

Si une clé et un identifiant de clé associé ainsi que les informations d'état ont été distribuées manuellement, alors la clé peut être utilisée pour tout objet OAKLEY. La clé doit être associée aux informations d'état usuelles : identifiants et algorithmes EHA.

La politique locale dicte quand une clé manuelle peut être incluse dans la base de données OAKLEY. Par exemple, seul les usagers privilégiés auraient la permission d'introduire des clés associées à des identifiants privilégiés, un usager non privilégié pourrait seulement introduire des clés associées à son propre identifiant.

2.11 Distribution d'une clé externe

Une fois que sont établis une clé de session OAKLEY et les algorithmes auxiliaires, le matériel de clé et l'algorithme "H" peuvent être utilisés pour distribuer une clé générée en externe et pour lui allouer un KEYID.

Dans ce qui suit, KEYID représente une clé de session OAKLEY authentifiée existante, et sNEWKEYID représente le matériel de clé généré en externe.

Dans ce qui suit, l'identifiant d'échange de clé est OAKLEY Mode externe. La charge utile Échange de clé contient la nouvelle clé, qui est protégée.

Initiateur **Répondant**

---------> KEYID, IEXTKEY, Ni, prf(sKEYID, Ni) ---------->
<--------- KEYID, IEXTKEY, Nr, prf(sKEYID, 1 | Nr | Ni) <----------
----------> KEYID, IEXTKEY, Kir xor sNEWKEYID*, prf(Kir, sNEWKEYID | Ni | Nr) ---------->

Kir = prf(sKEYID, Ni | Nr)

* ce champ est porté dans la charge utile Échange de clé.

Chaque partie doit valider les valeurs de hachage en utilisant la fonction "H" dans l'état KEYID avant de changer aucune information d'état de clé.

La nouvelle clé est récupérée par le répondant en calculant le OUX du champ dans la charge utile Authentification avec la valeur du Kir.

Le nouvel identifiant de clé, qui désigne le matériel de clé sNEWKEYID, est prf(sKEYID, 1 | Ni | Nr).

Noter que cet échange n'exige pas de chiffrement. Hugo Krawcyzk a suggéré la méthode et notés ses avantages.

2.11.1 Considérations de force cryptographique

La force de la clé utilisée pour distribuer la clé externe doit être au moins égale à la force de la clé externe. Généralement, cela signifie que la longueur du matériel sKEYID doit être supérieure ou égale à la longueur du matériel sNEWKEYID.

La déduction de la clé externe, sa force ou l'usage auquel elle est destinée ne sont pas traités dans le présent protocole ; les parties qui utilisent la clé doivent avoir une autre méthode pour déterminer ces propriétés.

Au début 1996, il apparaît que pour 90 bits de force cryptographique, on devrait utiliser un groupe d'exponentiation modulaire modulo 2000 bits. Pour 128 bits de force, un modulo de 3000 bits est requis.

3. Spécification et déduction des associations de sécurité

Lorsque une association de sécurité est définie, seul le KEYID a besoin d'être donné. Le répondant devrait être capable de rechercher l'état associé avec les valeurs de KEYID et trouver le matériel de clé approprié, sKEYID.

Déduire les clés à utiliser avec les protocoles IPsec tels que ESP ou AH est un sujet couvert dans le document Resolution ISAKMP/Oakley. Ce document décrit aussi comment négocier les ensembles de paramètres acceptables et les identifiants pour ESP et AH, et comment calculer exactement le matériel de clés pour chaque instance des protocoles. Comme le matériel de clé de base défini ici (g^xy) peut être utilisé pour déduire les clés pour plusieurs instances de ESP et AH, les mécanismes exacts d'utilisation de fonctions unidirectionnelles pour transformer g^xy en plusieurs clés univoques sont essentiels à une utilisation correcte.

4. Compatibilité ISAKMP

OAKLEY utilise les formats d'en-tête et de charge utile ISAKMP, comme décrits dans ce texte et à l'Appendice B. Il y a des extensions particulièrement remarquables au delà du projet de version 4.

4.1 Authentification avec les clés existantes

Dans le cas où deux parties n'ont pas en place de mécanismes convenables de clé publique pour s'authentifier l'un l'autre, elles peuvent utiliser des clés qui ont été distribuées manuellement. Après l'établissement de ces clés et de leur état associé dans OAKLEY, elles peuvent être utilisées pour des modes d'authentification qui dépendent des signatures, par exemple le mode agressif.

Lorsque une clé existante va apparaître dans une liste d'offres, elle devrait être indiquée avec un algorithme d'authentification de ISAKMP_EXISTING. Cette valeur sera allouée dans la RFC ISAKMP.

Lorsque la méthode d'authentification est ISAKMP_EXISTING, l'autorité d'authentification aura la valeur ISAKMP_AUTH_EXISTING ; la valeur pour ce champ ne doit pas entrer en conflit avec une autre autorité d'authentification enregistrée auprès de l'IANA et est définie dans la RFC ISAKMP.

La charge utile d'authentification aura deux parties :
le KEYID pour la clé préexistante
l'identifiant pour la partie à authentifier par la clé préexistante.

La fonction pseudo-aléatoire "H" dans les informations d'état pour ce KEYID sera l'algorithme de signature, et elle utilisera le matériel de clé pour cette clé (sKEYID) lors de la génération ou la vérification de la validité des données du message.

Par exemple, si la clé existante a un KEYID noté par KID et 128 bits de matériel de clé noté sKID et l'algorithme"H" une transformation dénommée HMAC, alors pour générer une "signature" pour un bloc de données, le résultat de HMAC(sKID, données) sera la charge utile Signature correspondante.

L'état KEYID aura les identités de la partie locale et de la partie distante pour lesquelles le KEYID a été alloué ; il appartient à la mise en œuvre de politique locale de décider quand il est approprié d'utiliser une telle clé pour authentifier l'autre partie. Par exemple, une clé distribuée pour être utilisée entre deux hôtes Internet A et B peut convenir pour authentifier toutes les identités de la forme "alice@A" et "bob@B".

