Sujets des exercices dirigés

Sécurité et Réseaux

UE RSX 112

2007-2008

G. Florin, S. Natkin

Table des matières

CHAPITRE 1 : FONCTIONS CRYPTOGRAPHIQUES

Exercice 1 : Chiffres à clés secrètes; cryptogramme 1

1) Cassez le cryptogramme suivant qui utilise une substitution monoalphabétique. Le texte en clair ne contient que des lettres. C'est un extrait d'une fable de La Fontaine.

gcxobwryv ib ogx tb syiib
ypsyxg ib ogx tbv qmgzev
t cpb wgqrp wrox qysyib
g tbv obiybwv t roxrigpv
vco cp xgeyv tb xcojcyb
ib qrcsbox vb xorcsg zyv
ab igyvvb g ebpvbo ig syb
jcb wyobpx qbv tbcd gzyv
ib obfgi wcx wrox mrppbxb
oybp pb zgpjjcgyx gc wbvxyp

2) En supposant que vous disposiez d'un histogramme de la répartition des lettres et des digrammes dans la langue française ainsi qu'une fonction Nombre_fautes(texte) qui donne (selon un correcteur orthographique) le pourcentage de mots dans un texte qui ont au moins une faute d'orthographe, imaginez un algorithme de cryptanalyse de ce type de crypto-systèmes

Répartition des lettres dans l'exemple
et dans la langue française

Exemple			Français	
lettre	pourcentage	Nb lettre	lettre	pourcentage
h	0,0	0	k	0,0
k	0,0	0	w	0,1
l	0,0	0	z	0,1
n	0,0	0	j	0,2
u	0,0	0	y	0,3
a	0,5	1	x	0,5
d	0,5	1	h	0,7
f	0,5	1	b	0,8
m	1,0	2	q	0,8
e	1,5	3	f	1,1
j	2,0	4	g	1,2
z	2,0	4	v	1,3
q	2,5	5	m	2,3
s	3,0	6	p	2,9
t	3,4	7	c	3,4
w	3,9	8	d	4,3
r	4,4	9	o	5,2
i	5,9	12	u	5,2
c	6,4	13	l	6,0
p	6,4	13	r	6,4
o	7,4	15	i	7,2
v	7,9	16	t	7,4
x	7,9	16	n	7,7
y	8,4	17	a	8,1
g	8,9	18	s	8,9
b	15,8	32	e	17,7
Total	100,0	203		99,9

Exercice 2 : Chiffres à clés secrètes; cryptogramme 2

Casser le cryptogramme suivant qui utilise une transposition par colonnes. Le texte en clair est issu d'un ouvrage classique sur l'informatique; on peut donc supposer que le mot "ordinateur" y figure. Le texte ne contient que des lettres (sans espace). Il est découpé en blocs de 5 caractères pour plus de lisibilité.

ntsus ueire eibps etsio ootuu rpmrn eaicq iunps cnlog
euern lndur raose xnntu dnaeo eseue clton nretd trels

Exercice 3 : Chiffres à clés secrètes; cryptogramme 3

1) Chiffrer avec le chiffre de Vigenère le texte suivant "textesecretadecoder" en utilisant comme clef le mot crypto.

2) Pour le même texte en clair on obtient le texte chiffré suivant "brqksmzcspxiqxtcxzr" .Quel est la clef ?

3) Supposons que vous disposiez d'un texte en clair et une partie du même texte chiffré, mais que ce texte soit plus court que la clef (par exemple vous ne connaissez que brq dans l'exemple précédent). Quelle information cela vous apporte t'il ? Imaginez des stratégies de cryptanalyse dans le cas ou la clef est un mot français et dans le cas ou c'est une suite aléatoire de lettres. Comment distinguer à priori ces deux cas ?

Exercice 4 : Fonctionnement du chiffre à clé publique RSA

On rappelle les principes du chiffre RSA (Rivest, Shamir, Adleman, 1978) qui a aussi été appelé chiffre du MIT ('Massachussetts Institute of Technology') ou se trouvaient les créateurs du chiffre.

1- Choisir deux nombres p et q premiers et grands ($p, q > 10^{100}$)

2 - Calculer n: = p * q z: = (p - 1) * (q - 1)

3 - Soit d un nombre premier avec z (z et d sont premiers entre eux).

4 - Déterminer un entier e tel que e * d = 1 mod z .

Soit A un message binaire. Découper A en blocs de k bits tel que k soit le plus grand entier tel que $2^k < n$.

Soit P un bloc de k bits, le codage de P est donné par :

$$C = E(P) = P^e \pmod n$$

Le déchiffrement d'un message chiffré C est donné par:

$$P = D(C) = C^d \pmod n$$

On donne les valeurs numériques suivantes : p = 3, q = 11 (trop petites en pratique, mais traitable en exercice).

1) Calculer les valeurs des nombres d et e vérifiant les conditions de l'algorithme du MlT. Pour avoir un couple unique on prend la plus petite valeur possible de d et pour cette valeur la plus petite valeur possible de e.
Quelle est la clé publique et quelle est la clé secrète?

2) Soit le message de 3 chiffres 1, 6, 15 soit par blocs de 5 bits la configuration de bits suivante:
00001 00110 01111
Coder ce message en utilisant les paramètres de chiffrement RSA précédents.

3) On reçoit le message suivant par blocs de 6 bits (4 , 14 , 24):
000100 001110 011000
Donner la valeur initiale du message (texte en clair), en prenant les mêmes valeurs pour d, e et k qu'à la question 2.

4) Pourquoi ne peut-on prendre p et q petits ?
Que se passe-t-il lorsque p et q sont de l'ordre de $l0^{l0}$?

Exercice 5 : Décryptage du RSA

On constate qu'un utilisateur du chiffre RSA a comme clé publique le couple (7,143) soient e = 7 et n = 143. Sachant que ces valeurs sont trop petites pour conférer la mondre sécurité au chiffre RSA, on veut casser ce chiffre, c'est-à-dire retrouver la clé secrète à partir de la connaissance de la clé publique.

1) On doit tout d'abord retrouver $z = \phi(n) = (p-1)(q-1)$ la fonction d'Euler de l'entier n. Pourquoi ?

2) Quelle est la valeur numérique de z pour les valeurs numériques prises en exemple dans ce problème?

3) Définissez une méthode pour retrouver la clé privée d à partir de e et de $z = \phi(n)$?

4) Quelle est la valeur numérique de d ?

5) Pourquoi a t'on pu casser facilement ce chiffre ?

Exercice 6 : Signatures RSA

Un attaquant souhaiterait désorganiser le fonctionnement d'une communication par messages protégés par des signatures RSA (des signatures qui utilisent comme algorithme de chiffrement à clé publique le chiffre RSA).

1) Rappelez les principes généraux du chiffre RSA ?

2) Rappelez les objectifs d'une signature (quelles sont les propriétés attendues d'une signature numérique pour un document ou un message dans l'idéal) ?

3) Rappelez les principes de fonctionnement de la signature RSA ?

4) En observant le fonctionnement du chiffre RSA, l'attaquant conjecture que le produit des chiffres RSA de deux messages M_1 et M_2 est le chiffre du produit des deux messages (on utilise comme produit la multiplication des entiers modulo n). Autrement dit $RSA(M_1 * M_2) = RSA(M_1) * RSA(M_2)$. Montrez que cette propriété est effectivement vraie.

5) A partir de la remarque précédente, l'attaquant se dit que s'il peut écouter deux messages correctement signés par leur émetteur au moyen d'une signature RSA, il peut générer un message produit des deux messages écoutés auquel il adjoindra une signature qui sera le produit des signatures des deux messages écoutés. Est-ce que l'attaquant réussira à désorganiser les communications en faisant accepter le message qu'il a forgé (le destinataire acceptera le message en le considérant comme correctement signé) ? (2 points)

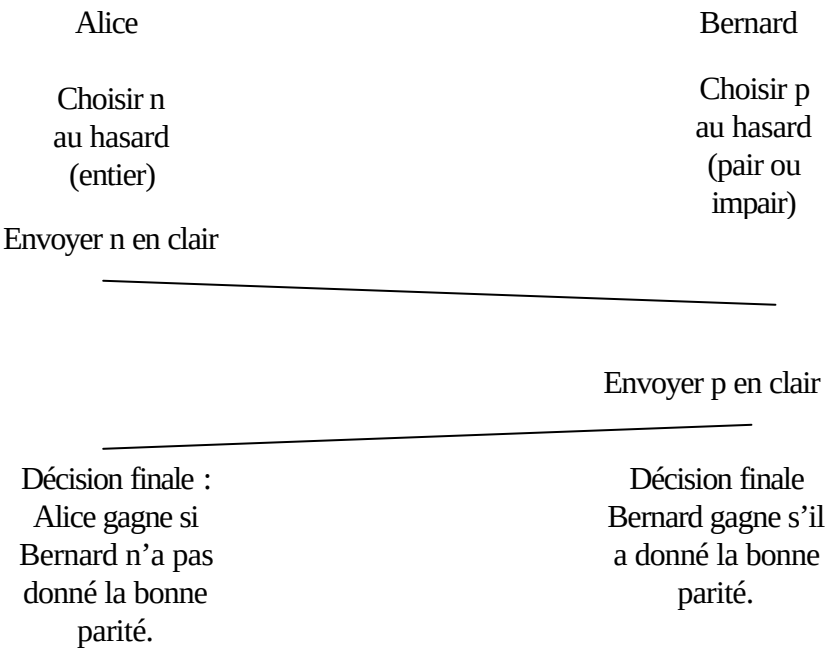
Exercice 7 : Etude du jeu de pile ou face en réseau

Alice et Bernard communiquent à distance au moyen d'un réseau (par messages asynchrones ou tout autre mode de communications par messagerie textuelle, par téléphone etc …). Ayant un différent relativement à une décision, ils veulent s'en remettre au hasard pour la décision finale. Pour cela ils doivent construire un protocole par échange de messages asynchrones qui réalise une décision aléatoire de même nature que celle de pile ou face lorsque deux personnes sont en présence.

La décision à pile ou face comporte traditionnellement le choix au hasard par l'un des participants du coté pile ou face d'une pièce de monnaie et le lancement de la pièce par l'autre participant. Pour remplacer le lancement de la pièce de monnaie Alice et Bernard décident de tirer un entier au hasard (noté n non nul). Selon que l'entier n est pair ou impair, on considère que la pièce est retombée coté pile ou coté face. Pour remplacer le choix au hasard entre pair et impair, Alice et Bernard décident d'utiliser le tirage aléatoire d'une variable entière binaire (notée p).

Alice et Bernard se méfient mutuellement l'un de l'autre et souhaitent déterminer un protocole ayant un bon niveau de sécurité c'est-à-dire que l'un comme l'autre ne doit pas pouvoir tricher et doit pouvoir vérifier que l'autre ne triche pas. Le protocole à définir n'utilisera pas de tiers de confiance qui assurerait pendant le déroulement un rôle de coordination ou d'arbitrage. Le protocole ne peut pas non plus disposer d'un état global commun qui serait réalisé par l'accès à une mémoire commune partagée.

Une version de base du protocole pourrait être la suivante. Dans cette version, dans une première phase, Alice est la participante qui tire l'entier au hasard et Bernard le participant qui choisit pair ou impair. Dans la seconde phase, Alice et Bernard échangent en clair leur valeur. Dans la troisième phase les deux partenaires décident qui est gagnant.

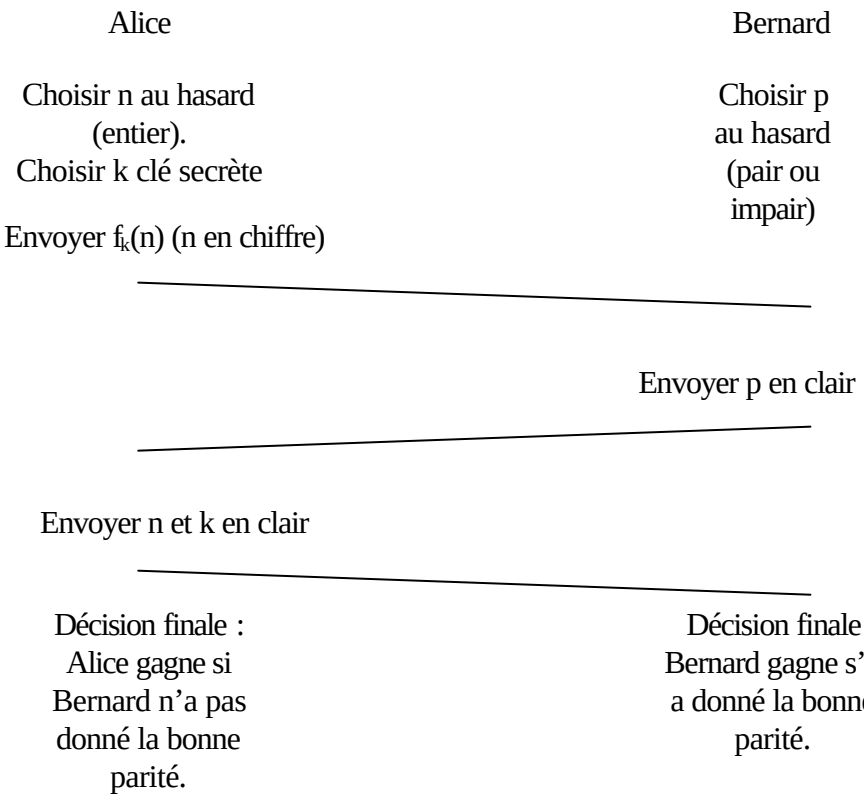


1) Etudiez le protocole précédent en sécurité. Quelles sont les fraudes possibles de la part de Alice ou Bernard ?

2) On cherche une solution au jeu de pile ou face en réseau sans utiliser de fonctions cryptographiques. On doit donc transmettre des valeurs aléatoires en clair par messages

asynchrones. Peut-on construire un protocole sécuritaire (par exemple rendre sécuritaire le protocole précédent)? Pourquoi est-ce difficile (cherchez à construire une solution et montrez ses difficultés) ?

Pour garantir la sécurité de la décision finale, Alice et Bernard étudient la possibilité d'utiliser des fonctions cryptographiques. On passe à une solution de base avec trois messages. Au lieu d'envoyer n directement dans son premier message, Alice va envoyer f(n) ou f est une fonction cryptographique appliquée à n. Une idée assez naturelle consiste à utiliser pour f une fonction de chiffrement symétrique à clé secrète k (f est par exemple le DES ou l'AES). Alice transmet donc dans son premier message $f_k(n)$ ou k est une clé secrète choisie par Alice. Dans cette première solution cryptographique, Bernard transmet ensuite p en clair. Dans le troisième message Alice transmet à Bernard n et k en clair.

