La gestion des communications sécurisées

Bernard Cousin

UNIVERSITE DE RENNES 1

Plan

- La gestion des communications sécurisées
- ISAKMP
  - Principes
  - Les messages
  - Le protocole : les modes d'échange
- IKE

Gestion des communications sécurisées

- Pour établir une communication sécurisée :
  - Une SA doit exister
  - Des clefs doivent avoir été échangées/générées
  - La SADB doit être remplie avec la SA
- Deux méthodes de gestion des clefs :
  - Manuelle
    - Pour les environnements petits et statiques
  - Automatique
    - Création des clefs de la SA "on-demand"
    - Dans le cadre d'un système réparti et en évolution

Gestion des clefs

- Type d'échange des clefs
  - Distribution de clefs ou
  - Génération de clefs
- Types de clefs
  - Partagées ou
  - Publiques
- Entre les deux entités :
  - Directement
  - Ou à l'aide d'un tiers

Gestion des communications de sécurité

- Deux entités peuvent avoir des besoins extrêmement variés, d'établir entre elles simultanément de nombreuses communications sécurisées.
  - Un grand nombre d'applications aux besoins variés
  - Un grand nombre de services de sécurité
  - Un grand nombre de phases de négociation des méthodes et des paramètres rendant ses services
  - Une gestion d'un grand nombre de clefs

ISAKMP

- "Internet Security Association and Key Management Protocol" (ISAKMP)
  - Rfc 2408 (nov 1998)
    - UDP port 500
  - Un cadre général
  - Inspiré de
    - "Oakley Key Determination Protocol" (rfc 2412)
      - Hilarie Orman – university of Arizona (1996)
    - SKEME ("Versatile secure key exchange protocol for key management ")
      - Hugo Krawczyk – IBM (1996)
  - Utilisé par IKE (rfc 2409)

Historique

- Oakley : "Key Determination Protocol"
  - Génération de clefs partagées basée sur Diffie-Hellman
  - Propose des services additionnels de sécurité
  - N'utilise pas de formats spécifiques
- SKEME : "key exchange protocol"
  - Permet de générer des clefs à partir d'informations authentifiées
  - Négociation de la politique de sécurité
  - Utilise Diffie-Hellman si le "Perfect Forward Secrecy" est nécessaire
  - Renouvellement rapide de clefs (sans PFS)

Diffie Hellman

- Produit une clef partagée : K
  - $K = (a^X)^Y \bmod q = (a^Y)^X \bmod q$
  - Nécessite aucune infrastructure pré existante
  - Utilisable à volonté : renouvellement
  - Sans chiffrement
- Inconvénients
  - Ne fournit aucune information sur l'identité des entités communicantes
  - Fragile vis-à-vis d'attaques de type "man-in-the-middle" (pas authentification des entités)
  - Calcul intensif (sensible aux attaques de type DOS)

Les caractéristiques d'Oakley (reprises par ISAKMP)

- Résistant aux attaques :
  - Utilise des "cookies" contre les attaques de type DOS
  - Utilise des "nonces" contre les attaques de type "replay"
  - Authentification des entités contre les attaques de type MiM
- Adaptable :
  - Négociation des paramètres du Diffie-Hellman ("group")
  - Flexibilité des méthodes d'authentification et d'échange des clefs
  - Négociation des algorithmes d'authentification et de chiffrement
  - Plusieurs formes d'échange
    => ISAKMP

Les "cookies"

- Chaque entité envoie un nombre pseudo-aléatoire (le "cookie")
  - Ne nécessite pas de stockage, peut être vérifiée localement
- L'autre entité doit renvoyer ce même cookie
  - Si l'adresse de l'autre entité est fausse il n'y aura pas de réponse
  - L'autre entité (l'attaquant) est obligée de travailler un peu et de (pour)suivre son attaque
- La première entité ne calcule rien tant qu'elle n'a pas reçu de réponse et vérifié le cookie

La génération du "cookie"

- Exigences
  - Chaque valeur de "cookie" doit être spécifique à chaque entité, mais cependant impossible à deviner par un attaquant
  - Il est impossible à un attaquant de générer un "cookie" qui sera accepté par la victime
  - La génération et la vérification du "cookie" est rapide
- Solution
  - Fonction de hachage rapide
    - MD5
  - Des paramètres locaux bien choisis
    - "IP source and destination adresses" + "UDP source and destination port numbers" + un secret dont la valeur est générée localement + "timestamp"