4.2 Authentification par un tiers

Une politique de sécurité locale pourrait restreindre la négociation de clé aux parties de confiance. Par exemple, deux démons OAKLEY fonctionnant avec des étiquettes de sensibilité égales sur deux machines pourraient souhaiter être les seuls arbitres des échanges de clés entre les utilisateurs avec cette même étiquette sensible. Dans ce cas, un moyen d'authentifier la provenance des demandes d'échange de clés est nécessaire. C'est-à-dire que les identités des deux démons devraient être liées à une clé, et cette clé sera utilisée pour former une "signature" pour les messages d'échange de clé.

La charge utile Signature, dans l'Appendice B, sert à cela. Cette charge utile désigne un KEYID qui est en existence avant le début de l'échange en cours. La transformation "H" pour le KEYID est utilisée pour calculer une valeur d'intégrité/authentification pour toutes les charges utiles qui précèdent la signature.

La politique locale peut imposer quels KEYID sont appropriés pour signer les échanges ultérieurs.

4.3 Mode de nouveau groupe

OAKLEY utilise un nouveau KEI pour l'échange qui définit un nouveau groupe.

5. Notes pour la mise en œuvre de la sécurité

Des attaques par analyse des temps qui sont capable de récupérer la valeur de l'exposant utilisé dans les calculs Diffie-Hellman ont été décrites par Paul Kocher [Kocher]. Afin de réduire cette attaque à néant, les mises en œuvre doivent prendre la peine de brouiller la séquence d'opérations impliquées dans le portage des exponentiations modulaires.

Un "facteur d'aveuglement" peut accomplir cet objectif. Un élément de groupe, r, est choisi au hasard. Lorsque un exposant x est choisi, la valeur r^(-x) est aussi calculée. Puis, lors du calcul de (g^y)^x, la mise en œuvre va calculer cette séquence :

A = (rg^y)

B = A^x = (rg^y)^x = (r^x)(g^(xy))
C = B*r^(-x) = (r^x)(r^-(x))(g^(xy)) = g^(xy)

Le facteur d'aveuglement n'est nécessaire que si l'exposant x est utilisé plus de 100 fois (estimé par Richard Schroeppel).

6. Analyse OAKLEY et automate à états

Il y a de nombreux chemins à travers OAKLEY, mais ils suivent un schéma d'analyse de gauche à droite des champs du message.

L'initiateur décide d'un message initial dans l'ordre suivant :

1. Offre d'un mouchard. Ceci n'est pas nécessaire mais aide pour les échanges agressifs.

2. Prendre un groupe. Les choix sont ceux des groupes bien connus ou de tous groupes privés qui peuvent avoir été négociés. Le premier échange entre deux démons Oakley sans état commun doit impliquer un groupe bien connu (0, signifiant pas de groupe, est un groupe bien connu). Noter que l'identifiant de groupe, et non le descripteur de groupe, est utilisé dans le message.
   Si un groupe non nul est utilisé, il doit être inclus avec le premier message spécifiant l'EHAO. Il n'est pas nécessaire qu'il soit spécifié jusque là.

3. Si PFS est utilisé, prendre un exposant x et présenter g^x.

4. Listes des offres de chiffrement, hachage, et authentification.

5. Utiliser PFS pour cacher les identités
   Si la dissimulation des identités n'est pas utilisée, l'initiateur a alors cette option:

6. Nommer les identités et inclure les informations d'authentification

Les informations de la section authentification dépendent de la première offre d'authentification. Dans cet échange agressif, l'initiateur espère que le répondant va accepter toutes les informations offertes et la première méthode d'authentification. La méthode d'authentification détermine la charge utile d'authentification comme suit :

1. Méthode de signature. La signature sera appliquée à toutes les informations offertes.

2. Une méthode de chiffrement de clé publique. L'algorithme sera utilisé pour chiffrer un nom occasionnel dans la clé publique de l'identité demandée du répondant. Il y a deux cas possibles, selon que la dissimulation d'identité est utilisée ou non.
   a. Pas de dissimulation d'identité. Les identités vont apparaître en clair.
   b. Dissimulation d'identité. Une identité bien connue, appelons la R', va apparaître en clair dans la charge utile d'authentification. Elle sera suivie pas deux identités et un nom occasionnel ; ceux-ci seront chiffrés avec la clé publique pour R'.

3. Une méthode de clé préexistante. La clé préexistante sera utilisée pour chiffrer un nom occasionnel. Si la dissimulation d'identité est utilisée, l'identité sera chiffrée en place dans la charge utile, en utilisant l'algorithme "E" associé à la clé préexistante.

Le répondant peut accepter tout ou partie du message initial ou rien du tout.

Le répondant accepte autant de champs qu'il le souhaite, en utilisant le même ordre de décision que l'initiateur. Il peut arrêter à toute étape, rejetant implicitement les autres champs (qui auront des valeurs nulles dans son message de réponse). La réponse minimum est un mouchard et le GRP.

1. Accepte le mouchard. Le répondant peut choisir de n'enregistrer aucune information d'état jusqu'à ce que l'initiateur réussisse à répondre par un mouchard choisi par le répondant. S'il en est ainsi, le répondant réplique par un mouchard, le GRP, et aucune autre information.
2. Accepte le GRP. Si le groupe n'est pas acceptable, le répondant ne va pas répondre. Le répondant peut envoyer un message d'erreur indiquant que le groupe n'est pas acceptable (modulo trop petit, identifiant inconnu, etc.) Noter que "pas de groupe" a deux significations dans le protocole : cela peut vouloir dire que le groupe n'est pas encore spécifié, ou cela peut signifier qu'aucun groupe ne sera utilisé (et donc que PFS n'est pas possible).

3. Accepte la valeur g^x. Le répondant indique son acceptation de la valeur g^x en incluant sa propre valeur g^y dans sa réponse. Il peut différer cela en ignorant g^x et en mettant une valeur de g^y de longueur zéro dans sa réponse. Il peut aussi rejeter la valeur de g^x avec un message d'erreur.

4. Accepter un élément de chacune des listes EHA. L'acceptation est indiquée par une proposition non zéro.

5. Si PFS pour la dissimulation d'identité est demandé, aucune autre donnée ne suivra.
6. Si la charge utile d'authentification est présente, et si le premier élément dans la classe d'authentification offerte est acceptable, alors le répondant doit valider/déchiffrer les informations de la charge utile d'authentification et de la charge utile de signature, si elle est présente. Le répondant devrait choisir un nom occasionnel et répondre en utilisant le même algorithme d'authentification/hachage que l'initiateur a utilisé.