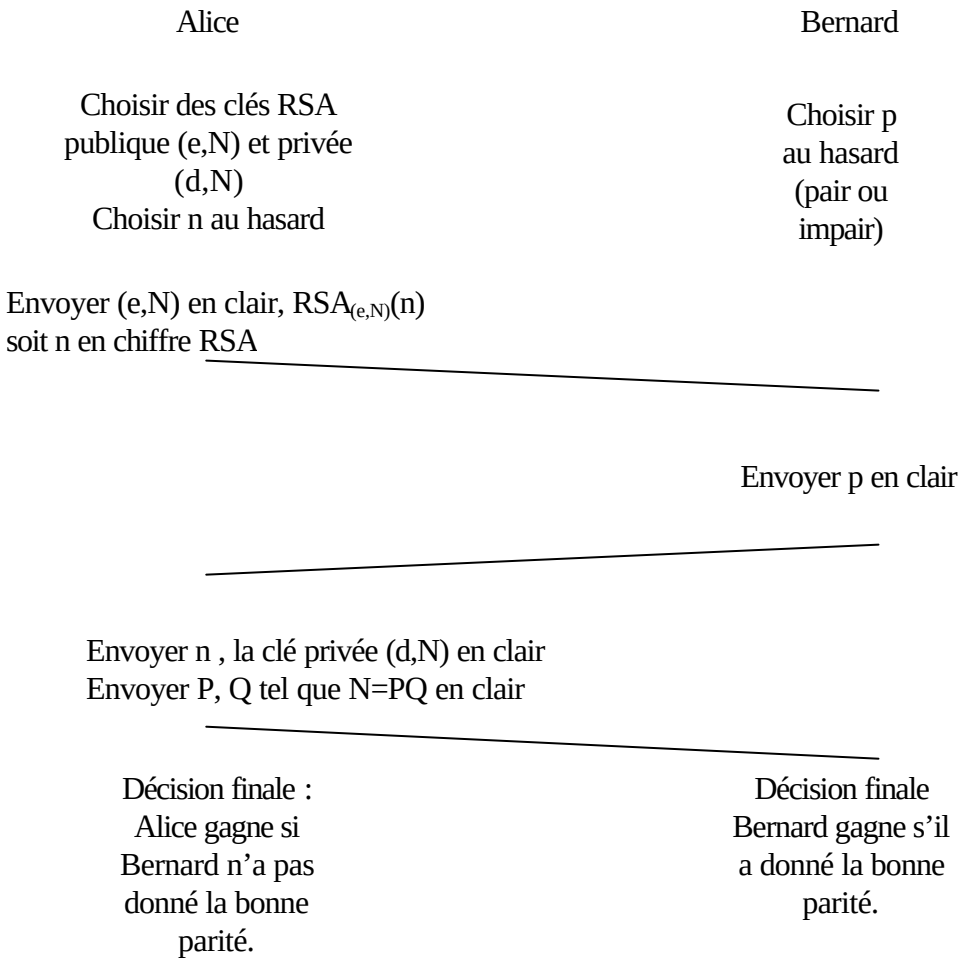


3) Le protocole obtenu est il sécuritaire (selon vous est ce que Alice ou Bernard en utilisant ce protocole avec un chiffre à clé secrète peuvent tricher) ?

4) Dans les mêmes conditions qu'à la question précédente, utilise maintenant une fonction f de chiffrement asymétrique (typiquement le chiffre RSA). Alice détermine au début du protocole un couple clé publique/clé privée. Pour envoyer n chiffré au début sous la forme $f_k(n)$, Alice doit-elle chiffrer avec la clé privée ou avec la clé publique qu'elle vient de déterminer. Alice termine en envoyant comme précédemment les moyens de déchiffrement de n (k,n). Le protocole obtenu est il sécuritaire (selon vous est ce qu'en utilisant ce protocole avec un chiffre à clé publique Alice ou Bernard peuvent tricher)? (2 points)

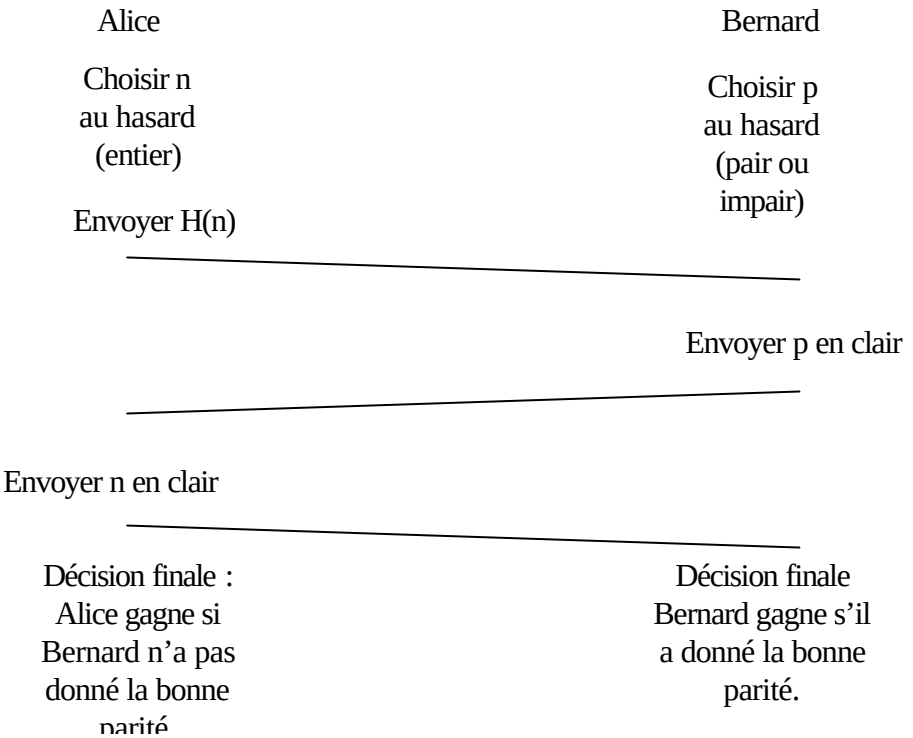
Dans une approche à clés publiques avec le chiffre RSA, Alice doit donc générer pour elle-même un couple clé publique (notée (e,N)), clé privée (notée (d,N)) avec N=PQ produit de deux

nombres premiers. Cependant, dans l'optique d'une approche à clé publique il est assez naturel que Alice après avoir choisi son couple clé publique, clé privée dévoile à Bernard sa clé publique. On va donc examiner maintenant une nouvelle solution dans laquelle Alice commence le protocole en générant un couple clé publique/clé privée. Elle choisit l'entier n aléatoire. Dans son premier message elle communique à Bernard la valeur de la clé publique (e, N) en clair et le chiffre RSA de n au moyen de sa clé publique. A la fin du protocole Alice dévoile à Bernard sa clé privée Pour simplifier la vérification Alice dévoile aussi les deux entiers premiers utilisés P et Q. Bernard a le même comportement que précédemment en choisissant p et en le transmettant en clair.



5) Le protocole obtenu est il sécuritaire (selon vous est ce que Alice ou Bernard en utilisant ce protocole avec le chiffre à clé publique RSA peuvent tricher)?

Après avoir cherché à utiliser un chiffre on cherche maintenant à utiliser une fonction de hachage cryptographique H (comme le MD5 ou le SHA-1). Une solution de base avec fonction de hachage est la suivante :



6) Rappelez la définition d'une fonction de hachage à sens unique. Que se passe t'il (du point de vue de la sécurité), si la fonction H n'est pas à sens unique ?

7) Rappelez la définition d'une fonction de hachage à collisions faibles difficiles. Que se passe t'il (pour la sécurité) si la fonction H n'est pas à collisions faibles difficiles ?

8) Rappelez la définition d'une fonction de hachage à collisions fortes difficiles. Que se passe t'il (pour la sécurité) si la fonction H n'est pas à collisions fortes difficiles ?

9) Le protocole sous l'hypothèse que H est à sens unique et à collisions fortes difficiles est il sécuritaire (Alice ou Bernard peuvent ils tricher) ?

10) Alice et Bernard souhaitent définir un protocole, qui puisse être en cas de litige, soumis à un juge. Pour cela, Alice et Bernard se proposent de modifier le protocole qui vient d'être décrit (avec fonction de hachage) ou d'ajouter l'utilisation d'autres fonctions cryptographiques dans les messages. Proposez des modifications pour une version du protocole qui puisse être soumise à l'arbitrage en non répudiation d'un juge (justifiez les mécanismes introduits permettant de défendre le point de vue que seuls Alice ou Bernard peuvent être l'auteur de leurs messages et que les messages émis ont effectivement été reçus) ?

11) La méthode de Diffie-Hellman est réputée permettre un échange de clé secrète aléatoire entre deux entités qui ne peuvent communiquer que par un réseau non sécurisé. L'objectif est donc que la clé ne circule jamais en clair sur le réseau et qu'elle soit aléatoire. On peut donc essayer de résoudre le problème posé par la sécurisation du jeu de pile ou face en réseau, en utilisant la solution de Diffie-Hellmann pour partager les deux variables aléatoires nécessaires au jeu de pile ou face. Peut-on construire une solution au jeu de pile ou face empêchant Alice ou Bernard de tricher et qui serait basé sur l'utilisation du protocole de Diffie et Hellmann? (2 points)

Exercice 8 : Partage d'un secret

Un conseil d'administration prend ses décisions au moyen de votes électroniques. Il comporte un président dont la voix compte double et trois membres dont la voix est simple (notés 1, 2, 3). Une décision n'est prise que si un vote rassemble au moins 3 voix. Les votes sont acquis lorsque le titulaire présente selon son rang soit une soit deux parts d'un secret.

- Les services de sécurité ont fait implanter la méthode de Shamir avec une arithmétique modulo 31.
- Le secret partagé est tiré aléatoirement à la valeur 7.
- Les coefficients aléatoires du polynôme retenu sont 1, 9, et le terme constant est 7.
- On déduit les parts de secret du président des valeurs aux points x=1; x=2.
- On déduit des parts du secret des membres 1, 2, 3 des valeurs du polynôme pour 3, 4 , 5.

1) Quelles sont les valeurs des parts?

2) On suppose que le président et le membre 1 sont d'accords. Montrez qu'ils peuvent emporter le vote.

CHAPITRE 2: PROTOCOLES DE SECURITE

Exercice 9 : Problème de notarisation

On souhaite réaliser un service ayant pour objectif la non répudiation de transactions entre deux sites A et B qui fonctionnent en mode A client et B serveur (service de notarisation). Le notaire est noté N.

On note b la clef de chiffrement (clef publique dans les crypto systèmes asymétriques) de Bob et B sa clef de déchiffrement (clef privée dans les crypto systèmes asymétriques).

Les étapes du protocole sont les suivantes :

1. Alice chiffre et signe M avec sa clef a et l'envoie au notaire.
2. Le notaire déchiffre le message, vérifie la signature qui garantit que le demandeur du service est bien Alice. Il attribue à l'envoi du message un numéro de séquence envoi(M), date cet envoi te(M), enregistre envoi(M), te(M), M dans un fichier intègre. Il constitue un message contenant envoi(M) signé avec la clef d'Alice et (envoi(M), te(M), M) chiffré avec sa clef.
3. Alice peut alors contrôler la signature qui authentifie le notaire (qui est le seul avec Alice à connaître la clef d'Alice) et constitue une référence de la preuve d'envoi. Par contre Alice ne peut pas déchiffrer la seconde partie du message. Elle l'envoie à Bob.
4. Bob ne peut pas, lui non plus, lire le message. Il le renvoie au notaire
5. Le notaire déchiffre le message avec sa clef. Il vérifie que le numéro d'envoi est dans son fichier. Il attribue à la réception du message un numéro de séquence réception(M), date cette réception tr(M), enregistre réception(M), tr(M), M dans le fichier intègre. Il constitue un message contenant (envoi(M), te(M), réception(M),tr(m), M) signé et chiffré avec la clef de Bob. Il transmet ce message à Bob. Il envoi un message contenant réception(M) à Alice.
6. Bob peut déchiffrer le message, contrôler la signature qui authentifie le notaire. Il envoie un accusé de réception signé avec sa clef au notaire.
7. Le notaire enregistre la fin de la transaction.

1) Rappelez la définition du problème de non répudiation.

2) Décrivez par des diagrammes d'échange de messages chaque étape en utilisant uniquement des algorithmes de chiffrement symétriques; pour chaque étape, exprimez les propriétés obtenues: confidentialité, intégrité, non répudiation, vis a vis de tous les participants (A, B, N, les autres entités présentes dans le réseau) qui sont réalisées par chacune des six étapes.

3) Proposez pour chacune des étapes une solution à clé publique qui atteigne exactement les propriétés précédentes vis a vis des mêmes intervenants. Expliquez de manière très précise pourquoi votre solution assure les propriétés recherchées.

4) On suppose que l'usager client A dispose d'un délai dmax pour signaler la perte de sa clé. Toute perte non signalée dans le délai dmax ne peut-être tenue à charge contre l'usager. Toute perte non signalée après le délai dmax entraîne la responsabilité totale du client en cas de contestation.

Que pourrait faire le notaire pour que le serveur B et le notaire ne soient jamais piégés par un client malveillant qui déclare la perte de sa clé après avoir donné un ordre et avant dmax. Dans quelles applications une telle stratégie est elle utilisable? Dans quelles applications est-elle inutilisable?

5) La solution utilise l'algorithme RSA. Si un utilisateur pour répudier sa signature électronique conteste la sécurité du protocole quels arguments peuvent être opposés?

6) Le site B peut essayer de dénier avoir reçu l'ordre (s'il ne lui convient pas) en n'émettant pas l'acquittement final et en prétendant ensuite qu'il n'a jamais reçu la transaction interprétable ou que son calculateur est tombé en panne à ce moment. Comment le notaire peut-il contrer cette attitude (recherchez une solution déduite des pratiques bancaires courantes)?

Exercice 10 : Changement périodique de clés en cryptographie à clés publiques.

On étudie un protocole de transmission d'informations développé pour sécuriser une voie point à point (logiciel de communication Nicecom). Dans l'une des versions on utilise un algorithme de cryptographie à clés publiques comme le RSA.

Chaque utilisateur dispose donc au départ d'un couple clé publique, clé privée pour un algorithme à clés publiques RSA:

Pour un utilisateur A: clé secrète DA et clé publique EA

Pour un utilisateur B: clé secrète DB et clé publique EB

Pour rendre excessivement difficiles les violations de sécurité on décide dans ce produit de pratiquer un changement périodique des clés de façon automatisée (par échange de messages en cours de dialogue).

En fait les clés vont donc être modifiées périodiquement de sorte que l'on utilisera pour un site comme A une suite de couples de clés: (EA(0) = EA, DA(0) = DA), puis (EA(1), DA(1)) , ..., puis (EA(i), DA(i)) , (EA(i+1), DA(i+1)) ,....