Exemple d'un échange typique d'Oakley (de type agressif)

1. **I => R:** CKY_I, OK_KEYX, GRP, g^x, EHAO, NIDP, ID_I, ID_R, N_I, $S_{KI}[ID_I \,||\, ID_R \,||\, N_I \,||\, GRP \,||\, g^x \,||\, EHAO]$
2. **R => I:** CKY_R, CKY_I, OK_KEYX, GRP, g^y, EHAS, NIDP, ID_R, ID_I, N_R, N_I, $S_{KR}[ID_R \,||\, ID_I \,||\, N_R \,||\, N_I \,||\, GRP \,||\, g^y \,||\, g^x \,||\, EHAS]$
3. **I => R:** CKY_I, CKY_R, OK_KEYX, GRP, g^x, EHAS, NIDP, ID_I, ID_R, N_I, N_R, $S_{KI}[ID_I \,||\, ID_R \,||\, N_I \,||\, N_R \,||\, GRP \,||\, g^x \,||\, g^y \,||\, EHAS]$

Notation:

I, R = Initiator, Responder

= CKY_I, CKY_R = Initiator, responder cookies

OK_KEYX = Key exchange message type

GRP = Name of Diffie-Hellman group for this exchange

g^x, g^y = Public key of initiator, responder; g^{xy} = session key from this exchange

EHAO, EHAS = Encryption, hash, authentication functions, offered and selected

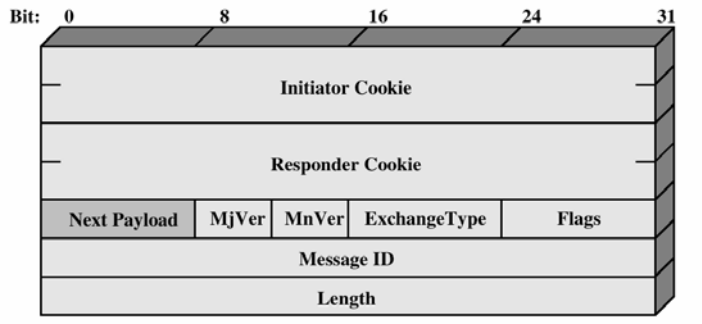
NIDP = Indicates encryption is not used for remainder of this message

ID_I, ID_R = Identifier for initiator, responder

N_I, N_R = Random nonce supplied by initiator, responder for this exchange

$S_{KI}[X]$, $S_{KR}[X]$ = Indicates the signature over X using the private key (signing key) of initiator, responder

Format des messages ISAKMP



(a) ISAKMP Header

(b) Generic Payload Header

Figure 6.12 ISAKMP Formats

13

Champs des messages ISAKM

- "Initiator cookie" :
  - Déterminé par l'entité qui initialise l'établissement du SA; idem "SA notification" ou "SA deletion"
- "Responder cookie" :
  - Déterminé par l'entité répondant
    - nul dans le premier message
- "Exchange type" et "Next payload" :
  - Cf. la suite
- "Flags" :
  - "Encryption bit" : toutes les "payloads" suivant l'entête sont chiffrées
  - "Commit bit" : assure que le SA est bien établi avant de l'utiliser pour transmettre des données sécurisées
- "Message ID" :
  - Identifie de manière unique le message
- "Length" :
  - Longueur totale du message (entête + "payloads") en octets

Les types de "Payload"

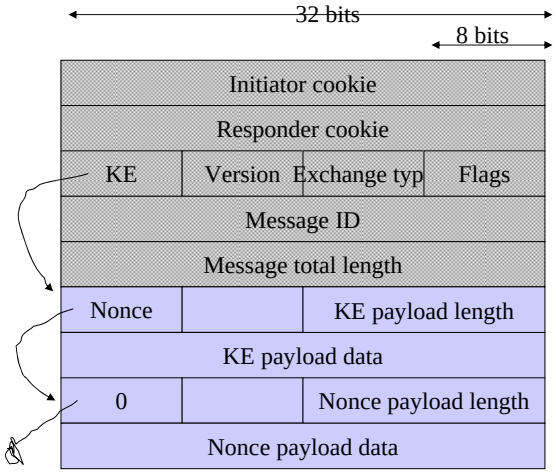
- Security association payload (SA)
- Proposal payload (P: encapsulated in a SA paylaod)
- Transform payload (T: encapsulated in a SA paylaod)
- Key exchange payload (KE)
- Identification payload (ID)
- Certificate payload (CERT)
- Certificate request payload (CR)
- Hash payload (HASH)
- Signature payload (SIG)
- Nonce payload (NONCE)
- Notification payload (N)
- Delete payload (D)
- Vendor ID payload
- Etc.