L'initiateur note les informations que le répondant a acceptées, valide/déchiffre tous les champs signés, hachés, ou chiffrés, et si les données sont acceptables, répond conformément à la méthode EHA choisie par le répondant. Les réponses de l'initiateur se distinguent de son message initial par la présence de la valeur non zéro pour le mouchard du répondant.

Le résultat de la signature ou de la fonction prf va être codé comme un entier à précision variable comme décrit à l'Appendice C. Le KEYID va indiquer le KEYID qui désigne le matériel de clé et le hachage ou la fonction de signature.

7. Charge utile d'accréditifs

Des certificats utiles avec des informations de clé publique peuvent être attachés aux messages OAKLEY en utilisant les charges utiles Credential, comme défini dans le document ISAKMP. On notera que l'option de protection de l'identité s'applique aux accréditifs aussi bien qu'aux identités.

Considérations pour la sécurité

L'objet de ce document est la sécurité ; donc, les considérations de sécurité imprègnent ce mémoire.

Adresse de l'auteur

Hilarie K. Orman
Department of Computer ScienceUniversity of Arizona
mél : ho@darpa.mil

Appendice A Descripteurs de groupe

Trois représentations de groupe distinctes peuvent être utilisées avec OAKLEY. Chaque groupe est défini par son opération de groupe et la sorte de champ sous-jacent utilisé pour représenter les éléments du groupe. Les trois types sont des groupes d'exponentiation modulaires (appelés ici MODP), des groupes de courbe elliptique sur le champ GF[2^N] (appelés ici EC2N), et des groupes de courbe elliptique sur GF[P] (appelés ici ECP). Pour chaque représentation, de nombreuses réalisations distinctes sont possibles, selon le choix des paramètres.

À quelques exceptions près, tous les paramètres sont transmis comme si ils étaient des entiers multi-précision non négatifs, en utilisant le format défini dans le présent appendice (noter que ceci est distinct du codage de l'Appendice C). Chaque entier multi-précision a un champ de longueur préfixé, même lorsque ces informations sont redondantes.

Pour le type de groupe EC2N, les paramètres sont plus précisément vus comme de très long champs binaires, mais ils sont représentés comme des entiers multi-précision (avec des champs de longueur, et justifiés à droite). C'est le codage naturel.

MODP signifie le groupe d'exponentiation modulaire classique, où l'opération est de calculer G^X (mod P). Le groupe est défini par les paramètres numériques P et G. P doit être premier. G est souvent 2, mais peut être un nombre plus grand : $2 \leq G \leq P-2$.

ECP est un groupe de courbe elliptique, modulo un nombre premier P. L'équation qui définit cette sorte de groupe est Y^2 = X^3 + AX + B. L'opération du groupe prend un multiple d'un point d'une courbe elliptique. Le groupe est défini par cinq paramètres numériques : le nombre premier P, deux paramètres de courbe A et B, et un générateur (X,Y). A, B, X, Y

sont tous interprétés mod P, et doivent être des entiers (non négatifs) inférieurs à P. Ils doivent satisfaire à l'équation de définition, modulo P.

EC2N est un groupe de courbe elliptique, sur le champ fini F[2^N]. L'équation de définition pour cette sorte de groupe est Y^2 + XY = X^3 + AX^2 + B. (Cette équation diffère légèrement du cas mod P : elle a un terme XY, et un terme AX^2 au lieu d'un terme AX.)

On doit spécifier la représentation du champ, et ensuite la courbe elliptique. Le champ est spécifié en donnant un polynôme irréductible (mod 2) de degré N. Ce polynôme est représenté par un entier d'une taille entre 2^N et 2^(N+1) comme si le polynôme de définition était évalué à la valeur U=2.

Par exemple, le champ défini par le polynôme U^155 + U^62 + 1 est représenté par l'entier 2^155 + 2^62 + 1. Le groupe est défini par 4 autres paramètres, A, B, X, Y. Ces paramètres sont des éléments du champ GF[2^N], et peuvent être vus comme un polynôme de degré < N, avec des coefficients (mod 2). Ils tiennent dans des champs de N bits, et sont représentés par des entiers < 2^N, comme si le polynôme était évalué à U=2. Par exemple, l'élément de champ U^2 + 1 serait représenté par l'entier 2^2+1, qui est 5. Les deux paramètres A et B définissent la courbe. A est fréquemment 0. B ne doit pas être 0. Les paramètres X et Y sélectionnent un point sur la courbe. Les paramètres A, B, X, Y doivent satisfaire l'équation de définition, modulo le polynôme de définition, et modulo 2.

Formats de descripteur de groupe

Type de groupe : Un champ de deux octets, les valeurs allouées pour les types "MODP", "ECP", "EC2N" seront définies (voir ISAKMP-04).
Taille d'un élément champ, en bits. C'est soit Plafond(log2 P) soit le degré du polynôme irréductible : un entier de 32 bits.
Le nombre premier P ou le polynôme champ irréductible : un entier multi précision.
Le générateur : 1 ou 2 valeurs, entiers multi précision.
EC seulement : Les paramètres de la courbe : 2 valeurs, entiers multi précision.

Les paramètres suivants sont facultatifs (chacun d'eux peut apparaître indépendamment) :
une valeur de 0 peut être utilisée comme bouche-trou pour représenter un paramètre non spécifié ; tout nombre des paramètres peut être envoyé, de 0 à 3.

Le plus grand facteur premier (LPF, *Largest Prime Factor*) : la valeur codée qui est le LPF de la taille du groupe, un entier multi précision.

EC seulement : L'ordre du groupe : entier multi précision. (La taille du groupe pour MODP est toujours P-1.)

Force du groupe : entier de 32 bits.
La force du groupe est approximativement le nombre de bits clés protégés.
t est déterminé par le log2 de l'effort pour attaquer le groupe. Il peut changer lorsque on en saura plus sur la cryptographie.

C'est un exemple générique pour un groupe "classique" d'exponentiation modulaire :
Type de groupe : "MODP"
Taille d'un élément champ en bits : Log2 (P) arrondi *à l'entier supérieur*. Entier de 32 bits.
Premier P de définition : un entier multi précision.
Générateur G : un entier multi précision. $2 \leq G \leq P-2$.