A l'instant initial A utilise (EA(0), DA(0)) et B connaît EA(0) et symétriquement. B utilise (EB(0), DB(0)) et A connaît EB(0).

On veut atteindre les objectifs suivants:

O1 : Authentification des correspondants.

Dans les exemples traités A est l'appelant et B l'appelé.

O2 : Confidentialité des informations qui circulent sur le réseau.

O3 : Authentification de l'origine de tous les messages échangés.

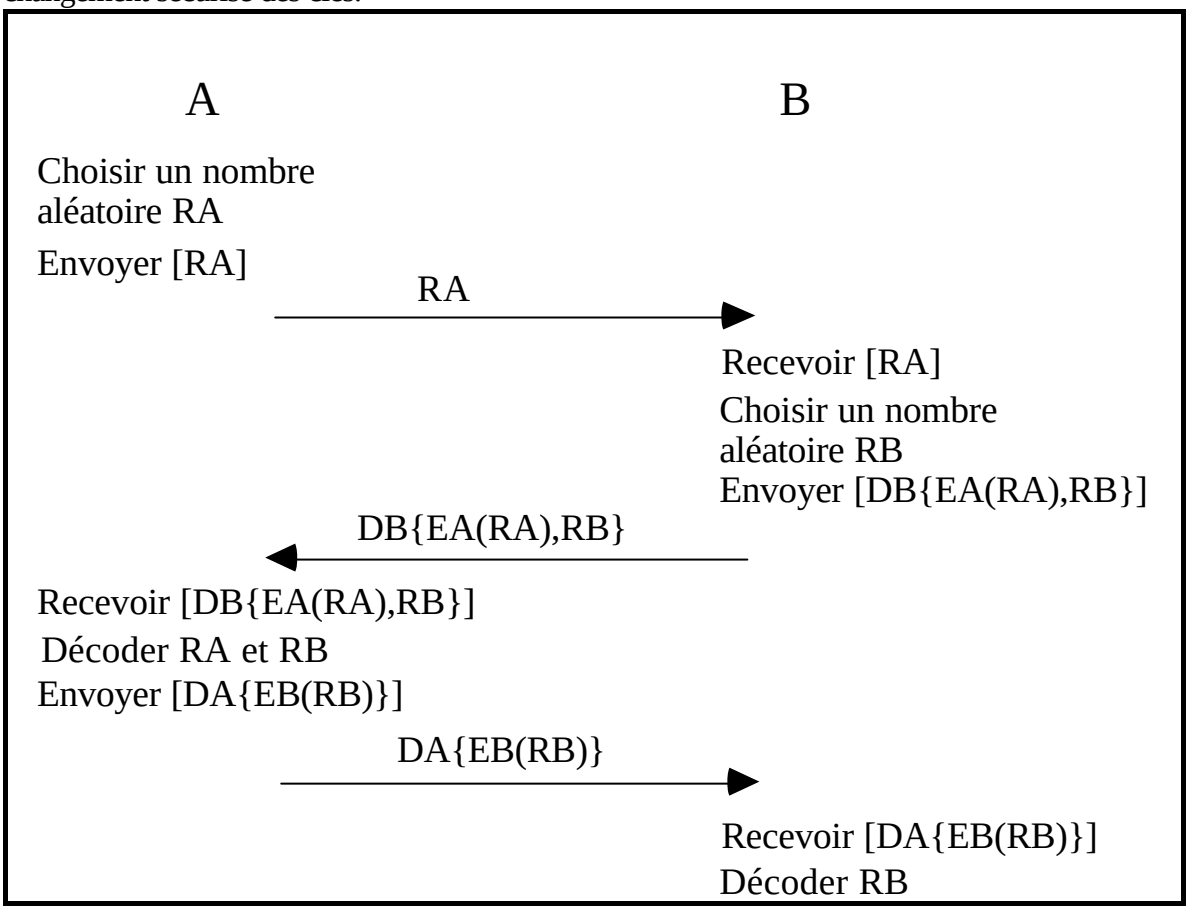
O4 : Limitation de la durée d'usage des clés par changement périodique.

Le logiciel considéré utilise un protocole Px suivant entre les deux usagers A et B. Dans le schéma suivant on ne décrit pas complètement le rôle de tous les messages et l'utilisation des variables échangées (objet de questions du problème).

1) Quel type de problème permet de résoudre le protocole Px (justifiez votre réponse en discutant du fonctionnement du protocole Px en particulier du rôle des nombres RA et RB)? Que pensez vous de l'efficacité de ce protocole pour résoudre ce problème en termes de sécurité et de performances?

2) On prend le protocole Px comme base pour obtenir une version modifiée Py permettant le changement sécurisé des clés c'est à dire le passage du couple (EA(i), DA(i)) à (EA(i+1), DA(i+1)) et également du couple (EB(i), DB(i)) à (EB(i+1), DB(i+1)). On suppose que les sites disposent d'algorithmes leur permettant de déterminer quand ils le souhaitent des couples clé publique, clé privée comme (EA(i+1), DA(i+1)) pour l'algorithme RSA.

2.1) Proposez des modifications du protocole Px pour définir le protocole Py qui assurent le changement sécurisé des clés.



2.2) Quelles clés échange-t-on, à quel moment, selon quel cryptage? Justifiez votre solution. On rappelle que la sécurité de l'algorithme RSA repose sur le fait que les clés privées doivent rester confidentielles.

3) En quoi la technique de modification des clés appliquée rend t-elle extrêmement difficile l'action des pirates.

3.1) On examinera d'une part le problème de décryptage par un pirate qui essaierait de casser le code par analyse des messages échangés.
3.2) On examinera ensuite le cas d'un pirate qui connaîtrait les clés initiales d'un utilisateur A ou B par des moyens autres que le décryptage (négligence de l'utilisateur). Que se passe-t-il?

3.3) L'accès au calculateur de l'émetteur ou du destinataire en session à distance constitue une possibilité d'attaque d'un tel protocole. Que doit faire un pirate qui a réussi à se connecter?

3.4) Quels sont les moyens matériels et logiciels qui protègent les systèmes de ce type d'attaque?

4) 4.1) Quand on est en phase i (avec les couples des clés (EA(i), DA(i)) (EB(i), DB(i))) comment sont assurées la confidentialité et l'authentification des messages en mode RSA ?

4.2) Quel est l'inconvénient majeur de cette approche ?

4.3) Comment pourrait-on résoudre ce dernier problème ?

Exercice 11 : Protocoles d'échanges de clés sécurisés

1) Le problème de l'échange de clés secrètes en utilisant un réseau (le partage d'un secret), est l'un des problèmes les plus importants de la sécurité. Pourquoi ?

2) De manière générale, citez les propriétés de sécurité qui doivent être satisfaites par un protocole qui transporte une clé secrète ?

3) Une solution d'échange de clés entre Alice et Bernard (A et B) en cryptographie à clé secrète est souvent réalisée en supposant l'existence d'un tiers de confiance qui est un gardien de clés. On fait l'hypothèse que chaque utilisateur A (Alice) et B (Bernard) possède une clé secrète. On fait l'hypothèse que le gardien connaît ces clés secrètes KA et KB. Proposez un protocole sécurisé d'échange d'une clé entre Alice et Bernard en utilisant les services du gardien de clés. Alice demande au début du protocole, une clé de session au gardien, pour dialoguer avec Bernard.

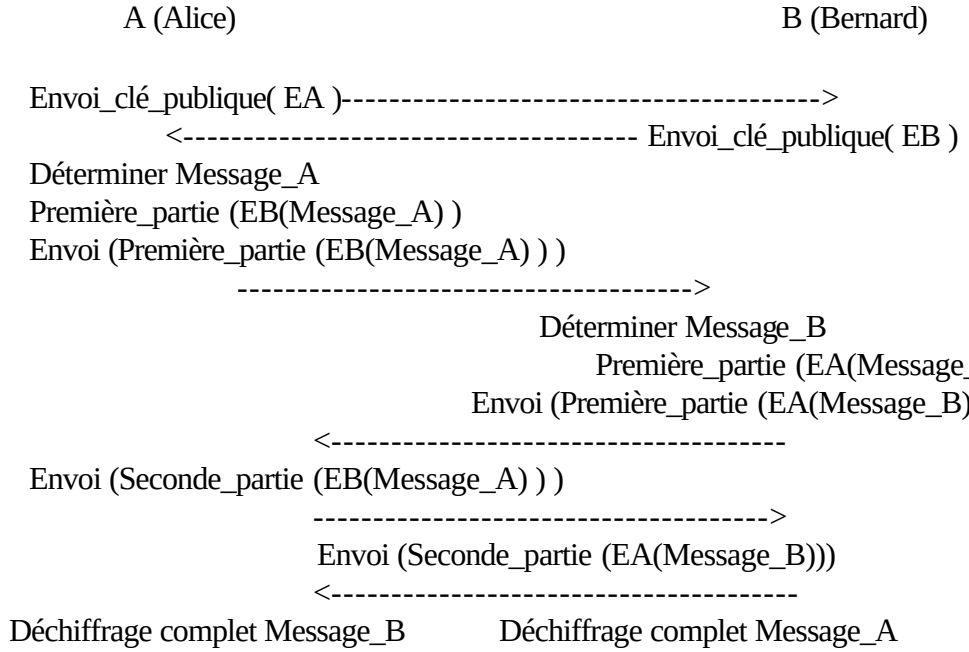
(Alice) Gardien B (Bernard)

Demande_clé (A, B)

---------->

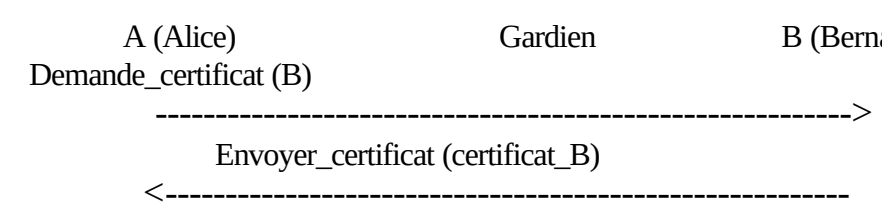
4) La cryptographie à clés publique est réputée permettre à des interlocuteurs ne se connaissant pas de communiquer de façon confidentielle. On cherche une solution à clé publique entre deux entités A et B (Alice et Bernard). La solution recherchée est à discipline intrinsèque c'est à dire que le protocole garantit par lui même la sécurité poursuivie sans utiliser de gardien de clé, de juge, d'arbitre, de notaire etc. pour régler d'éventuels problèmes. Proposez un protocole sécurisé d'échange de clé à discipline intrinsèque utilisant la cryptographie à clé publique. Analysez votre protocole en termes de répétition et d'insertion dans le réseau entre Bernard et Alice d'un pirate Martin.

5) Une idée due à Rivest et Shamir, deux des inventeurs du RSA a été baptisée protocole à 'cliquets' car elle fonctionne par étapes qui ne permettent pas de revenir en arrière. Il s'agit d'une solution à clé publique et à discipline intrinsèque dont l'objectif est de parer une attaque de l'intercepteur Martin. Le protocole suivant est plus général que l'échange d'une clé : il propose l'échange d'un message noté Message_A entre Alice et Bernard et d'un Message_B de Bernard vers Alice :



Le découpage des messages cryptés en deux parties (par exemple par moitié) doit simplement être tel qu'une partie d'un message chiffré n'est pas déchiffrable directement sans la connaissance de l'autre partie. On enverrait par exemple dans un codage par blocs, la première moitié de chacun des blocs puis la seconde moitié de chacun des blocs. Pourquoi un tel protocole gène t' il considérablement une attaque d'un intercepteur ?

6) Avec la méthode d'échange d'un secret de Diffie et Hellman (1976) deux sites peuvent partager une valeur secrète. Chaque site émet une valeur caractéristique du secret (qui est donc publique) et chaque site utilisant cette valeur peut calculer un secret. Pourquoi ce secret ne peut être deviné par un écouteur passif ? Que peut faire un écouteur actif pour contrer ce protocole ?
7) Pour donner un bon niveau de sécurité, la plupart des solutions implantées utilisent un annuaire (un tiers de confiance) qui distribue des clés publiques de manière sécurisée. On appelle certificat une donnée liant un nom logique (un nom d'usager) et une clef publique associée à ce nom. Un protocole d'échange de clés de session commence alors par un accès sécurisé à une clé publique comme suit :



Comment le protocole suivant est il exécuté pour empêcher une attaque de l'intercepteur Martin ? On peut ensuite par exemple procéder à un échange de clé de session en toute confiance.

8) DAP ("Directory Access Protocol") est le protocole client serveur de gestion répartie de l'annuaire X500 associé à la messagerie X400. LDAP ("Lightweight Directory Access Protocol") est une version simplifiée de DAP adaptée à l'environnement Internet pour en faire un protocole de niveau application de l'Internet. Il permet à un client LDAP d'accéder, de modifier, d'ajouter des données à un annuaire géré par un serveur LDAP en utilisant les protocoles TCP/IP. De très nombreux types de données peuvent être stockés dans un annuaire LDAP mais la donnée

essentiellement gérée par les annuaires LDAP sont des certificats au format X509. Les certificats X509 ont été définis au départ dans la norme X509 de la suite des protocoles X500. La spécification suivante définit l'essentiel de la structure de donnée associée à un certificat au format X509 (langage de syntaxe de données ASN1). Commentez la structure de donnée ainsi retenue. Comment est vérifiée une signature ?

```
Certificate ::= SIGNED {
        SEQUENCE{
                        version [0]                     Version DEFAULT v1,
                        serialNumber                    CertificateSerialNumber,
                        signature                       AlgorithmIdentifier,
                        issuer                          Name,
                        validity                        Validity,
                        subject                         Name,
                        subjectPublicKeyInfo    SubjectPublicKeyInfo,
                        issuerUniqueIdentifier
                                [1] IMPLICIT UniqueIdentifier OPTIONAL
                         }
                                }
Version ::= INTEGER { v1(0), v2(1), v3(2) }
CertificateSerialNumber ::= INTEGER
AlgorithmIdentifier ::= SEQUENCE {
        algorithm               ALGORITHM.&id({SupportedAlgorithms}),
        parameters              ALGORITHM.&Type ({SupportedAlgorithms}
                                        {@algorithm}) OPTIONAL }
SupportedAlgorithms ALGORITHM ::= { ... }
Validity ::= SEQUENCE { notBefore UTCTime, notAfter UTCTime }
SubjectPublicKeyInfo ::= SEQUENCE { algorithm   AlgorithmIdentifier,
                                    subjectPublicKey    BIT STRING      }
Time ::= CHOICE {       utcTime                 UTCTime,
                                generalizedTime         GeneralizedTime }

SIGNED { ToBeSigned } ::= SEQUENCE {
                        toBeSigned ToBeSigned,
                        encrypted ENCRYPTED { HASHED {ToBeSigned }}}
```

Exercice 12 : Construction d'un système de paiement par chèques à support électronique

On souhaite construire un système de paiement par chèques tel que les chèques puissent être acheminés dans un réseau non sécurisé (par exemple sur l'Internet dans des courriers électroniques).