Les types de "Payload"

Type	Paramètres	Description
Security Association (SA)	Domain of Interpretation, Situation	Used to negotiate security attributes and indicate the DOI (IPsec or other) and Situation under which negotiation is taking place.
Proposal (P)	Proposal #, Protocol-ID, SPI Size, # of Transforms, SPI	Used during SA negotiation; indicates protocol to be used and number of transforms.
Transform (T)	Transform #, Transform-ID, SA Attributes	Used during SA negotiation; indicates transform and related SA attributes.
Key Exchange (KE)	Key Exchange Data	Supports a variety of key exchange techniques.
Identification (ID)	ID Type, ID Data	Used to exchange identification information.

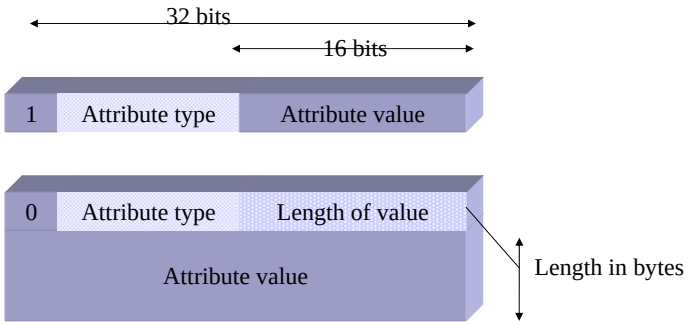
Type	Paramètres	Description
Certificate (CERT)	Cert Encoding, Certificate Data	Used to transport certificates and other certificate-related information.
Certificate Request (CR)	# Cert Types, Certificate Types, # Cert Auths, Certificate Authorities	Used to request certificates; indicates the types of certificates requested and the acceptable certificate authorities.
Hash (HASH)	Hash Data	Contains data generated by a hash function.
Signature (SIG)	Signature Data	Contains data generated by a digital signature function.
Nonce (NONCE)	Nonce Data	Contains a nonce.
Notification (N)	DOI, Protocol-ID, SPI Size, Notify Message Type, SPI, Notification Data	Used to transmit notification data, such as an error condition or status.
Delete (D)	DOI, Protocol-ID, SPI Size, # of SPIs, SPI (one or more)	Indicates a SA that is no longer valid.

Le chaînage des "payloads"



Les attributs d'une "payload"

- Deux formats :



La "payload" de type SA

- Peux contenir plusieurs "payloads" de type "proposal". Ce qui permet une négociation :
  - "Matching Proposals' #" : ET logique
  - "Differing Proposals' #" : Ou logique
- "Proposals" (P_i) et "Transforms" (T_i) sont choisis par la politique de sécurité en utilisant la "Security Policy Database"

-P1 : AH
  - T1 : HMAC-SHA
  - T2 : HMAC-MD5

-P2 : ESP
  - T1 : 3DES with HMAC-SHA
  - T2 : 3DES with HMAC-MD5
  - T3 : DES with HMAC-SHA
  - T4 : DES with HMAC-MD5

-P3 : ESP
  - T1 : 3DES with HMAC-SHA
  - T2 : 3DES with HMAC-MD5
  - T3 : DES with HMAC-SHA
  - T4 : DES with HMAC-MD5

-P3 : PCP
  - T1 : LZS
  - T2 : Deflate

La "palyload" de type "Proposal"

- Indique le protocole de ce SA pour lequel les services et les mécanismes sont négociés
- Le SPI de l'entité émettrice + nombre de "transforms"
  - Chaque "transform" est décrit par un "payload" de type " transform"
- L'Initiateur peut proposer plusieurs "transforms"
- Le Répondeur doit en choisir un ou rejeter l'offre

La "payload" de type "Transform"

- Le paramètre "Tranform#" identifie cette "payload" particulière
- Les champs "Transform ID" and "Attribute" décrivent un "transform" :
  - 3DES for ESP + HMAC(SHA-1) for AH + hash length + etc.