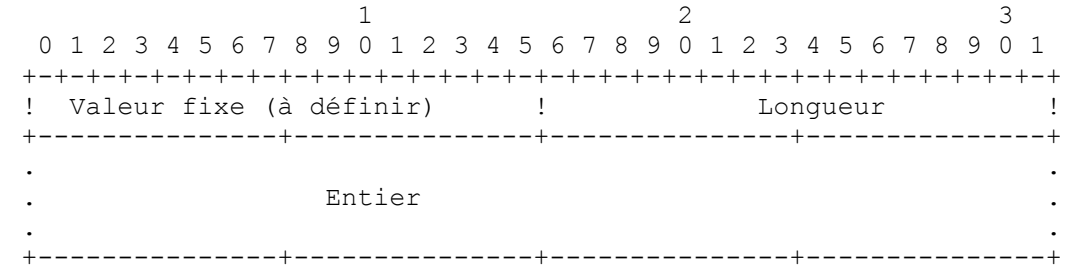
<facultatif>
Plus grand facteur premier de P-1 : l'entier multi précision Q
Force du groupe : un entier de 32 bits. On spécifiera une formule pour calculer ce nombre (à définir).

Ceci est un exemple générique pour un groupe à courbe elliptique, mod P:
Type de groupe : "ECP"
Taille d'un élément champ en bits : Log2 (P) arrondi *à l'entier supérieur*, un entier de 32 bits.
Premier P de définition : un entier multi précision.
Générateur (X,Y) : 2 entiers multi précision, chacun < P.
Paramètres A, B de la courbe : 2 entiers multi précision, chacun < P.
<facultatif>
Plus grand facteur premier de l'ordre du groupe : un entier multi précision.
Ordre du groupe : un entier multi précision.
Force du groupe : un entier de 32 bits. Formule à définir.

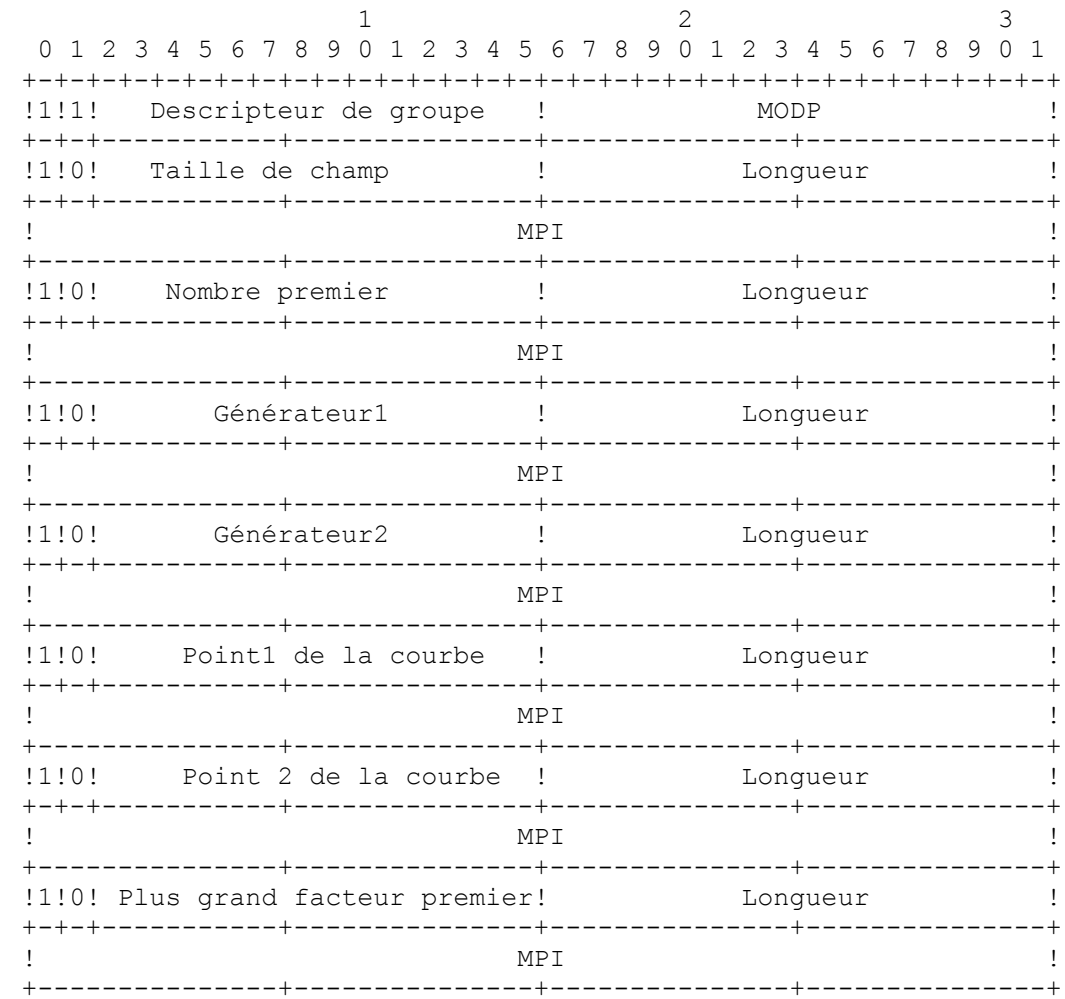
Voici un exemple spécifique pour un groupe à courbe elliptique :

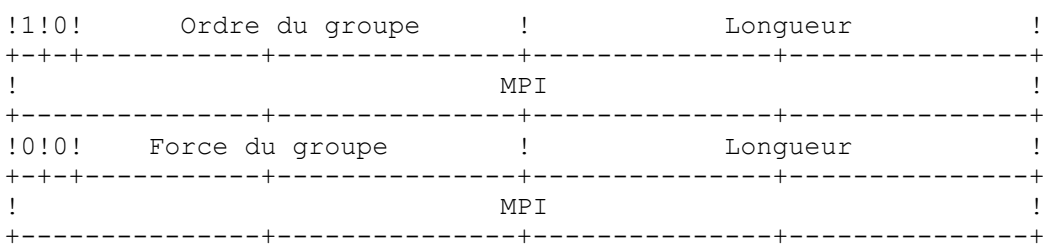
Type de groupe : "EC2N"
Degré du polynôme irréductible : 155
Polynôme irréductible : U^155 + U^62 + 1, représenté comme un entier multi précision 2^155 + 2^62 + 1.
Générateur (X,Y) : représenté comme 2 entiers multi précision, chacun < 2^155.
Pour notre présente courbe, il y a (en décimal) 123 et 456. Chacun est représenté comme entier multi précision.
Paramètres de la courbe A, B : représentés comme 2 entiers multi précision, chacun < 2^155.
Pour notre présente courbe, ces sont 0 et (en décimal) 471 951, représenté comme deux entiers multi précision.
<facultatif>
Plus grand facteur premier de l'ordre du groupe : 3805993847215893016155463826195386266397436443,
représenté comme entier multi précision.
L'ordre du groupe : 4567192616659071619386556591434463519676923731 6
représenté comme entier multi précision.
Force du groupe : 76, représenté comme un entier de 32 bits.