On appelle A (pour acheteur) la personne qui émet le chèque, C (pour commerçant) la personne qui reçoit le paiement. BA est la banque de A et BC la banque de C.

On construit la solution en cryptographie à clés publiques (chiffrement asymétrique utilisant typiquement l'algorithme RSA) de sorte que les partenaires possède tous un couple (E , D) ou E est la méthode de chiffrement à clé publique et D est la méthode de déchiffrement. On note respectivement (Ea , Da) , (Eba , Dba) , (Ec , Dc) , (Ebc , Dbc) les couples de clés.

Les clés publiques sont accessibles sous forme de structures de données certificats dans un annuaire de certificats. Enfin pour faire de la signature on utilise une fonction de hachage H ayant de bonnes propriétés de sécurité. La fonction de hachage sécuritaire notée H est par exemple le MD5 ou le SHA.

I) Protocole chèque standard

A peu près comme dans la vie courante, pour régler un achat, l'acheteur A prépare une formule de chèque qui comporte les coordonnées de A (en fait l'identifiant A qui peut comporter différentes informations dont le numéro de compte etc.), les coordonnées de C (l'identifiant du destinataire C), un identifiant unique de chèque Id_Cheq_A (un numéro de chèque de A qui doit faire qu'une formule de chèque a un identifiant unique) et le montant du chèque (baptisé tout simplement Montant). Toutes ces informations sont signées par A. On a donc tout d'abord :

A, C , Id_Cheq_A , Montant , Da (H (A , C , Id_Cheq_A , Montant))
A ——————————————————————————————— C

Pour mettre le chèque en paiement, C signe le chèque en utilisant sa clé privée et transmet le chèque 'endossé' à sa banque BC pour que le montant soit viré à son compte.

C, BC , Id_Cheq_A , Montant , Dc (Da (H (A , C , Id_Cheq_A , Montant)))
C ——————————————————————————————— BC

I.1) Avec ce protocole un intrus peut-il faire payer ses dépenses auprès de C en se faisant passer pour A . Comment C vérifie que seul A, a pu émettre le chèque ?

I.2) Avec ce protocole pourquoi la banque BC est elle sure que seul A a pu émettre le chèque, qu'il est destiné à C et que C a accepté le paiement ? Voyez vous une différence entre la signature Dc(Da(H(A,C,Id_Cheq_A ,Montant))) et l'utilisation de deux signatures concaténées Da(H(A,C,Id_Cheq_A,Montant)), Dc(H(A,C,Id_Cheq_A,Montant))

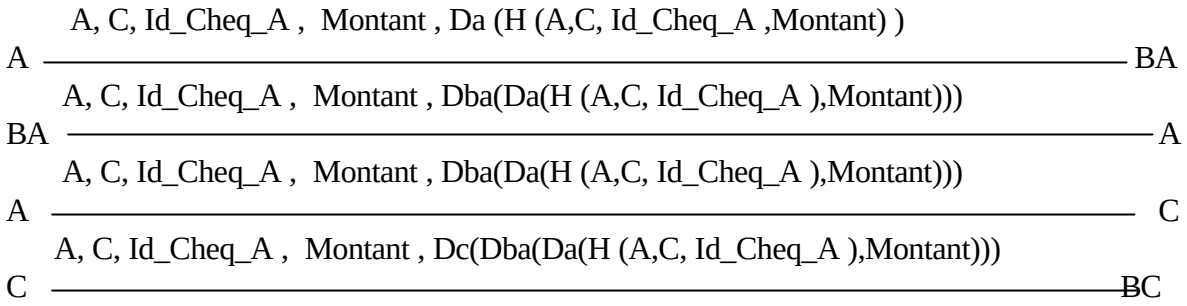
I.3) Avec ce protocole, C peut-il présenter plusieurs fois le chèque en paiement à sa banque ? A peut-il utiliser plusieurs fois le même chèque pour payer C (le même commerçant) ou des commerçants différents ?

I.4) Est ce que les paiements effectués par A sont confidentiels ?

I.5) Est ce que A peut faire des chèques sans provision ?

II) Protocole chèque de banque

On utilise maintenant le protocole suivant.



II.1) Ce protocole s'apparente à la notion de chèque de banque. Pourquoi ? (Avec ce protocole comment s'assurer simplement que A ne peut faire des chèques sans provision)

II.2) Avec ce protocole comment assurer que A ne peut utiliser plusieurs fois le même chèque pour payer le même commerçant ou des commerçants différents ?

II.3) Comment construire une solution pour que les paiements effectués par A soient confidentiels ?

Exercice 13 : Protocole de micro-paiement sur Internet

Dans ce problème Bob souhaite faire réaliser n opérations successives par Alice. Ces opérations ont la particularité d'être toutes les mêmes et d'avoir les mêmes paramètres. En fait nous allons utiliser dans la suite du problème ces opérations pour des paiements de sommes très faibles sur Internet (par exemple une somme fixe d'un centime d'euro à chaque opération).
Alice souhaite authentifier l'émetteur d'une opération (ici c'est Bob) à chaque opération.

1) Quels sont les avantages et les inconvénients d'une authentification unique en début d'une session (comme dans le protocole telnet) et d'une authentification pour chaque opération (authentification en continu) ?

Dans ce problème nous supposons que pour l'authentification en continu, Bob et Alice ont préalablement échangé une clef secrète C comportant k bits. H est une fonction de hachage de sécurité qui a été négociée entre Alice et Bob, $\oplus$ est le ou exclusif, s est un nombre aléatoire sur k bits (nombre aléatoire de sécurité qui est différent à chaque authentification). On note $\{C \oplus s\}^H$ le résultat de l'application de la fonction de hachage H au ou exclusif de C et de s. On note un message : émetteur, destinataire, type du message, données.

De manière à décomposer la solution on l'étudie progressivement. Dans une première étape on considère le cas d'une seule opération (donc n=1). Le protocole appliqué est le suivant:

	Bob	Réseau	Alice
1	Bob génère un nombre aléatoire s sur k bits; il calcule $h_1 = \{C \oplus s\}^H$		
2		Bob, Alice, 'Transmission du nombre s sur k bits', s ———	
3	Bob demande à Alice la réalisation de l'opération	Bob, Alice,'Demande opération', h_1 ———	
4			Alice vérifie que $h_1 = \{C \oplus s\}^H$ Si oui Alice réalise l'opération

2) Quelles sont les propriétés des fonctions de hachage de sécurité ?

3) Pourquoi Estelle ne peut, en écoutant les messages qui circulent entre Alice et Bob, accéder à la clé secrète C partagée par Alice et Bob?

4) Comment Alice peut-elle authentifier Bob ?
5) Comment s'appelle la technique de chiffrement utilisée dans ce protocole?

On considère maintenant le protocole suivant défini pour n opérations successives :

	Bob	Réseau	Alice
1	Bob génère s aléatoire sur k bits et calcule : $h_0 = C \oplus s$ $h_1 = \{h_0\}^H$ ..... $h_i = \{h_{i-1}\}^H$ ….. $h_n = \{h_{n-1}\}^H$ Bob conserve la suite h_1 , …. h_i, … , h_n		
2		Bob, Alice, 'demande préliminaire d'ouverture de n opérations', s	
3			Alice calcule la suite $h_0 = C \oplus s$ $h_1 = \{h_0\}^H$ ..... $h_i = \{h_{i-1}\}^H$ ….. $h_{n+1} = \{h_n\}^H$ Alice détruit les h_i et ne conserve que h_{n+1}
4	Demande de la première opération	Bob, Alice, 'demande d'opération', h_n	
5		———————————	Authentification de Bob ? Réalisation de l'opération 1 si OK
…	…….	……..	……
2+2i	Demande de la ième opération ($i \leq n$)	Bob, Alice, 'demande d'opération', h_{n-i+1} ———————————	
2+2i+1			Authentification de Bob ? Réalisation de l'opération i si OK

6) Bob mémorise toute la suite des h_i qu'il présente un par un à chaque opération en commençant par h_n puis h_{n-1}…. puis h_{n-i+1}….. Alice mémorise une seule valeur en commençant par h_{n+1} qu'elle remplace ensuite par h_n puis h_{n-1} et ainsi de suite à chaque nouvelle opération. Quelle est la vérification réalisée par Alice pour authentifier Bob aux étapes 5 et 2+2i+1 ?

7) Expliquez pourquoi Estelle, qui observe ce protocole ne peut à aucune étape en tirer des informations suffisantes pour pouvoir usurper l'identité de Bob.

Supposons que l'on se place dans une approche de cryptographie à clé publique. Alice et Bob disposent donc de couples clé publique, clé privée et de certificats accessibles dans un annuaire de certificats. Pour réaliser le protocole précédent d'authentification, dans une étape préalable, Alice et Bob doivent s'authentifier en utilisant un protocole à clef publique, Alice doit générer la clé secrète C et doit l'envoyer à Bob en utilisant également la méthode à clés publiques.

8) Donnez une solution pour cette authentification mutuelle et pour l'échange de clé C. Quel(s) protocole(s) de l'Internet pourrait servir à faire ces authentifications.

Supposons que Alice soit une banque et Bob un de ses clients. On utilise maintenant le protocole précédent pour faire des micro paiements. Un protocole de micro paiement est un protocole permettant de débiter des sommes très faibles pour des applications de consultation payantes sur Internet (lire une page d'un document ou d'un journal par exemple). On suppose que n représente une somme en centimes d'euro (n = 1000 centimes par exemple). Cette somme est débitée du compte courant de Bob au début et crédite un compte "de micro paiement" utilisable par Bob.

On suppose que chaque débit de micro paiement vaut un centime, ce centime est débité du compte de micro paiement de Bob lors de la réalisation par Alice d'une opération. Chaque opération du protocole précédent est donc un paiement d'un centime.

Quand le compte de micro paiement est vide, Bob doit alimenter à nouveau son compte de micro paiement et Alice et Bob recommencent toute une autre série d'opérations de micro paiement. Si Bob décide de ne plus payer de cette façon, alors que son compte de micro paiement n'est pas vide, son compte courant est crédité du solde du compte de micro paiement.

9) Expliquez pourquoi Bob ne peut pas faire de la fausse monnaie (chaque centime est identifié, ne peut être rejoué).

10) Expliquez pourquoi Estelle ne peut se substituer à Bob pour utiliser le compte de micro paiement de Bob.

11) Quel est l'avantage de ce protocole par rapport à un paiement qui utiliserait à chaque étape une authentification selon les méthodes habituelles (par exemple utilisation de SSL pour les paiements par carte bancaire sur Internet).

Exercice 14 : Propriétés des certificats

Un administrateur observe des certificats générés par la même autorité de certification dans le cadre d'une infrastructure à clé publique (une PKI). Il examine des certificats différents. Pour fixer les idées on suppose par exemple que les certificats sont générés selon la norme X509.

1) En matière de sécurité informatique, rappelez la définition d'un certificat ?

2) L'administrateur constate que deux certificats différents portent une signature identique. Quelle peut être la source de cette situation ? Quel est le danger résultant (que doit faire l'administrateur) ?

3) L'administrateur examine deux certificats différents et constate qu'ils portent des clés publiques identiques. Quelle peut être la source de cette situation ? Quel est le danger résultant ?

4) L'administrateur constate que deux certificats différents portent un numéro de série identique. Quelle peut être la source de cette situation ? Quel est le danger résultant (que doit faire l'administrateur) ?

5) L'administrateur constate que deux certificats différents sont signés au moyen de la même clé privée. Quelle peut être la source de cette situation ? Quel est le danger résultant (que doit faire l'administrateur) ?

CHAPITRE 3 : NORMES ET IMPLANTATIONS INDUSTRIELLES

Exercice 15 : Kerberos

Le projet Athena du MIT (Massachusetts Institutes of Technology) était, dans le milieu des années 80, un projet de recherche portant sur les systèmes répartis. Il avait pour objectif la définition d'un environnement homogène d'accès pour des PC sous MS/DOS, des stations de travail ou des calculateurs sous UNIX en grand nombre et reliés par plusieurs réseaux locaux. L'un des points forts de cet environnement qui reste, est le protocole qui gère à la fois l'authentification des utilisateurs et certains mécanismes de protection. Ce protocole est dénommé Kerberos (Cerbère est le chien à trois têtes, gardien de l'enfer).

Depuis ces origines Kerberos a connu de nombreuses variantes et a été implantés dans différents produits (en particulier chez Digital et plus récemment chez Microsoft). Il constitue le standard des systèmes de distribution des clefs symétriques (gardiens des clefs) et est très utilisé dans l'Internet, en particulier par les organismes de recherche et d'enseignement. La plupart des protocoles de l'Internet (PPP, HTTP, SMTP, FTP…) supportent une authentification des entités communicantes basée sur Kerberos. Ses principes ont été repris dans des architectures de protection comme DCE et Windows. La version à l'IETF est la v5 qui n'utilise que la cryptographie symétrique (DES et triple DES) et les fonctions de hachage sécuritaires (MD5 et SHA-1), la v4 faisant l'objet de la RFC1501.

Description du protocole

Kerberos met en communication trois entités :

- l'entité client : un programme ou un utilisateur sur une machine donnée. Pour simplifier nous considérons que l'entité client figure uniquement un utilisateur humain. Pour nous ce sera Alice.
- l'entité serveur: un programme requis par l'entité client (par exemple le serveur de fichier). Ce sera dans la suite le serveur Bob.
- le centre de distribution des clefs, appelé KDC (pour Key Distribution Center), il connaît les clefs privées de toutes les entités clients ou serveurs qu'il conserve dans une base de données, il génère aussi des clefs de session (temporaires).

Pour fonctionner Kerberos nécessite une synchronisation des horloges de toutes les machines du réseau. Ce protocole doit réaliser entre tout couple d'horloge du réseau un écart inférieur à 5 mn pour des dates lues au même instant (ce qui est assez facile à réaliser).

Pour pouvoir utiliser un serveur, Kerberos réalise un protocole de distribution de clés de session qui respecte les étapes suivantes :

1. Alice s'authentifie auprès du KDC
2. Alice demande un ticket d'utilisation du serveur Bob au KDC. Un ticket au sens de Kerberos comporte essentiellement une clé de session et différentes informations annexes détaillées plus loin. KDC lui renvoie un ticket valable pour le serveur demandé.