La "payload" de type "Key exchange"

- Définie la technique d'échange des clefs :
  - Oakley
  - Diffie-Hellman
  - PGP basé sur RSA
- Contient les données nécessaires à la génération de la clef de session (dépend de l'algorithme d'échange)

La "payload" de type Identification

- Détermine l'identité des entités communicantes
- Peut être utilisée pour déterminer l'authenticité des informations :
  - Par exemple le champ "ID Data" contient une adresse IP

La payload de type "Certificate"

- Transporte un certificat de clef publique
- Le champ "Certificate Encoding" définie le type du certificat ou information relative au certificat :
  - PKCS#7 wrapped X.509 certificate
  - PGOP certificate
  - DNS signed key
  - X.509 certificate – signature
  - X.509 certificate – key exchange
  - Kerberos tokens
  - Certificate Revocation List
  - Authority Revocation List
  - SPKI certificate
- Durant un échange ISAKMP, à tout moment, une entité peut (re-)demander le certificat de l'autre : "Certificate request"
- Plus d'un type de certificat (d'une autorité de certification) peut être listé dans une "payload" de type "Certificate"

La payload de type "Notification"

- Informe d'une erreur ou de l'état de la session
  - Error
    - Invalid payload type, DOI not supported, situation not supported, Invalid Cookie, invalid major version, invalid minor version, invalid exchange type, invalid Flags, invalid Message ID, invalid Protocol ID, Invalid SPI, invalid transform ID, attributes not supported, no proposal chosen, bad proposal syntax, payload malformed, invalid key information, invalid CERT encoding, Invalid certificate, Bad Cert Request syntax, invalid Cert Authority, invalid hash information, authentication failed, invalid signature, address notification
  - Status
    - Connected (ISAKMP), Responder-Lifetime (IPsec) : durée du SA, Replay-status (IPsec) : confirme la mise en oeuvre d'un mécanisme anti-rejeu, Initial-Contact (IPsec) : les SA établies avec le même système distant et préexistantes doivent être détruites

Les types d'échanges ISAKMP

- Cinq types d'échanges, par défaut.
  - "Base Exchange"
  - "Identity Protection Exchange"
    - Utilisé pour IPsec
  - "Authentification Only Exchange"
  - "Agressive Exchange"
    - Utilisé pour IPsec
  - "Informational Exchange"
- D'autres protocoles utilisent les mêmes types d'échanges :
  - Par ex. TLS

Echange de base

- Echange de clefs + authentification
- Peu de messages mais n'offre pas de protection des identités
  - Les 2 premiers messages utilise des "cookies" et un SA est établi pour un des protocoles et une des transformations négociés
    - utilise des nonces contre les attaques de type rejeu
    - SA représente la "payload" de type SA et les "payloads" de type "Proposal" et "Transform" associées
  - Les 2 derniers messages échangent des données permettant la génération des clefs, l'échange de l'identité des partenaires, celles authentifiant les clefs, les identités et les nonces des 2 premiers messages

(a) Base Exchange

(1) **I** => **R:** SA; NONCE	Begin ISAKMP-SA negotiation
(2) **R** => **I:** SA; NONCE	Basic SA agreed upon
(3) **I** => **R:** KE; ID_I; AUTH	Key generated; Initiator identity verified by responder
(4) **R** => **I:** KE; ID_R; AUTH	Responder identity verified by initiator; Key generated; SA established

Echange protégeant les identités

- Etend l'échange de base en protégeant l'identité des entités
  - Les 2 premiers messages : établissement du SA
  - Les 2 suivants : l'échange des clefs, avec des nonces.
  - Les 2 derniers : des messages chiffrés avec les clefs de session. Ils contiennent des données d'authentification : par ex. signature digitale et optionnellement des certificats authentifiant les clefs publiques

(b) Identity Protection Exchange

(1) **I** => **R:** SA	Begin ISAKMP-SA negotiation
(2) **R** => **I:** SA	Basic SA agreed upon
(3) **I** => **R:** KE; NONCE	Key generated
(4) **R** => **I:** KE; NONCE	Key generated
(5)* **I** => **R:** ID_I; AUTH	Initiator identity verified by responder
(6)* **R** => **I:** ID_R; AUTH	Responder identity verified by initiator; SA established

Notation: * = signifies payload encryption after the ISAKMP header

Echange avec seulement une authentification

- Permet l'authentification mutuelle sans échange de clefs
  - les 2 premiers messages : établissement du SA
  - Le second, transporte l'identité du Répondeur
  - Le troisième, celui de l'Initiateur

(c) Authentication Only Exchange

(1) **I** => **R:** SA; NONCE	Begin ISAKMP-SA negotiation
(2) **R** => **I:** SA; NONCE; ID_R; AUTH	Basic SA agreed upon; Responder identity verified by initiator
(3) **I** => **R:**ID_I; AUTH	Initiator identity verified by responder; SA established

Echange agressif

- Minimise le nombre de messages
- Pas de protection des identités
- Négociation limitée :
  - Ne permet pas de choisir parmi les différents groupes proposés pour le Diffie-Hellman
  - Si l'authentification s'appuie sur les nonces, on ne peut pas offrir le choix entre différents algorithmes de chiffrement ou hachage.