Le codage d'entier à précision variable pour les champs de descripteur de groupe est le suivant. C'est une légère variation par rapport au format défini dans l'Appendice C en ce qu'une valeur fixe de 16 bits est utilisée d'abord, et que la longueur est limitée à 16 bits. Cependant, l'interprétation est par ailleurs identique.



Le format d'un descripteur de groupe est :



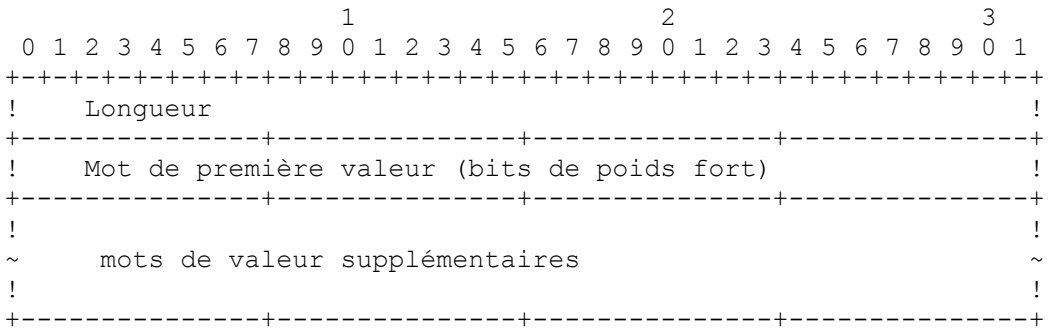


Appendice B Formats de message

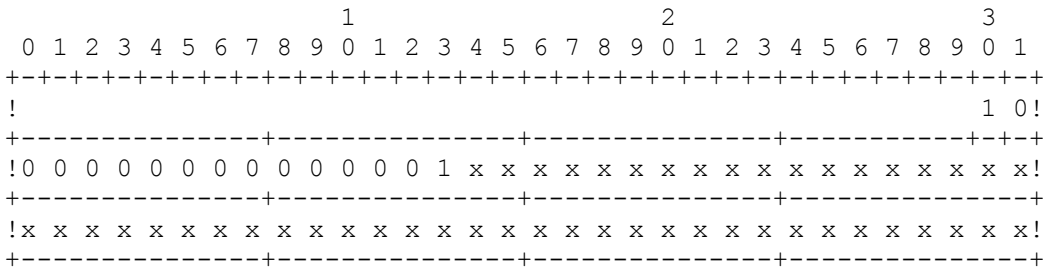
Le codage des messages Oakley dans les charges utiles ISAKMP est reporté au document Resolution ISAKMP/Oakley.

Appendice C Codage d'un entier à précision variable

Les entiers à précision variable seront codés comme un champ d'une longueur de 32 bits suivi par une ou plusieurs quantités de 32 bits contenant la représentation de l'entier, alignée sur le bit de plus fort poids dans le premier élément de 32 bits.



Un exemple d'un tel codage est donné ci-dessous, pour un nombre avec 51 bits significatifs. Le champ Longueur indique que 2 quantités de 32 bits suivent. Le bit de poids fort non zéro du nombre est dans le bit 13 de la première quantité de 32 bits, les bits de moindre poids sont dans la seconde quantité de 32 bits.



Appendice D Forces cryptographiques

L'algorithme Diffie-Hellman est utilisé pour calculer les clés qui seront utilisées avec des algorithmes symétriques. Il ne devrait pas être plus facile de casser le calcul Diffie-Hellman que de faire une recherche exhaustive sur l'espace de clés symétrique. Une recommandation récente par un groupe de cryptographes [Blaze] a recommandé une taille de clé symétrique de 75 bits pour un niveau pratique de sécurité. Pour 20 années de sécurité, ils recommandent 90 bits.

Sur la base de ce rapport, une stratégie prudente pour les utilisateurs OAKLEY serait de s'assurer que leurs calculs Diffie-Hellman sont au moins aussi sûrs qu'un espace de clé de 90 bits. Afin de réaliser cela pour les groupes d'exponentiation modulaires, la taille du plus grand facteur premier du module devrait être d'au moins 180 bits, et la taille du module devrait être d'au moins 1400 bits. Pour les groupes à courbe elliptique, le LPF devrait être d'au moins 180 bits.

Si le secret à long terme de la clé de chiffrement n'est pas un problème, alors, les paramètres suivants peuvent être utilisés pour le groupe d'exponentiation modulaire : 150 bits pour le LPF, 980 bits pour la taille du module.

La taille du module seul ne détermine pas la force du calcul Diffie-Hellman ; la taille de l'exposant utilisé dans les puissances de calcul au sein du groupe est aussi importante. La taille de l'exposant en bits devrait être au moins deux fois la taille de toute clé symétrique qui en sera déduite. On recommande que les mises en œuvre de ISAKMP utilisent au moins 180 bits d'exposant (deux fois la taille d'une clé symétrique de 20 ans).

La justification mathématique de ces estimations peut se trouver dans les textes qui estiment les efforts pour résoudre le problème du logarithme discret, une tâche qui est en forte relation avec l'efficacité de l'utilisation du champ numérique Sieve pour factoriser les grands entiers. Le lecteur se reportera à [Stinson] et [Schneier].