3. Quand Alice sollicite le serveur Bob, elle utilise le ticket que lui a remis le KDC qu'il met dans sa requête.

Nous allons détailler les trois étapes précédentes. Pour cela, nous définissons précisément les notations des différents objets manipulés par le protocole :

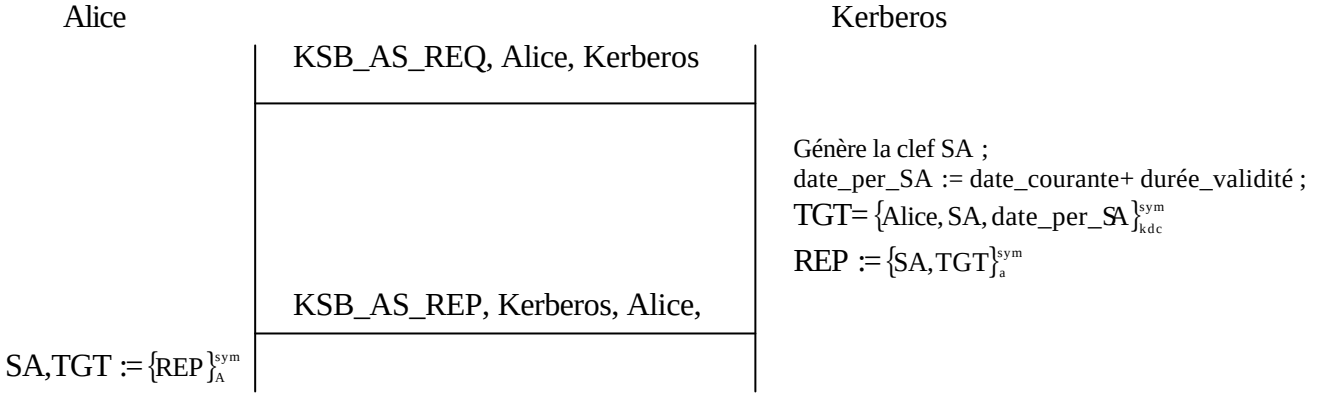
Alice	Le nom du client.
A,a	Une clef utilisée par Alice pour s'authentifier qui en général est dérivée d'un mot de passe A est connue d'Alice et de Kerberos (en majuscule quand elle est utilisée pour le déchiffrement, en minuscule pour le chiffrement)
SA,sa	La clef de session d'Alice SA est connue d'Alice et de Kerberos
Bob	Le nom du serveur invoqué.
B,b	La clef privée du serveur Bob. (en majuscule quand elle est utilisée pour le déchiffrement, en minuscule pour le chiffrement)
Kerberos	Le nom du serveur de distribution des clefs
KDC,kdc	La clef de Kerberos connue de lui seul(en majuscule quand elle est utilisée pour le déchiffrement, en minuscule pour le chiffrement)
KAB,kab	La clef de session déterminée entre client et serveur. (en majuscule quand elle est utilisée pour le déchiffrement, en minuscule pour le chiffrement)
Ticket	Le ticket donné à un client Alice pour utiliser un serveur Bob.
date_fab_X	La date de fabrication de l'objet X
date_courante	Date donnée par l'horloge locale de la machine

Comme dans tout système d'accès a mots de passe, le client a convenu préalablement avec l'administrateur système d'un nom (Alice), et d'un mot de passe qui devient la clef privée A d'Alice. Pareillement, il a été convenu, pour le programme serveur, d'un nom (Bob), et d'une clef privée B. Les clefs ne sont connues que de leur propriétaire (qui doivent donc les garder secrètes), et du serveur d'authentification Kerberos. Ce système doit donc être particulièrement bien protégé.

Etape 1 : Authentification du client

Alice commence par envoyer une requête d'authentification KSB_AS_REQ contenant son Nom de login (Alice) à Kerberos. Kerberos lui renvoie un message KSB_AS_REP contenant une clef de session SA personnelle à Alice et un ticket de contrôle d'accès TGT (Ticket Granting Ticket).

1) Quelles sont les fonctions que doit réaliser un gardien de clefs ? Expliquez les principes du triple DES et ses avantages par rapport au DES ? A quoi peut servir la combinaison d'une fonction de hachage et d'un crypto système symétrique ?



2) Expliquez l'usage des deux chiffrements réalisés dans cette étape. Contre quelle attaque est utilisée la variable date_per_SA ? Expliquez son usage.

Etape 2 : Obtention du ticket d'accès au serveur

Lorsque Alice désire (durant la période de validité de son ticket d'authentification) utiliser le service du serveur Bob, elle adresse à Kerberos une requête d'accès à ce serveur. Cette requête contient outre le nom du serveur, TGT et la date courante chiffrée avec SA. A réception de cette demande Kerberos déchiffre TGT, récupère la clef SA, déchiffre la date avec SA et vérifie que cette date est voisine de son heure courante.

3) Que prouve ce contrôle ? Quelle technique de protection peut utiliser Kerberos pour le contrôle des droits d'Alice? Quel est l'usage du chiffrement réalisé par Kerberos?

Etape 3 : Accès au serveur applicatif

Dans un délai compatible avec la durée de validité du Ticket, Alice va demander au serveur Bob d'exécuter la transaction pour laquelle elle a obtenu le ticket. Elle envoi donc à ce serveur une requête avec le ticket.

4) A quoi sert la variable d1 et pourquoi est elle chiffrée ? Expliquez le principe du contrôle réalisé par Bob ? Quel est l'usage de la variable REP2 ? A quoi pourra servir KAB dans la suite des échanges de la transaction courante entre Alice et Bob ? Comment appelle t'on une telle variable?

Extension de Kerberos

Une extension de Kerberos est en cours de normalisation. Elle consiste à utiliser un crypto système asymétrique pour toutes les opérations utilisant des clefs rémanente (A,B, KDC). Dans ce cas Kerberos n'a plus besoins de stocker les clefs symétriques de chaque participant. Il ne stocke que des clefs symétriques de session et sa clef (KDC) et des certificats pour chaque utilisateur. Ceci réduit considérablement le risque lié à une attaque de Kerberos.

5) Donnez le schéma d'échange de messages de l'étape 1 (cas nominal) correspondant à cette extension en justifiant l'utilisation de chaque opération cryptographique.

[Adr_c ; Adr_s ; ({ Ac }Clé_sess_c,s ; { Tc,s }Clé_s)]

Exercice 16 : Authentification d'accès au réseau Internet : solution des protocoles PAP et CHAP

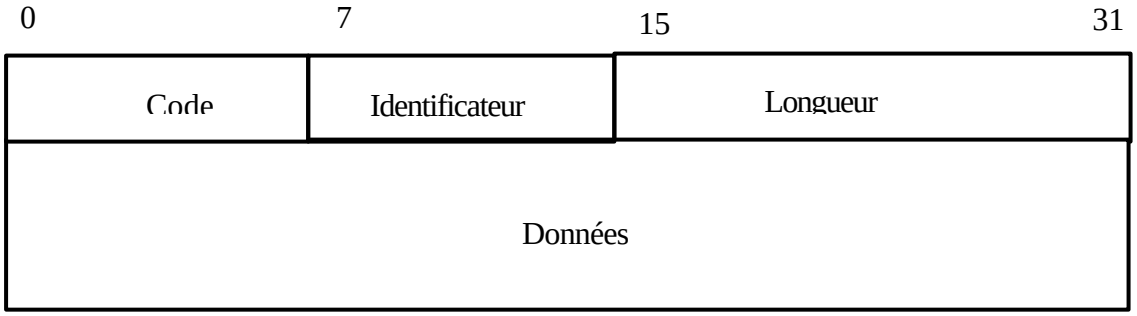
L'accès au réseau Internet est le plus souvent réalisé en utilisant le protocole de liaison PPP ('Point to Point Protocol'). C'est un protocole de communication point à point entre deux sites voisins dans le réseau. Il lui est associé une suite de protocoles dont un protocole de négociation des paramètres de la liaison et deux protocoles d'authentification permettant de réaliser un contrôle d'accès au réseau.

Le protocole PAP ('Password Authentication Protocol', RFC 1334), constitue la forme la plus simple et la plus classique d'authentification utilisée avec PPP. La figure suivante montre la structure des messages échangés :

Code définit le type du message.

Identificateur permet d'associer les requêtes et les réponses.

Longueur définit la longueur de la zone donnée.



PAP est basé sur la connaissance par les deux sites d'un mot de passe secret associé au demandeur. Après négociation des paramètres de liaison et avant l'échange de données, le site demandant l'accès au réseau avec PAP doit fournir son mot de passe au prestataire d'accès. L'échange est le suivant :

Trame Code 1 : Demande d'authentification (par l'usager)

La zone données comporte :

- longueur du nom du demandeur utilisé (1 octet)
- nom du demandeur d'accès ('user's name')
- longueur de mot de passe utilisé (1 octet)
- mot de passe ('password')

Le mot de passe est traité comme dans les systèmes classiques. Il est comparé avec une valeur stockée dans un fichier de mots de passes (éventuellement cryptés) sur le site du prestataire d'accès.

Trame Code 2 : Acquittement positif d'authentification

La zone données est inexistante.

Le demandeur peut ensuite émettre des trames PPP.

Trame Code 3 : Rejet d'authentification

L'utilisateur ne peut pas transmettre de trames. Il doit émettre à nouveau une trame de code 1 et réussir les étapes 1 et 2.

1) Citez deux faiblesses de cette méthode ?

Le protocole CHAP ('Challenge Handshake Authentication Protocol', RFC 1994 rend obsolète PAP RFC 1334), a ensuite été proposé. Il est basé sur le même format de message et sur la même connaissance par les deux sites demandeur et prestataire, d'un nom d'usager associé à un mot de passe secret. Après la phase de négociation des paramètres de la liaison c'est le prestataire d'accès qui prend l'initiative d'authentifier le demandeur de connexion :

Trame Code 1 : Envoi d'un challenge

La zone données comporte :

- longueur du challenge utilisé (1 octet)
- valeur du challenge (suite binaire)

Trame Code 2 : Réponse

Le demandeur de connexion concatène :

- la valeur de l'identificateur
- la valeur du mot de passe secret
- la valeur du challenge

L'ensemble est condensé au moyen d'une fonction de hachage à sens unique. L'algorithme de hachage employé est négociable.

MD5 est le plus souvent utilisé (16 octets de hachage).

Trame Code 3 : Acquittement positif d'authentification.

Le prestataire compare la valeur reçue à la valeur résultat d'un calcul identique effectué localement.

Si les résultats sont identiques l'authentification a réussie.

Trame Code 4 : Rejet d'authentification.

Les valeurs reçues et calculées étant différentes
l'authentification est rejetée.

Ultérieurement, le prestataire périodiquement détermine une valeur de challenge et répète les étapes un à trois.

2) Quelles sont les avantages de cette solution en matière de résistance aux attaques par écoute et répétition de trafic écouté?

3) Pourrait-on avec CHAP authentifier le prestataire comme on authentifie le demandeur ?

4) Peut-on utiliser plusieurs noms et plusieurs mots de passe pour un même utilisateur? Pour quel usage ?

5) Quelles sont les inconvénients de cette solution

6) Quelles sont les qualités indispensables de la solution en matière de mot de passe ?

7) Quelles sont les qualités indispensables de la solution en matière de fonction de hachage à sens unique?

8) Quelles sont les qualités indispensables de la solution en matière de valeur de challenge?

9) Est ce que ce protocole résiste à l'insertion d'un intrus sur la ligne entre le demandeur et le prestataire qui peut être écouteur actif ('active wiretaping') ?

Exercice 17 : Authentification d'accès au réseau Internet : le protocole RADIUS

L'authentification des utilisateurs par des prestataires d'accès au réseau Internet peut être réalisée au moyen des protocoles **PAP** ('**P**assword **A**uthentication **P**rotocol', RFC 1334 octobre 1992) ou **CHAP** ('**C**hallenge **H**andshake **A**uthentication **P**rotocol', RFC 1994, août 1996). Ces protocoles appartiennent à la suite des protocoles de niveau liaison de l'Internet **PPP** ('**P**oint to **P**oint **P**rotocol', RFC 1661, juillet 1994)

On étudie dans ce problème une solution plus récente définie par le protocole **RADIUS** ('**R**emote **A**uthentication **D**ial **I**n **U**ser **S**ervice', RFC 2138, avril 1997). Ce protocole a été défini par la société Linvingston qui l'a donné à la communauté Internet. Il est largement implanté actuellement par les principaux fournisseurs de matériels d'accès Internet et est donc utilisé par de nombreux prestataires d'accès.

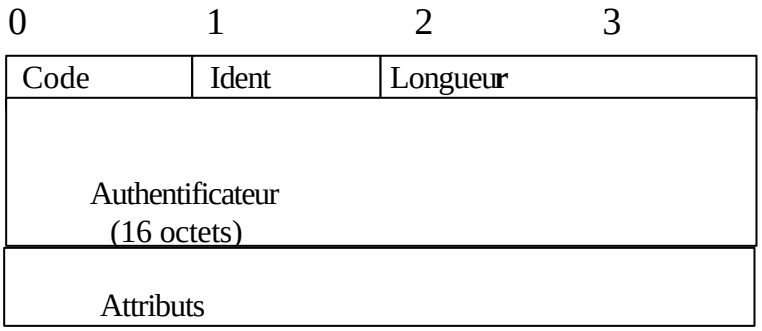
RADIUS définit un nouveau protocole d'authentification tout en permettant d'utiliser les principaux protocoles existants pour les prestataires qui souhaitent conserver leurs méthodes de travail (PPP/PAP, PPP/CHAP, UNIX/login local ou NIS 'Network Information System' de SUN, ou même Kerberos). Il étend les fonctions d'authentification à d'autres fonctions d'administration de réseau, baptisées dans le jargon métier fonctions AAA pour 'Authentication', 'Authorization', 'Accounting' (Vérification d'identité, Contrôle des droits, Comptabilité).

RADIUS est un protocole client serveur de niveau application. Les clients sont soit des sites serveurs d'accès Internet (NAS 'Network Access Server'), soit des routeurs, des pare_feux, des applications d'accès distant comme telnet, rlogin. Le serveur RADIUS est un site de gestion centralisée des informations AAA.

Pour assurer le transport des informations entre client et serveur, RADIUS utilise dans tous les cas le réseau Internet par son protocole de transport UDP.

Notons enfin que le protocole RADIUS est extensible car les échanges de données se font au moyen de la notion d'attribut. Un attribut est défini par son type, sa longueur, sa valeur de sorte que de nouveaux attributs associés à de nouvelles fonctions peuvent être ajoutés sans modifier l'existant.

RADIUS est basé sur un format de message déduit du format **LCP** ('**L**ink **C**ontrol **P**rotocol') de PPP utilisé également par les protocoles PAP et CHAP.



Les différents champs dans le dessin précédent ont la signification suivante :

a) Le code définit en fait sur un octet le type du message.