(d) Aggressive Exchange	
(1) **I** => **R:** SA; KE; NONCE; ID_I	Begin ISAKMP-SA negotiation and key exchange
(2) **R** => **I:** SA; KE; NONCE; ID_R; AUTH	Initiator identity verified by responder; Key generated; Basic SA agreed upon
(3)* **I** => **R:** AUTH	Responder identity verified by initiator; SA established

Notation: * = signifies payload encryption after the ISAKMP header

Echange d'information

- Une transmission unidirectionnelle permettant la gestion du SA

(e) Informational Exchange	
(1)* **I** => **R:**N/D	Error or status notification, or deletion

IKE

- "Internet Key Exchange" (IKE)
  - Rfc 2409
    - UDP port 500
  - S'appuie sur le cadre général ("framework") d'ISAKMP
    - "Internet Security Association and Key Management Protocol"
  - Buts d'IKE :
    - Etablir une association de sécurité :
      - Partager les paramètres de sécurité
      - Les clefs d'authentification
    - Un protocole général d'échange de clefs pour la sécurisation
      - Négociation de l'ensemble des contextes de sécurité applicables
        - » L'ensemble des méthodes de sécurité (chiffrement, authentification, etc.) communes aux 2 entités
      - Détermination des clefs authentifiées
      - Utilisé par IPsec (DOI: rfc 2407), SNMPv3, OSPFv2, RIPv2

Coopération typique entre IPsec et IKE

- Peer P detects that traffic wishes to use an IPsec tunnel to peer Q.
- Peer P initiates an IKE SA with peer Q.
- A transform set is issued by peer P for the IKE SA
- A transform set is issued by peer Q for the IKE SA
- On agreement of the transform set, peer P sends their digital certificate across the IKE SA
- Peer Q sends their digital certificate across the same bi-directional IKE SA.
- Peer P and Peer Q then exchange Diffie-hellman numbers that are digitally signed so that they can establish a shared key that they can be confident genuinely comes from the other peer.
- Peers P and Q verify each other's signature using each other's public key.
- The shared key is calculated using the Diffie-Hellman numbers.
- The IKE SA is now established
- Peer P sends a transform set on the IKE SA.
- Peer Q sends a transform set on the IKE SA and peers P and Q decide the lowest common set.
- Peer P now initiates a uni-directional IPsec SA for data transfer.
- If Peer Q needs to send data then peer Q initiates its own uni-directional IPsec SA.
- The Diffie-Hellman key exchange is used again for P and Q to negotiate a shared key, because IPsec is being used, the shared key derived at this stage is totally independent of the IKE shared key.
- Data is now sent over the IPsec SAs.
- As the IPsec SAs near expiration as defined by their timers, IKE creates new IPsec SAs and data transfer continues seamlessly.

Les différentes phases d'IKE

- La première phase
  - Etablissement d'une communication sécurisée et authentifiée.
    - En 2 étapes :
      - Etablissement d'un secret partagé (basé sur Diffie-Hellman)
      - Authentification (5 méthodes proposées)
    - Utilise l'une des 2 types d'échange :
      - "Identity protection exchange"
      - "Agressive exchange"
- La deuxième phase
  - La négociation des services de sécurité pour IPsec, RIPv2, OSPFv2, etc.
    - "Quick-mode exchange"
    - Les clefs sont soit dérivées de la clefs de session IKE, soit (pour offrir un "Perfect Forward Secrecy") générées en utilisant Diffie-Hellman
  - Plusieurs autres services de sécurité peuvent être négociés successivement ou simultanément en utilisant le même IKE SA
- D'autres phases supplémentaires
  - "Informational exchange"
  - "New group exchange" (cf. plus loin)

Un contexte de sécurité

- Un contexte de sécurité ("Protection Suite") défini :
  - L'algorithme de chiffrement
  - L'algorithme de hachage
  - La méthode d'authentification
  - Le groupe Diffie-Hellman (cf. plus loin)
  - La durée de vie du SA (optionnel)
- IKE négocie durant la première phase, le sous-ensemble des contextes de sécurité partagés par les entités et applicables