Appendice E Groupes bien connus

Identifiants de groupe :
0 Pas de groupe (utilisé comme bouche trou et pour les échanges non DH)
1 Un groupe d'exponentiation modulaire avec un module de 768 bits
2 Un groupe d'exponentiation modulaire avec un module de 1024 bits
3 Un groupe d'exponentiation modulaire avec un module de 1536 bits (à définir)
4 Un groupe à courbe elliptique sur GF[2^155]
5 Un groupe à courbe elliptique sur GF[2^185]

Les valeurs 2^31 et supérieures sont utilisées pour les identifiants de groupes privés

Richard Schroeppel a effectué tout le travail mathématique et de calcul pour cet appendice.

Groupes d'exponentiation modulaires Diffie-Hellman classiques
Les nombres premiers pour les groupes 1 et 2 ont été choisis comme ayant certaines propriétés. Les 64 bits de poids fort sont forcés à 1. Cela aide l'algorithme classique du reste, parce que le chiffre quotient d'essai peut toujours être pris comme mot de poids fort du dividende, éventuellement +1. Les 64 bits de moindre poids sont forcés à 1. Cela aide les algorithmes de reste de style Montgomery, parce que le chiffre multiplicateur peut toujours être pris comme étant le mot de moindre poids du dividende. Les bits du milieu sont tirés de l'expansion binaire de pi. Cela garantit qu'ils sont effectivement aléatoires, tout en évitant tout soupçon que les nombres premiers auraient été secrètement choisis comme étant faibles.

Parce que les deux nombres premiers sont fondés sur pi, il y a une grande section de chevauchement dans les représentations hexadécimales des deux nombres premiers. Ceux-ci sont choisis pour être des nombres premiers Sophie Germain (c'est-à-dire, (P-1)/2 est aussi premier) pour avoir le maximum de force contre l'attaque de la racine carrée sur le problème du logarithme discret.

Les nombres d'essai du début ont été incrémentés de façon répétée de 2^64 jusqu'à ce que des nombres premiers convenables soient localisés.

Parce que ces deux nombres premiers sont congruents à 7 (mod 8), 2 est un résidu quadratique de chaque premier. Toutes les puissances de 2 seront aussi des résidus quadratiques. Cela empêche un opposant d'apprendre le bit de moindre poids de l'exposant Diffie-Hellman (autrement dit, les problèmes du confinement du sous-groupe). Utiliser 2 comme générateur est efficace pour certains algorithmes d'exponentiation modulaire. [Noter que 2 n'est pas techniquement un générateur au sens de la théorie des nombres, parce qu'il omet la moitié des résidus possibles mod P. D'un point de vue cryptographique, c'est une vertu.]

E.1 Groupe bien connu 1 : un nombre premier de 768 bits

Le nombre premier est 2^768 - 2^704 - 1 + 2^64 * { [2^638 pi] + 149686 }. Sa valeur décimale est
155251809230070893513091813125848175563133404943451431320235119490296623994910210725866945387659164
244291000768028886422915080371891804634263272761303128298374438082089019628850917069131659317536746
955176311984337163722100721057791 9

Cela a été vérifié rigoureusement comme premier.

La représentation du groupe en OAKLEY est
Type de groupe : "MODP"
Taille de l'élément champ (bits) : 768
Module premier : 21 (en décimal)

Longueur (mots de 32 bits) : 24
Données (en hexadécimal) : FFFFFFFF FFFFFFFF C90FDAA2 2168C234 C4C6628B 80DC1CD1 29024E08 8A67CC74 020BBEA6 3B139B22 514A0879 8E3404DD EF9519B3 CD3A431B 302B0A6D F25F1437 4FE1356D 6D51C245 E485B576 625E7EC6 F44C42E9 A63A3620 FFFFFFFF FFFFFFFF
Générateur : 22 (en décimal)
Longueur (en mots de 32 bits) : 1
Données (hex) : 2

Paramètres facultatifs :
Plus grand facteur premier de l'ordre du groupe : 24 (décimal)
Longueur (mots de 32 bits) : 24
Données (hex) : 7FFFFFFF FFFFFFFF E487ED51 10B4611A 62633145 C06E0E68 94812704 4533E63A 0105DF53 1D89CD91 28A5043C C71A026E F7CA8CD9 E69D218D 98158536 F92F8A1B A7F09AB6 B6A8E122 F242DABB 312F3F63 7A262174 D31D1B10 7FFFFFFF FFFFFFFF
Force du groupe : 26 (décimal)
Longueur (mots de 32 bits) : 1
Data (hex) : 00000042

E.2 Groupe bien connu 2 : un nombre premier de 1024 bits

Le nombre premier est 2^1024 - 2^960 - 1 + 2^64 * { [2^894 pi] + 129093 }.
Sa valeur décimale est :
179769313486231590770839156793787453197860296048756011706444423684197180216158519368947833795864925541502180565485980503646440548199239100050792877003355816639229553136239076508735759914822574862575007425302077447712589550957937778424442426617334727629299387668709205606050270810842907692932019128194467627007

La primarité du nombre a été rigoureusement prouvée.

La représentation du groupe en OAKLEY est
Type de groupe : "MODP"
Taille de l'élément champ (en bits) : 1024
Module du nombre premier : 21 (décimal)
Longueur (mots de32 bits) : 32
Données (hex) : FFFFFFFF FFFFFFFF C90FDAA2 2168C234 C4C6628B 80DC1CD1 29024E08 8A67CC74 020BBEA6 3B139B22 514A0879 8E3404DD EF9519B3 CD3A431B 302B0A6D F25F1437 4FE1356D 6D51C245 E485B576 625E7EC6 F44C42E9 A637ED6B 0BFF5CB6 F406B7ED EE386BFB 5A899FA5 AE9F2411 7C4B1FE6 49286651 CE65381 FFFFFFFF FFFFFFFF
Générateur : 22 (décimal)
Longueur (en mots de 32 bits) : 1
Données (hex) : 2

Paramètres facultatifs :
Plus grand facteur premier de l'ordre du groupe : 24 (décimal)
Longueur (mots de 32 bits) : 32
Données (hex) : 7FFFFFFF FFFFFFFF E487ED51 10B4611A 62633145 C06E0E68 94812704 4533E63A 0105DF53 1D89CD91 28A5043C C71A026E F7CA8CD9 E69D218D 98158536 F92F8A1B A7F09AB6 B6A8E122 F242DABB 312F3F63 7A262174 D31BF6B5 85FFAE5B 7A035BF6 F71C35FD AD44CFD2 D74F9208 BE258FF3 24943328 F67329C0 FFFFFFFF FFFFFFFF
Force du groupe : 26 (décimal)
Longueur (mots de 32 bits) : 1
Données (hex) : 0000004D

E.3 Groupe bien connu 3 : une définition de groupe à courbe elliptique

La courbe se fonde sur le champ de Galois GF[2^155] avec 2^155 éléments de champ. Le polynôme irréductible pour le champ est u^155 + u^62 + 1.