1 Access-Request
2 Access-Accept
3 Access-Reject

4	Accounting-Request
5	Accounting-Response
11	Access-Challenge
12	Status-Server (experimental)
13	Status-Client (experimental)
255	Reserved

b) L'identificateur ('identifier' sur un octet) associe les requêtes et les réponses. La zone longueur ('length') définit sur deux octets la longueur totale des données incluant code, longueur etc.

c) L'authentificateur ('Authenticator') (16 octets) est utilisé comme son nom l'indique pour l'authentification. Sa méthode de calcul est différente selon les types de messages (voir plus loin l'explication du protocole).

d) Les attributs ('Attributes') sont utilisés pour véhiculer toutes les informations nécessaires aux échanges AAA. Le format d'un attribut est le suivant :

Quelques exemples de types d'attributs.

1 User-Name	2 User-Password	3 CHAP-Password
4 NAS-IP-Address	5 NAS-Port	6 Service-Type
7 Framed-Protocol	8 Framed-IP-Addres	9 Framed-IP-Netmask
10 Framed-Routing	11 Filter-Id	12 Framed-MTU
13 Framed-Compression	14 Login-IP-Host	15 Login-Service
16 Login-TCP-Port	17 (unassigned)	18 Reply-Message
19 Callback-Number	20 Callback-Id	21 (unassigned)
22 Framed-Route	23 Framed-IPX-Network	24 State
25 Class	26 Vendor-Specific	

Version de base du protocole d'authentification

Les principes de sécurité du protocole d'authentification sont les suivants. Comme pour le protocole CHAP les échanges RADIUS sont sécurisés par une clé secrète partagée entre client et serveur (baptisé 'secret' dans la suite). Les usagers sont connus par un nom d'usager et sont authentifiés par un mot de passe secret. Le protocole offre deux possibilités différentes d'authentifier un utilisateur distant. Ce premier échange constitue la version de base.

Client_RADIUS Serveur_RADIUS

Access-Request

Access-Accept ou Access-reject

Structure du message Access-Request

- Le champ authentificateur contient une valeur aléatoire de 16 octets. Dans le message Access-Request cette valeur est baptisée authentificateur de requête ('Request Authenticator').

- Un champ attribut de type nom d'usager contient le nom d'utilisateur en clair.

- Un champ attribut de type mot de passe contient le mot de passe usager codé par :

MD5 (Secret || Authentificateur) **OUEX** Mot_de_passe

|| : l'opérateur de concaténation.
MD5 : fonction de hachage à sens unique pour créer une signature de 16 octets.
OUEX : ou exclusif (en anglais XOR) entre l'empreinte MD5 et le mot de passe sur 16 octets.

Une technique est proposée pour crypter les mots de passe de plus de 16 octets sur 16 octets. Si pi est le iéme bloc de 16 octets du mot de passe le calcul est donné par :

b1 = MD5(Secret || Authentificateur)	c(1) = p1 ouex b1
b2 = MD5(Secret || c(1))	c(2) = p2 ouex b2
...	…
bi = MD5(Secret || c(i-1))	c(i) = pi ouex bi

Structure du message Access-Accept ou Accept-Reject

- Ces messages de réponse servent essentiellement par leur type à accepter ou à rejeter l'authentification.
- Le champ Authentificateur des messages Access-Accept ou Access-Reject contient une signature MD5 des informations suivantes concaténées (notion de 'Response Authenticator')

MD5 (Code de réponse ||
Identificateur de réponse ||
Longueur de réponse ||
Authentificateur (celui de la requête) ||
Attributs de réponse (si nécessaire) ||
Secret)

- Zones attributs (si nécessaire).

1) Dans le message access-request comment est utilisé l'authentificateur (à quoi sert il) ? Quelles sont les caractéristiques associées à l'authentificateur de requête pour que le secret ne puisse être décrypté ?

2) On remarque dans la méthode d'authentification RADIUS l'emploi de MD5 et du ou exclusif. Quelle est la méthode de chiffrement qui se rapproche le plus de la méthode employée pour le mot de passe (pourquoi le mot de passe ne peut-il être décrypté)?

3) Comment le serveur vérifie t'il l'identité de l'usager ? Peut-on utiliser un fichier des mots de passe cryptés comme en UNIX ou doit-on avoir les mots de passe en clair sur le serveur RADIUS?

4) Le client peut-il également authentifier le serveur ?

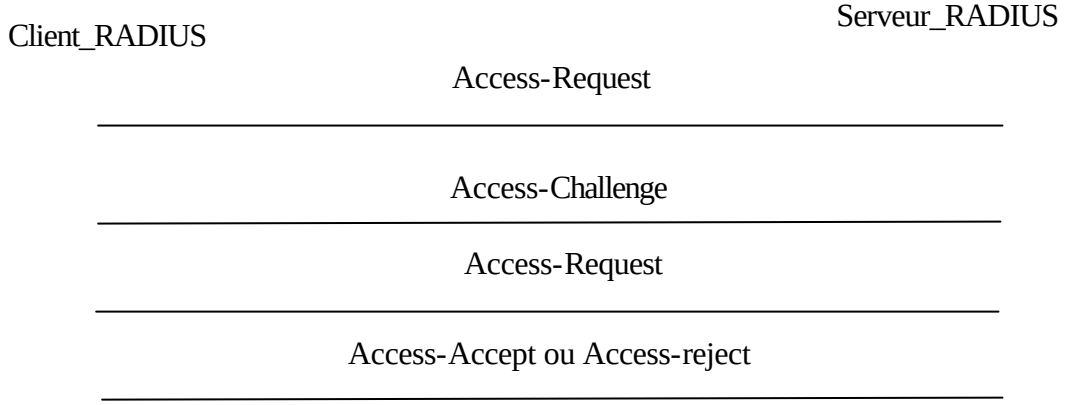
5) La solution résiste t'elle à une attaque par répétition ?

Version complète du protocole d'authentification

Cette authentification est définie dans la norme pour renforcer la sécurité par exemple lors de l'authentification automatique d'un usager possesseur d'une carte à puce. Elle comporte deux échanges requête/réponse.

Après une première requête (message access-request déjà vu) le serveur ne répond pas par une acceptation ou un rejet. Il répond par un message de type access-challenge.

Le second échange requête/réponse comporte une nouvelle requête d'authentification access-request qui est alors acceptée ou rejetée.



Structure du message Access-Challenge

- Le champ authentificateur contient une valeur qui est un authentificateur de réponse ('Response Authenticator') calculée d'une façon exactement identique à celle décrite dans le protocole de base pour les messages Access-Accept ou Access-Reject.
- Un attribut de type 'state' contient une chaîne binaire définie à la discrétion du serveur.
- Un autre attribut de type 'reply_message' contient une zone d'information. Il s'agit en fait d'une chaîne de texte à caractère explicatif.

Structure du second message Access-Request

- A la réception de access-challenge le client doit répondre par un Access-Request analogue à celui du premier échange avec cependant les modifications suivantes:

. utilisation d'un nouvel identificateur

. utilisation d'un nouvel authentificateur

. la chaîne binaire fournie par le serveur dans l'attribut state doit être concaténée au mot de passe avant l'utilisation de celui-ci.

6) A quoi sert le message Access-Challenge (quels sont les principes généraux de ce protocole d'authentification)?

7) A quels types d'attaque résiste le protocole complet? A quels types d'attaque ne résiste t'il pas ?

Exercice 18 : Sécurisation des réseaux sans fils WIFI : les normes WEP

Le développement des réseaux locaux sans fils IEEE 802.11 connus sous le sigle commercial WIFI ('WIreless FIdelity') est très rapide et très important. Les réseaux locaux sans fils (Wireless LANs) permettent de déployer de petites infrastructures de réseau local sans avoir besoin de câbler les bâtiments. Dans la plupart des cas, ces réseaux sans fils fonctionnent pour acheminer et recevoir du trafic de l'Internet mondial auquel le réseau est connecté via un point d'accès. Le système informatique qui joue le rôle de point d'accès possède quand à lui une liaison filaire avec l'Internet.

Dans ce problème on étudie les techniques de sécurité qui ont accompagné la première génération du WIFI (802.11 de base en 1997, 802.11b en 1999). Ces mécanismes de sécurité sont réunis sous le sigle WEP ('Wired Equivalent Privacy').

1) Le problème de sécurité est un problème considéré comme majeur dans un réseau sans fil. Pourquoi (citez différentes attaques qui rendent ces infrastructures particulièrement vulnérables) ?

Authentification dans le WEP

En matière d'authentification, le WEP définit principalement deux méthodes baptisées authentification ouverte et authentification à clé partagée. Nous examinons aussi une solution utilisant les adresses de niveau liaison (adresses MAC).

a) L'authentification ouverte

Chaque réseau Wifi est caractérisé par un identifiant unique baptisé SSID ('Service Set Identifier'). Cet identifiant est défini pour distinguer les réseaux WIFI. Une station doit donc connaître le SSID du réseau dans lequel elle souhaite s'insérer. La méthode d'authentification consiste à faire vérifier par le point d'accès qu'une station qui dialogue avec le point d'accès présente le bon SSID identifiant du réseau. L'identifiant SSID présenté par une station circule en clair dans les messages. Dans le mode PCF, cet identifiant peut également être diffusé en clair par le point d'accès dans certains messages.

2) Quelle est la sécurité offerte par la méthode d'authentification par l'identificateur de réseau SSID?

Une autre technique pour authentifier les usagers d'un réseau sans fils n'est pas normalisée mais elle est assez souvent implantée. Elle consiste à se baser sur les adresses de niveau liaison (adresses de la carte réseau ou adresse MAC 'Medium Access Control'). En effet, de façon exactement identique à un réseau local Ethernet, chaque station d'un réseau WIFI possède une adresse unique de niveau liaison. Dans certains réseaux WIFI, le point d'accès maintient une liste des adresses MAC autorisées à utiliser ses services. Si une adresse MAC, fournie comme adresse source par une station, n'appartient pas à la liste, cette station ne peut communiquer via le point d'accès.

3) Quels sont les avantages et inconvénients de la méthode d'authentification par les adresses MAC (particulièrement du point de vue de la sécurité)?

c) L'authentification à clé partagée

Le WEP suppose qu'une clé secrète est partagée entre toutes les stations et le point d'accès. Dans cette méthode d'authentification, lorsqu'une station apparaît dans un réseau, elle émet une requête d'authentification, le point d'accès lui répond par un message contenant en clair un nonce. La station doit répondre en retournant la valeur du nonce chiffrée en RC4 au moyen de la clé secrète. Le chiffre RC4 est le chiffre utilisé pour le chiffrement en confidentialité du WEP et il est décrit plus loin. Si le point d'accès peut vérifier que le nonce est correctement chiffré alors l'authentification réussit sinon elle échoue.

4) Rappelez la définition d'un nonce. Quels sont les avantages et inconvénients de la méthode d'authentification en WIFI par clés partagées (particulièrement du point de vue de la sécurité)?

Confidentialité dans le WEP

La confidentialité dans le WEP est basée sur le chiffre à clé secrète RC4 ('Rivest Cipher' n° 4). C'est le quatrième chiffre inventé par Ron Rivest, l'un des inventeurs du RSA. Il existe aussi des chiffres RC5, RC6. Le chiffre RC4 est officiellement propriété de la société RSADSI ('RSA Data Security Incorporated', société qui détenait le brevet du RSA). En dehors du réseau local 802.11 WIFI, RC4 est retenu dans un très grand nombre de logiciels très diffusés comme MS Access, Adobe Acrobat, Oracle Secure SQL, SSL, Lotus Notes, chiffrement de mots de passe Windows, ...

RC4 est un chiffre symétrique, utilisant une clé secrète qui doit être connue des entités communicantes autorisées (clé secrète partagée). Dans le WEP on peut utiliser deux tailles de clés: 40 bits (5 octets) ou 104 bits (13 octets).

Le chiffre RC4 est généralement présenté sous la forme des deux algorithmes suivants.

A) Initialisation d'un tableau aléatoire S de 256 octets à partir de la clé secrète stockée dans le tableau key

Ce premier algorithme initialise un tableau S (utilisé dans le second algorithme pour chiffrer). Cette initialisation prend comme base la clé secrète rangée dans un tableau d'octets key de longueur key_length. Par exemple en WEP on a key_length = 5 ou 13. Le tableau S est de 256 octets. A la fin de l'initialisation, S contient toutes les valeurs possibles des octets (de 0 à 255) distribuées aléatoirement. Cette distribution est obtenue à partir de la clé dans le tableau key par une suite d'échanges de cases dans le tableau S. Pour cela on réalise des calculs d'adresse modulo 256 (puisque le tableau S possède 256 entrées) pour déterminer les cases que l'on échange.

```
j := 0 ;                    (Initialisation de la variable entière j)
pour i = 0..255
  S[i] := i ;                    (Initialisation du tableau S de 256 cases)
pour i = 0..255                  (Permutation du tableau S)
  j := (j+S[i]+key[i mod key_length]) mod 256; (Choix de case modulo 256)
       aux:=S[i] ; S[i]:=S[j] ; S[j]:=aux ;     (Echange de S[i] et S[j])
```

B) Boucle de chiffrement octet par octet

Cet algorithme définit le chiffrement de chaque octet en utilisant le tableau S. Pour chaque octet à chiffrer on commence par permuter à nouveau le tableau S. Pour cela on fait un calcul d'adresse et

on échange deux cases du tableau S. Pour le chiffrement proprement dit, on commence par choisir une valeur stockée dans le tableau S. Cet octet, noté k, résulte d'un calcul d'adresse et de l'extraction de la valeur de la case du tableau S correspondante. k est utilisé pour réaliser le chiffrement effectif, par un ou exclusif avec l'octet en clair à chiffrer.

On rappelle la définition du ou exclusif : $0 \oplus 0 = 0$, $0 \oplus 1 = 1$, $1 \oplus 0 = 1$, $1 \oplus 1 = 0$

Avec cette définition on rappelle que le ou exclusif est son propre inverse (l'opération inverse du ou exclusif , la soustraction, est le ou exclusif $x \oplus x = 0$).

```
i:= 0; j := 0 ;
répéter                                  (Boucle pour tous les octets à chiffrer)
  i := (i+1) mod 256 ;                   (Choix d'une case i)
  j := (j+S[i]) mod 256 ;                          (Choix d'une case j)
  aux := S[i] ; S[i] := S[j] ; S[j] := aux ;       (Echange de S[i] et S[j])
  k := S[(S[i]+S[j]) mod 256] ;                    (On prend une valeur k dans S)
  octet_chiffré := octet_en_clair ⊕ k ;            (Å ou exclusif avec k)
jusqu'à fin du texte en clair ;
```

5) Si l'on utilise l'opération de chiffrement RC4 spécifiée par les deux algorithmes précédents, quels sont les algorithmes utilisés pour le déchiffrement (commentez votre réponse) ?

6) Quelle est la méthode de chiffrement vue en cours qui se rapproche le plus de ce chiffre (citez les points de rapprochement qui vous font proposer votre choix)?