Les méthodes d'authentification d'IKE

- Plusieurs méthodes d'authentification peuvent être proposées :
  1. Pre-shared key (clef partagée)
  2. Signatures numériques avec fonction de hachage puis chiffrement DSA (clef publique)
  3. Signatures numériques avec fonction de hachage puis chiffrement RSA (clef publique)
  4. Nonce chiffré avec DSA (clef publique)
  5. Nonce chiffré avec RSA (clef publique)

Les paramètres de Diffie-Hellman

Il existent de nombreuses variantes de D-H
- Certaines valeurs particulières des paramètres D-H sont sélectionnées
- Un seul code permet de représenter l'ensemble des valeurs des paramètres choisies pour une variante de D-H

- On définit ainsi la notion de groupe D-H
- Les groupes D-H sont:
  1. Exponentiation over a prime modulus – 768 bits
  2. Exponentiation over a prime modulus – 1024 bits
  3. Exponentiation over a prime modulus - 1680 or 1536 bits (to be determined)
  4. Elliptic curve group over 155-bit field
  5. Elliptic curve group over 185-bit field
  - Le groupe 1 est obligatoire
  - Les groupes 1 et 4 (les groupes 2 et 5) proposent approximativement le même niveau de sécurité
- En utilisant l'IKE "New Group Exchange" deux entités peuvent déterminer leur propre groupe D-H

La première phase d'IKE

- A l'issue de la première phase, les deux entités partagent un secret :
  - SKEY
- Les autres clefs peuvent être déduites de celui-ci :
  - SKEY_d : pour générer d'autres clefs (par ex. pour IPsec)
  - SKEY_a : pour fournir et vérifier l'intégrité et l'authentification de la source des données
  - SKEY_e : pour chiffrer les messages d'IKE

La génération de la SKEY

- "Preshared key authentication" :
  - SKEY = PRF(preshared-key, N_I | N_R)
- "Signature authentication" :
  - SKEY = PRF(N_I | N_R, g^{xy})
- "Encrypted nonce authentication"
  - SKEY = PRF(hash (N_I | N_R, CKY_I | CKY_R)

  où PRF ("Pseudo-random function") est une fonction de hachage négociée : généralement HMAC

Génération des autres clefs

- Les SKEY_x sont générées à partir de la SKEY, ainsi :
  - SKEY_d = PRF(SKEY, $g^{xy,} | CKY_I | CKY_R | 0$)
  - SKEY_a = PRF(SKEY, SKEY_d | $g^{xy,} | CKY_I | CKY_R | 1$)
  - SKEY_e = PRF(SKEY, SKEY_a | $g^{xy,} | CKY_I | CKY_R | 2$)
- Concaténation de SKEY_e
  - Nécessaire quand la sortie de la fonction PRF est trop petite pour la largeur de la clef de chiffrement :
    - K = K1|K2|K3
    - K1 = PRF(SKEY, 0)
    - K2 = PRF(SKEY, K1| 1)
    - K3 = PRF(SKEY, K2| 2)
- Pour la génération des vecteurs d'initialisations
    - par ex. quand IKE utilise le mode CBC
  - IV_0 = PRF(SKEY, $g^x | g^y$)
  - IV_n = le dernier bloc chiffré lors du chiffrement/déchiffrement précédant (n-1ème)

Calcul d'un résumé ("Digest")

- Hash_I = PRF(SKEY, $g^i | g^r | CKY_I | CKY_R |$SA-offer$| ID_I$)
- Hash_I = PRF(SKEY, $g^r | g^i | CKY_R | CKY_I |$SA-offer$| ID_R$)
- SA-offer est la totalité de la "payload" de type SA
  - Résiste aux attaques de type MiM
    - Par exemple empêche l'attaquant de modifier une négociation qui proposait "3DES ou DES" en "DES".

Bidirectionnalité

- Les SA d'IKE sont bidirectionnelles :
  - Durant l'établissement du SA, une entité est l' Initiateur, l'autre le Répondeur, mais …
  - Une fois le SA établie, la communication sécurisée est bidirectionnelle !
- Par contre les SA d'IPsec sont unidirectionnelles !
  - Il en faut donc deux pour définir un canal bidirectionnel d'échanges de données
  - Cependant, chaque entité peut prendre l'initiative de lancer la deuxième phase d'IKE.