L'équation pour la courbe elliptique est Y^2 + X Y = X^3 + A X + B.

X, Y, A, B sont les éléments du champ.

Pour la courbe spécifiée A = 0 et B = u^18 + u^17 + u^16 + u^13 + u^12 + u^9 + u^8 + u^7 + u^3 + u^2 + u + 1.

B est représenté en binaire par la chaîne binaire 1110011001110001111 ; en décimal c'est 471951, et en hex 7338F.

Le générateur est un point (X,Y) sur la courbe (satisfaisant l'équation de la courbe, mod 2 et modulo le polynôme du champ).

X = u^6 + u^5 + u^4 + u^3 + u + 1 et Y = u^8 + u^7 + u^6 + u^3.

Les chaînes binaires pour X et Y sont 1111011 et 111001000 ; en décimal ce sont 123 et 456.

L'ordre de groupe (le nombre de points de la courbe) est 4567192616659071619386556591434463519676923731 6 qui est 12 fois le nombre premier 380599384721589301615546382619538626639743644 3.
(Ce nombre premier a été rigoureusement prouvé.) Le point générateur (X,Y) a pour ordre 4 fois le nombre premier ; le générateur est le triple d'un point de la courbe.

Représentation OAKLEY de ce groupe :

Type de groupe :	"EC2N"
Taille de l'élément champ (bits) :	155
Polynôme champ irréductible :	21 (décimal)
Longueur (mots de 32 bits) :	5
Données (hex) :	08000000 00000000 00000000 40000000 00000001
Générateur :	
Coordonnée X :	22 (décimal)
Longueur (mots de 32 bits) :	1
Données (hex) :	7B
coordonnée Y :	22 (décimal)
Longueur (mots de 32 bits) :	1
Données (hex) :	1C8
Paramètres de courbe elliptique :	
Paramètre A :	23 (décimal)
Longueur (mots de 32 bits) :	1
Données (hex) :	0
Paramètre B :	23 (décimal)
Longueur (mots de 32 bits) :	1
Données (hex) :	7338F

Paramètres facultatifs :

Plus grand facteur premier de l'ordre du groupe :	24 (décimal)
Longueur (mots de 32 bits) :	5
Données (hex) :	00AAAAAA AAAAAAAA AAAAB1FC F1E206F4 21A3EA1B
Ordre du groupe :	25 (décimal)
Longueur (mots de 32 bits) :	5
Données (hex) :	08000000 00000000 000057DB 56985371 93AEF944
Force du groupe :	26 (décimal)
Longueur (mots de 32 bits) :	1
Données (hex) :	0000004C

E.4 Groupe bien connu 4 : Définition de grand groupe à courbe elliptique

Cette courbe se fonde sur le champ de Galois GF[2^185] avec 2^185 éléments de champ. Le polynôme irréductible pour le champ est u^185 + u^69 + 1.

L'équation de la courbe elliptique est Y^2 + X Y = X^3 + A X + B.

X, Y, A, B sont les éléments du champ. Pour la courbe spécifiée, A = 0 et
B = u^12 + u^11 + u^10 + u^9 + u^7 + u^6 + u^5 + u^3 + 1.

B est représenté en binaire par la chaîne binaire 1111011101001 ; en décimal c'est 7913, et en hex 1EE9.

Le générateur est un point (X,Y) sur la courbe (satisfaisant l'équation de la courbe, mod 2 et modulo le polynôme du champ) ; X = u^4 + u^3 et Y = u^3 + u^2 + 1.

Les chaînes binaires pour X et Y sont 11000 et 1101 ; en décimal c'est 24 et 13. L'ordre du groupe (le nombre de points de la courbe) est 4903985730770844346746710485765268224805238500104505311 6, qui est 4 fois le nombre premier 1225996432692711086686677621441317056201309625026126327 9.

(Ce nombre premier a été rigoureusement prouvé.)

Le point générateur (X,Y) a pour ordre 2 fois le nombre premier ; le générateur est le double d'un certain point de la courbe.

La représentation OAKLEY de ce groupe est :
Type de groupe : "EC2N"
Taille de l'élément champ (bits) : 185
Polynôme champ irréductible : 21 (décimal)
Longueur (mots de 32 bits) : 6
Données (hex) : 02000000 00000000 00000000 00000020 00000000 00000001
Générateur :
Coordonnée X : 22 (décimal)
Longueur (mots de 32 bits) : 1
Données (hex) : 18
Coordonnée Y : 22 (décimal)
Longueur (mots de 32 bits) : 1
Données (hex) : D
Paramètres de courbe elliptique :
Paramètre A : 23 (décimal)
Longueur (mots de 32 bits) : 1
Données (hex) : 0
Paramètre B : 23 (décimal)
Longueur (mots de 32 bits) : 1
Données (hex) : 1EE9

Paramètres facultatifs :
Plus grand facteur premier de l'ordre du groupe : 24 (décimal)
Longueur (mots de 32 bits) : 6
Données (hex) : 007FFFFF FFFFFFFF FFFFFFFF F6FCBE22 6DCF9210 5D7E53AF
Ordre du groupe : 25 (décimal)
Longueur (mots de 32 bits) : 6
Données (hex) : 01FFFFFF FFFFFFFF FFFFFFFF DBF2F889 B73E4841 75F94EBC
Force du groupe : 26 (décimal)
Longueur (mots de 32 bits) : 1
Données (hex) : 0000005B

E.5 Groupe bien connu 5 : un nombre premier de 1536 bits

Le nombre premier est 2^1536 - 2^1472 - 1 + 2^64 * { [2^1406 pi] + 741804 }.
Sa valeur en décimal est
24103124269210325885520760221975660748569505485024599426541169419581088316826122288900938582613416146732271414779040121965036489570505826319427307068050092230627347453410734066962460145893616597740410271692494532003787294341703258437786591981437631937768598695240889401955773461198435453015470437472077499697637500843089263392955599688824578724129938101291302945929999479263652640592846472097303849472116814344647144384885209401274598442888593365268963209196339 19

Le caractère premier du nombre a été rigoureusement prouvé.