7) Quelles sont les différences principales entre le RC4 et le chiffre vu en cours que vous proposez ?

8) On suppose qu'on utilise le chiffre RC4 sur deux messages différents avec la même clé. Un attaquant écoute le trafic chiffré. Cet attaquant peut connaître sur certains messages tout ou partie du texte en clair. Par exemple, parce que le format de certains messages est connu et que de nombreuses valeurs qui correspondent aux différents champs sont connues (messages complets du protocole WIFI connus, existence d'entêtes de messages connues avec les protocoles IP, TCP, HTTP, SMTP …). On suppose donc que sur un message un attaquant connaît un octet en clair et il connaît la valeur chiffrée qui circule (octet_chiffré). Sur un second message, utilisant la même clé, l'attaquant connaît seulement l'octet chiffré. En utilisant la technique de chiffrement RC4 basée sur le ou exclusif montrez que l'attaquant peut déchiffrer l'octet du second message ?

En WEP, pour ne pas utiliser sur chaque message la même clé secrète (on vient de voir que c'est très mauvais), on complète la clé secrète partagée en lui rajoutant en tête une valeur de 24 bits. Cette valeur est appelée vecteur d'initialisation. De la sorte, pour un message donné, la clé RC4 utilisée est composée de trois octets de vecteur d'initialisation suivis de 5 ou 13 octets de clé secrète partagée.

WEP ne spécifie pas de règle précise concernant le choix du vecteur d'initialisation. La norme indique seulement que le vecteur d'initialisation utilisé pour un message doit être transmis en clair avec ce message. Certaines implantations commencent toujours en prenant pour le premier message émis la valeur 0. Ensuite, le vecteur d'initialisation est géré comme un numéro de séquence. Il est simplement incrémenté pour chaque nouveau message émis.

9) Dans cette gestion du vecteur d'initialisation, on souhaite faire une première estimation de la réutilisation de la même clé. On fait un calcul approché en supposant un réseau WIFI fonctionnant à

54 megabits/s et transmettant uniquement des messages de longueur moyenne 1000 octets en permanence. Il s'agit d'une simple estimation car les protocoles ont un comportement beaucoup plus complexe qu'on ne peut résumer à une émission continue de messages de 1000 octets. Selon cette estimation, au bout de combien de temps réutilise t'on le même vecteur d'initialisation donc la même clé de chiffrement WEP (en ouvrant ainsi une possibilité de déchiffrement) ?

Il existe d'autres techniques de piratage nettement plus complexes non décrites ici qui permettent de casser une clé WEP en peu de temps.

Intégrité dans le WEP

En matière de contrôle d'intégrité, le WEP propose d'utiliser le code polynomial habituel des réseaux locaux IEEE 802 (on parle aussi souvent du CRC 'Cyclic Redundancy Check'). Ce code porte sur les données d'une trame et il est baptisé ICV ('Integrity Check Value'). Il est défini exactement de la même façon que dans Ethernet. Une zone de données en WIFI est complétée par un ICV de longueur 32 bits, obtenu comme le reste dans une division de polynôme.

On rappelle que si M(x) est le polynôme associé à un message M et CRC32(x) est le polynôme de degré 32 utilisé comme générateur du code polynomial pour tous les réseaux IEEE 802, alors le contrôle d'intégrité est défini par les coefficients du polynôme ICV(x) obtenu comme le reste dans la division de M(x) multiplié par x^{32} par CRC32(x). De manière précise on définit ICV par la division suivante dans laquelle Q(x) est le polynôme quotient :

x^{32} M(x) = CRC32(x) Q(x) $\oplus$ ICV(x)

Tout se passe du point de vue de la sécurité comme si le code polynomial ICV était considéré comme une fonction de hachage permettant la détection des modifications. De façon à protéger cette forme de hachage des modifications d'un pirate, le code polynomial ICV est chiffré en confidentialité comme les données du message en RC4. Pour une clé secrète K et une zone de données à chiffrer M, on transmet donc dans une trame WIFI, le message M chiffré en RC4 suivi de ICV chiffré en RC4. Si l'on note || la concaténation, la trame transmise contient donc RC4(K , M || ICV).

10) Montrer que si l'on transmet un message M dont le polynôme associé M(x) est de la forme M1(x) $\oplus$ M2(x) alors l'ICV associé à M est le ou exclusif des ICV de M1 et de M2 (on dit encore que le calcul d'ICV est linéaire soit ICV(x) = ICV1(x) $\oplus$ ICV2(x))

11) Un pirate intercepte un message M1 correctement construit et chiffré en WEP par son émetteur. Le pirate acquiert donc RC4(K , M1 || ICV1). Il ajoute (en ou exclusif) à ce message correct un message M2 en clair avec son code polynomial ICV2 correct. En fait il remplace RC4(K , M1 || ICV1) par le message modifié (M2 || ICV2) (RC4(K , M1 || ICV1) et le transmet au destinataire. Montrez que la vérification d'intégrité basée sur la vérification du code polynomial ICV chiffré ne détecte pas cette modification.

12) Au total que penser de l'ensemble des mécanismes de sécurité proposés dans le WEP. Comment faire mieux ?

Exercice 19 : Architecture de sécurité IPSEC

Le document qui suit est un chapitre de la norme qui définit l'architecture de sécurité des communications en IP (IP SECurity, RFC 'Request For Comments' numéro 2401). L'objectif poursuivi ici concerne la compréhension de cette norme. Le chapitre 5, qui a été sélectionné, s'intéresse au traitements de sécurité à appliquer aux datagrammes IP par IPSEC ('IP Traffic Processing'), tout d'abord en sortie ('Outbound', section 5.1) puis en entrée ('Inbound', section 5.2). Le document est suivi d'un ensemble de questions. Chaque question appelle en réponse une explication nécessitant la compréhension de l'un des mécanismes de sécurité évoqué dans le document, en relation avec votre connaissance de IPSEC. La compréhension peut nécessiter plusieurs lectures mais il n'est pas nécessaire de s'attacher à comprendre tous le document des le départ. On peut répondre aux questions successives en progressant dans le document. Par contre, les réponses aux questions posées qui seraient une simple traduction de l'anglais, n'ont pas d'intérêt.

```
RFC 2401              Security Architecture for IP         November 1998

5. IP Traffic Processing

   As mentioned in Section 4.4.1 "The Security Policy Database (SPD)",
   the SPD must be consulted during the processing of all traffic
   (INBOUND and OUTBOUND), including non-IPsec traffic.  If no policy is
   found in the SPD that matches the packet (for either inbound or
   outbound traffic), the packet MUST be discarded.

   NOTE: All of the cryptographic algorithms used in IPsec expect their
   input in canonical network byte order (see Appendix in RFC 791) and
   generate their output in canonical network byte order.  IP packets
   are also transmitted in network byte order.

5.1 Outbound IP Traffic Processing

5.1.1 Selecting and Using an SA or SA Bundle

   In a security gateway or BITW implementation (and in many BITS
   implementations), each outbound packet is compared against the SPD to
   determine what processing is required for the packet.  If the packet
   is to be discarded, this is an auditable event.  If the traffic is
   allowed to bypass IPsec processing, the packet continues through
   "normal" processing for the environment in which the IPsec processing
   is taking place.  If IPsec processing is required, the packet is
   either mapped to an existing SA (or SA bundle), or a new SA (or SA
   bundle) is created for the packet.  Since a packet's selectors might
   match multiple policies or multiple extant SAs and since the SPD is
   ordered, but the SAD is not, IPsec MUST:

          1. Match the packet's selector fields against the outbound
             policies in the SPD to locate the first appropriate
             policy, which will point to zero or more SA bundles in the
             SAD.

          2. Match the packet's selector fields against those in the SA
             bundles found in (1) to locate the first SA bundle that
             matches.  If no SAs were found or none match, create an
             appropriate SA bundle and link the SPD entry to the SAD
             entry.  If no key management entity is found, drop the
```

packet.

3. Use the SA bundle found/created in (2) to do the required IPsec processing, e.g., authenticate and encrypt.

In a host IPsec implementation based on sockets, the SPD will be consulted whenever a new socket is created, to determine what, if any, IPsec processing will be applied to the traffic that will flow on that socket.

NOTE: A compliant implementation MUST not allow instantiation of an ESP SA that employs both a NULL encryption and a NULL authentication algorithm. An attempt to negotiate such an SA is an auditable event.

5.1.2 Header Construction for Tunnel Mode

This section describes the handling of the inner and outer IP headers, extension headers, and options for AH and ESP tunnels. This includes how to construct the encapsulating (outer) IP header, how to handle fields in the inner IP header, and what other actions should be taken. The general idea is modeled after the one used in RFC 2003, "IP Encapsulation with IP":

- The outer IP header Source Address and Destination Address identify the "endpoints" of the tunnel (the encapsulator and decapsulator). The inner IP header Source Address and Destination Addresses identify the original sender and recipient of the datagram, (from the perspective of this tunnel), respectively. (see footnote 3 after the table in 5.1.2.1 for more details on the encapsulating source IP address.)
- The inner IP header is not changed except to decrement the TTL as noted below, and remains unchanged during its delivery to the tunnel exit point.
- No change to IP options or extension headers in the inner header occurs during delivery of the encapsulated datagram through the tunnel.
- If need be, other protocol headers such as the IP Authentication header may be inserted between the outer IP header and the inner IP header.

The tables in the following sub-sections show the handling for the different header/option fields (constructed = the value in the outer field is constructed independently of the value in the inner).

5.1.2.1 IPv4 -- Header Construction for Tunnel Mode

	<-- How Outer Hdr Relates to	Inner Hdr -->
	Outer Hdr at	Inner Hdr at
IPv4	Encapsulator	Decapsulator
Header fields:	----------	----------
version	4 (1)	no change
header length	constructed	no change
TOS	copied from inner hdr (5)	no change
total length	constructed	no change
ID	constructed	no change
flags (DF,MF)	constructed, DF (4)	no change
fragmt offset	constructed	no change

```
      TTL              constructed (2)             decrement (2)
      protocol         AH, ESP, routing hdr        no change
      checksum         constructed                 constructed (2)
      src address      constructed (3)             no change
      dest address     constructed (3)             no change
    Options            never copied                no change

       1. The IP version in the encapsulating header can be different

----------------------------
Document interrompu ici.
```

1) Le paragraphe d'introduction du chapitre prend comme point de départ la notion de SPD qui est un élément de base de IPSEC. Rappelez la définition et le rôle de la SPD. La norme souligne qu'un datagramme entrant ou sortant doit obligatoirement correspondre à une entrée de la SPD sinon le datagramme doit être détruit. Pourquoi?

Pour faciliter la lecture du texte nous fournissons les informations suivantes. Il existe trois façons d'ajouter des traitements de sécurité IPSEC à des équipements en réseau existants (trois types de solutions d'implantation d'IPSEC).

a) Par la modification du code du protocole IP en intégrant complètement IPSEC dans une implantation IP existante.

b) Une solution voisine de la précédente mais plus modulaire consiste à séparer le plus clairement possible le code de IPSEC du code IP standard. La partie de code spécifique à IPSEC apparaît alors comme un module séparé de IP qui est activé si nécessaire. On parle alors en anglais de solution "Bump-In-The-Stack" d'où le sigle BITS. Certains aspects comme la fragmentation et le réassemblage des datagrammes qui peuvent être nécessités par IPSEC, doivent quand même être traités par une modification dans le code du protocole IP (qui est alors assez légère).

Les deux solutions a) et b) sont applicables à tous les types d'entités réseaux (hôtes, routeurs ou passerelles de sécurité) par modification logicielle.

c) La solution baptisée "Bump-In-The-Wire" ayant pour sigle BITW consiste à faire réaliser la sécurisation IPSEC par un composant matériel/logiciel dédié, à l'extérieur de l'entité réseau demandant une sécurisation IPSEC. Il peut s'agir d'une passerelle de sécurité (un équipement de réseau dédié à la sécurité) ou d'un routeur jouant également le rôle de passerelle de sécurité.

2) A la fin du premier paragraphe de la section 5.1.1, la norme rappelle que dans la définition d'une politique de sécurité IPSEC, les règles sont ordonnées (elles sont exploitées dans un ordre qui est défini par l'administrateur de la sécurité). A quoi correspond cette notion d'ordre (pourquoi les règles sont-elles ordonnées) ? Indication : vous pouvez relier la définition d'une politique de sécurité IPSEC à la définition d'une politique de sécurité dans un filtre de paquets d'un pare-feu.

Le paragraphe 5.1.1 distingue deux cas possibles en ce qui concerne la consultation de la SPD. Les paragraphes associés aux deux cas commencent l'un par 'In a security gateway' et l'autre par 'In a host'). Dans le premier cas, on voit en lisant le texte, que chaque datagramme en émission commence par une consultation de la SPD. Dans le second cas le texte s'intéresse à l'application de IPSEC sur le trafic de datagrammes créé par une socket (communication de niveau transport en TCP ou en UDP). La SPD est consultée seulement à chaque fois qu'une socket est créée.

3) Expliquez cette différence de comportement entre les deux cas (quels sont les avantages et les inconvénients des deux méthodes). A quel type d'implantation serait plutôt associé le cas d'IPSEC fonctionnant pour une communication socket ?

4) Le premier paragraphe dans le point 5.1.1 fait référence à la notion de SA. Rappelez le rôle d'une SA ?

5) Le premier paragraphe du 5.1.1 fait aussi référence à une notion de "SA bundle" ('SA bundle' : littéralement un paquet ou une liasse de SA). En effet, dans certains cas, un trafic de communication IP est soumis à plusieurs SAs, chacune d'entre elle définissant une transformation particulière d'où la notion de SA bundle. Dans quelle circonstance enchaîne t'on plusieurs traitements en sécurité IPSEC? Quel intérêt a-t-on alors à acquérir ensemble les SA associées ?

6) On évoque dans le même paragraphe, par exemple dans la formule '`If no SAs were found or none match, create an appropriate SA'`, la possibilité de création dynamique d'une SA. On a donc un mode statique (une SA est créée au préalable) et un mode dynamique (une SA est créée lorsqu'un datagramme circule). Les deux modes sont adaptés à des réseaux de nature différente. A quels types de réseaux correspondent les deux modes de création des SAs ?

7) Comment peut-on dynamiquement créer une SA (donnez un ou des exemples de solution technique permettant cette création) ? Indication : le mode dynamique est principalement réalisé par les protocoles IKE/ISAKMP/OAKLEY.

Exercice 20 : IPSEC : Le protocole d'échanges de clés IKE/OAKLEY à secret pré partagé

La sécurisation du protocole IP (Internet Protocol) s'effectue dans le cadre des normes IPSEC ('IP Security'). IPSEC est très utilisé industriellement en raison du déploiement énorme du protocole IP. IPSEC définit des méthodes pour la confidentialité, l'authentification et l'intégrité des datagrammes IP.