La représentation du groupe en OAKLEY est
Type de groupe : "MODP"
Taille de l'élément champ (bits) : 1536
Module premier : 21 (décimal)

Longueur (mots de 32 bits) : 48
Données (hex) : FFFFFFFF FFFFFFFF C90FDAA2 2168C234 C4C6628B 80DC1CD1 29024E08 8A67CC74 020BBEA6 3B139B22 514A0879 8E3404DD EF9519B3 CD3A431B 302B0A6D F25F1437 4FE1356D 6D51C245 E485B576 625E7EC6 F44C42E9 A637ED6B 0BFF5CB6 F406B7ED EE386BFB 5A899FA5 AE9F2411 7C4B1FE6 49286651 ECE45B3D C2007CB8 A163BF05 98DA4836 1C55D39A 69163FA8 FD24CF5F 83655D23 DCA3AD96 1C62F356 208552BB 9ED52907 7096966D 670C354E 4ABC9804 F1746C08 CA237327 FFFFFFFF FFFFFFFF
Générateur : 22 (décimal)
Longueur (mots de 32 bits) : 1
Données (hex) : 2

Paramètres facultatifs :
Plus grand facteur premier de l'ordre du groupe : 24 (décimal)
Longueur (mots de 32 bits) : 48
Données (hex) : 7FFFFFFF FFFFFFFF E487ED51 10B4611A 62633145 C06E0E68 94812704 4533E63A 0105DF53 1D89CD91 28A5043C C71A026E F7CA8CD9 E69D218D 98158536 F92F8A1B A7F09AB6 B6A8E122 F242DABB 312F3F63 7A262174 D31BF6B5 85FFAE5B 7A035BF6 F71C35FD AD44CFD2 D74F9208 BE258FF3 24943328 F6722D9E E1003E5C 50B1DF82 CC6D241B 0E2AE9CD 348B1FD4 7E9267AF C1B2AE91 EE51D6CB 0E3179AB 1042A95D CF6A9483 B84B4B36 B3861AA7 255E4C02 78BA3604 6511B993 FFFFFFFF FFFFFFFF
Force du groupe : 26 (décimal)
Longueur (mots de 32 bits) : 1
Données (hex) : 0000005B

Appendice F Mise en œuvre du fonctionnement de groupe

Le fonctionnement du groupe doit être mis en œuvre comme une séquence d'opérations arithmétiques ; les opérations exactes dépendent du type de groupe. Pour les groupes d'exponentiation modulaire, l'opération est une multiplication d'entier et de restes multi-précision par le modulo du groupe. Voir dans le volume 2 de Knuth [Knuth] un exposé sur la façon de mettre en œuvre cela pour de grands entiers. Les recommandations de mise en œuvre pour les opérations sur les groupes de courbe elliptique sur GF[2^N] sont décrites dans [Schroeppel].

Bibliographie

[RFC1750] D. Eastlake, 3rd et autres, "Recommandations d'aléa pour la sécurité", décembre 1994. *(Info., remplacée par la* RFC4086)

[RFC2065] D. Eastlake 3rd, C. Kaufman, "Extensions de sécurité du système de noms de domaines", janvier 1997. (*Obsolète, voir* RFC2535) *(P.S.)*

[RFC2401] S. Kent et R. Atkinson, "Architecture de sécurité pour le protocole Internet", novembre 1998. (*Obsolète, voir* RFC4301)

[RFC2402] S. Kent et R. Atkinson, "En-tête d'authentification IP", novembre 1998. (*Obsolète, voir* RFC4302, 4305)

[RFC2406] S. Kent et R. Atkinson, "Encapsulation de charge utile de sécurité IP (ESP)", novembre 1998. *(Obsolète, voir* RFC4303*)*

[RFC2408] D. Maughan, M. Schertler, M. Schneider et J. Turner, "Association de sécurité Internet et protocole de gestion de clés (ISAKMP)", novembre 1998. *(Obsolète, voir la* RFC4306 *puis la* RFC5996)

[Blaze] Blaze, Matt et al., "Minimal key lengths for symmetric ciphers to provide adequate commercial security". Rapport d'un groupe ad hoc de cryptographes et d'informaticiens... -- http://www.bsa.org/policy/encryption/cryptographers.html

[STS] W. Diffie, P.C. Van Oorschot, et M.J. Wiener, "Authentication and Authenticated Key Exchanges," dans Designs, Codes et Cryptography, Kluwer Academic Publishers, 1992, pp. 107

[Kocher] Kocher, Paul, "Timing Attack", http://www.cryptography.com/timingattack.old/timingattack.html

[Knuth] Knuth, Donald E., "The Art of Computer Programming, Vol. 2, Seminumerical Algorithms", Addison Wesley, 1969.

[Krawcyzk] Krawcyzk, Hugo, "SKEME: A Versatile Secure Key Exchange Mechanism for Internet", ISOC Secure Networks and Distributed Systems Symposium, San Diego, 1996

[Schneier] Schneier, Bruce, "Applied cryptography: protocols, algorithms, and source code in C", Seconde édition, John Wiley & Sons, Inc. 1995, ISBN 0-471-12845-7, relié. ISBN 0-471-11709-9, broché.

[Schroeppel] Schroeppel, Richard, et al.; "Fast Key Exchange with Elliptic Curve Systems", Crypto '95, Santa Barbara, 1995. Disponible à ftp://ftp.cs.arizona.edu/reports/1995/TR95-03.ps (et .Z).

[Stinson] Stinson, Douglas, "Cryptography Theory et Practice". CRC Press, Inc., 2000, Corporate Blvd., Boca Raton, FL, 33431-9868, ISBN 0-8493-8521-0, 1995

[PGP] Philip Zimmermann, "The Official Pgp User's Guide", Publié par MIT Press Trade, date de publication : juin 1995, ISBN: 0262740176

Déclaration complète de droits de reproduction