Dans ce problème, nous étudions une partie importante de IPSEC associée aux protocoles d'échanges de clés. En effet un échange sécurisé en IPSEC commence par l'utilisation d'un protocole d'échange de clés globalement appelé IKE ('Internet Key Exchange' RFC 2407, 2408, 2409, 2412). IKE est considéré comme un protocole de session indépendant de IP. Il fonctionne en utilisant les services du transport UDP sur le port 500. Pour réaliser ses services, IKE combine des éléments issus de protocoles différents.

a) La partie ISAKMP ('Internet Security Association and Key Management Protocol') définit un cadre général pour l'échange de clés. ISAKMP décrit de manière générique les phases d'un échange de clés. ISAKMP définit la négociation des méthodes de sécurité utilisées et la façon de déterminer les paramètres associés aux méthodes de sécurité.

De manière générale IPSEC donc IKE, ISAKMP etc.. sont des protocoles très flexibles qui offrent la possibilité d'utiliser un ensemble très complet de méthodes de sécurité. Pour l'utilisateur de nombreux choix sont donc à effectuer. Ainsi un mode de fonctionnement sécurisé doit définir différentes informations de politique de sécurité : adresses IP autorisées, ports autorisés. L'utilisateur doit aussi choisir les fonctions cryptographiques qu'il souhaite (DES, 3DE, AES, MD5, SHA …) et enfin les protocoles de sécurité appliqués (à clés secrètes , à clés publiques …). Il faut ensuite définir les paramètres spécifiques qui doivent être utilisés pour appliquer la politique retenue (valeur des clés, etc…).

Pour spécifier tout ces choix, une association de sécurité (SA 'Security Association') est une structure de données qui stocke l'ensemble des paramètres associés à une communication donnée. De manière schématique, une association de sécurité est définie par une adresse IP, le protocole de sécurité utilisé, le pointeur dans un enregistrement d'une base de données contenant l'ensemble des paramètres de sécurité nécessaires au bon fonctionnement du protocole.

ISAKMP définit deux phases successives. La phase 1 comprend la négociation de l'association de sécurité utilisée pour les propres besoins de l'échange de clé (fonctions cryptographiques et protocoles utilisés par IKE). La phase 1 comprend ensuite un échange de clés secrètes le plus souvent réalisé par le protocole de Diffie-Hellman (il existe d'autres possibilités non étudiées ici). La phase 1 comporte enfin une authentification mutuelle des deux entités communicantes.

La phase 2 du protocole ISAKMP est la phase de négociation des associations de sécurité (SAs) utilisées dans les échange sécurisés IP ultérieurs.

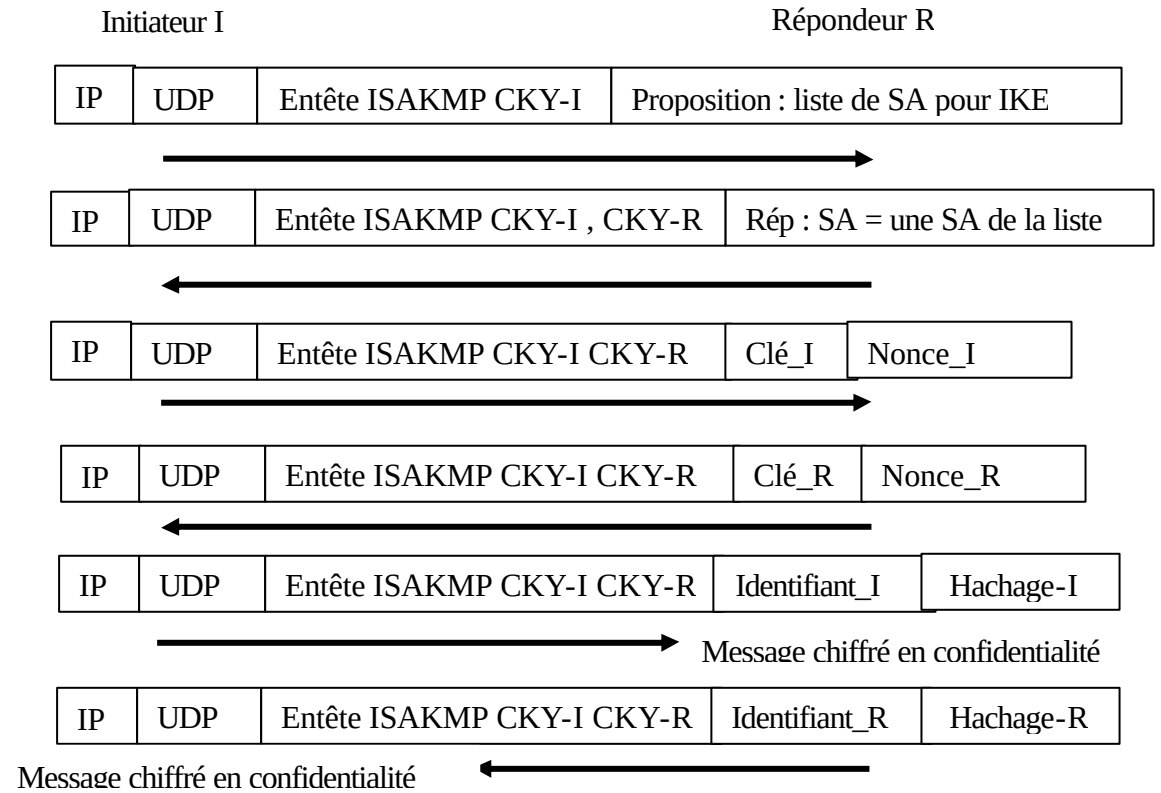
b) ISAKMP définit un cadre générique. Il faut donc définir des protocoles réels qui réalisent ISAKMP. Différentes propositions ont été effectuées pour réaliser ISAKMP. On peut citer les protocoles nommés SKIP, Photuris, OAKLEY ou SKEME qui définissent concrètement des formats de messages et des règles d'enchaînement de ces messages entre deux partenaires. Nous nous intéressons ici au protocole OAKLEY qui est l'un des plus utilisés.

Utilisation de clés échangées manuellement en OAKLEY

Nous étudions maintenant dans ISAKMP/OAKLEY la solution de base, baptisée 'Pre Shared Keys'. Dans cette solution, pour assurer la sécurité du protocole d'échange de clés, la même valeur d'une clé secrète PSK ('Pre-Shared-Key') doit être au préalable, affectée manuellement dans chacune des deux entités communicantes. Par exemple, un administrateur réseau doit se connecter successivement sur deux routeurs IP reliés et installer sur chaque routeur la valeur de la clé secrète PSK. D'autres solutions existent, par exemple une solution basée sur l'utilisation du chiffre à clés publiques RSA au moyen de certificats stockés dans un annuaire. Nous n'étudions pas cette solution ici.

La phase 1 du protocole ISAKMP/OAKLEY (qui négocie, échange des clés en Diffie-Hellman et authentifie mutuellement) est réalisée concrètement dans le cas des clés prè partagées en six messages. On appelle ce mode le mode de fonctionnement principal ('main mode'). Il existe un mode annexe plus court en trois messages baptisé 'agressive mode' que nous n'étudions pas ici.

Le diagramme en flèches d'échange des six messages montre que les deux premiers messages permettent de sélectionner les protocoles et méthodes de cryptographie utilisées dans les messages qui suivent. On y sélectionne une association de sécurité (une SA) utilisée pour la partie échange de clés IKE. Les deux messages suivants servent pour l'échange de clé de session. Les deux derniers messages servent pour l'authentification et l'intégrité. L'ouvreur d'une communication sécurisée est baptisé initiateur, le site distant est baptisé répondeur (terminologie utilisée dans la norme).



1) Dans l'entête ISAKMP du premier message, l'initiateur place ce qui est appelé en ISAKMP un cookie. Il est noté CKY-I. Le répondeur agit en retournant ce cookie CKY-I et en ajoutant son propre cookie CKY-R. On utilise ensuite le même couple de cookies dans tous les messages. Le choix du nom cookie et la présence dans tous les messages indique un objectif d'identification d'une communication. Les cookies doivent donc identifier de façon unique un échange de clés à la manière d'une référence de connexion. Le terme cookie est aussi associé à l'idée de gérer une approche de

fonctionnement client serveur sans état. Rappelez la définition d'un mode client serveur sans état ? Le protocole ISAKMP/OAKLEY peut-il être sans état (éventuellement jusqu'où) ?

2) ISAKMP indique que le cookie a surtout pour objectif de parer des attaques en déni de service (DOS 'Denial of Service'). On rappelle qu'un déni de service est obtenu quand un site (ici le répondeur) est soumis par un pirate à des demandes en avalanche (ici des échanges de clés) pour l'empêcher de travailler. Généralement cette attaque est préparée sur différents sites dont l'intrus a pris le contrôle et ces sites sont déclanchés simultanément ('flooding attack'). Peu importe que les requêtes échouent. En fonction des caractéristiques du protocole ISAKMP/OAKLEY, pourquoi le répondeur est-il sensible au problème de déni de service (ce qui justifie un mécanisme de prévention du déni de service)?

3) Pour parer des attaques en déni de service d'un répondeur, la norme indique que le cookie répondeur CKY-R doit être caractéristique du site qui le génère et vérifiable par ce site lorsqu'on le lui retransmet (dans les messages ultérieurs). Il doit donc reposer sur un secret connu du site qui génère le cookie. On recommande d'implanter le cookie en appliquant une fonction de hachage avec clé secrète à la concaténation de l'adresse IP source, IP destination, port UDP source, port UDP destination. Pour cela on peut utiliser l'un des MACs dont l'implantation est obligatoire en IPSEC : HMAC-MD5 ou HMAC-SHA-1 avec comme clé un secret choisi par le répondeur.

Imaginez une attaque en déni de service que ce cookie empêche (éventuellement des attaques)?

4) Avec ISAKMP/OAKLEY on utilise généralement la solution d'échange de clés de Diffie-Hellman. On voit que l'initiateur et le répondeur échangent des valeurs Clé_I et Clé_R. Quelle sont les valeurs définissant Clé_I et Clé_R selon le protocole de Diffie-Hellman que doivent s'échanger l'initiateur et le répondeur dans les zones clés du troisième et du quatrième message. Quel est ensuite le secret partagé entre l'initiateur et le répondeur selon le protocole de Diffie-Hellman ? On appelle dans la suite du problème cette valeur: clé_Diffie_Hellman.

5) Quels sont les avantages et quels sont les inconvénients de l'utilisation du protocole d'échange de clés de Diffie-Hellman ?

A la suite de l'échange Diffie-Hellman du troisième et du quatrième message le protocole ISAKMP/OAKLEY définit alors une méthode de calcul des clés secrètes qui vont être utilisées ensuite. En fait ISAKMP/OAKLEY ne propose pas d'utiliser directement la Clé_Diffie_hellman mais propose de calculer des clés dérivées. Dans le mode clé pré-partagée que nous étudions ici, la norme indique que les deux entités calculent tout d'abord une clé maîtresse baptisée SKEYID définie par la forme suivante dans laquelle la notation || définit la concatenation:

SKEYID = PRF (Pre_Shared_Key , Nonce_I || Nonce_R)

Dans cette formule, le sigle PRF définit une fonction qui est un générateur de nombres pseudo aléatoires (PRF est un abrégé pour 'Pseudo Random Function'). PRF peut donc être choisi par une implantation comme tout générateur de nombres pseudo-aléatoires à deux paramètres qui sont une clé (pour un générateur cette clé servira de graine) et un message M. En fait la norme indique aussi que PRF pourra être le plus souvent réalisée par une fonction de hachage à clé secrète c'est-à-dire un MAC(clé,M) (de clé secrète 'clé' et appliqué au message M).

A partir de SKEYID (clé maîtresse) la norme définit le calcul de trois clés dérivées, trois clés secrètes utilisées par la suite.

- SKEYID_e est une clé secrète utilisée pour protéger les messages du protocole d'échange de clés IKE/ISAKMP/OAKLEY en confidentialité. On l'utilise tout de suite sur les deux derniers messages.

- SKEYID_a est une clé secrète utilisée en phase 2 par l'échange de clés pour authentifier ses communications.

- SKEYID_d est une clé secrète utilisée comme clé maîtresse pour générer ensuite d'autres clés secrètes qui vont protéger les communications proprement dites en IP (sécurité des données usagers). C'est donc une clé utilisée par d'autres associations de sécurité négociées en phase 2 ISAKMP.

Ces clés secrètes sont définies comme suit :

SKEYID_d = PRF (SKEYID, clé_Diffie_Hellman || CKY_I || CKY_R || 0)
SKEYID_a = PRF (SKEYID, clé_Diffie_Hellman || CKY_I || CKY_R || 1)
SKEYID_e = PRF (SKEYID, clé_Diffie_Hellman || CKY_I || CKY_R || 2)

On voit qu'elles sont fabriquées à partir de SKEYID comme clé et avec un hachage de la concaténation de la clé de Diffie-Hellman, des deux cookies et d'un entier différenciant les trois clés qui est tout simplement sur un octet.

Les deux derniers messages de l'échange en six messages sont dédiés à l'authentification et à l'intégrité. Le paramètre principal est un identifiant de l'émetteur du message (Identifiant_I ou Identifiant_R selon les cas) qui est soit une adresse IP (sur 4 octets) soit un nom de domaine DNS (FQDN 'Fully Qualified Domain Name'). Deux mécanismes de sécurité sont utilisés dans ces deux derniers messages.

a) Les deux partenaires calculent des quantités baptisées HASH_I et HASH_R selon les formules :

HACHAGE_I = PRF (SKEYID, Clé_I || Cle_R || CKY-I || CKY-R || SA || Identifiant_I)
HACHAGE_R = PRF (SKEYID, Clé_I || Cle_R || CKY-I || CKY-R || SA|| Identifiant_R)

b) La norme indique que la charge utile du cinquième et du sixième message est chiffrée en confidentialité en utilisant la clé SKEYID_e. Donc dans le cinquième et le sixième message l'identifiant suivi du hachage sont chiffrés au moyen du chiffre à clé secrète sélectionné dans l'association de sécurité (par exemple DES, 3DES, AES) en utilisant la clé SKEYID_e.

6) Comment l'authentification mutuelle de l'initiateur et du répondeur est elle effectuée?

7) Pourquoi doit-on dans cet échange de clés s'occuper plus particulièrement de l'intégrité des messages échangés? Comment l'intégrité des messages est elle obtenue ?

8) Comment le secret de la clé pré partagée est-il protégé ?

9) A quoi servent les nonces I et R ?

10) Qu'apporte le chiffrement en confidentialité des deux derniers messages ?

11) Rappelez les propriétés d'une bonne clé secrète. Quels sont les mécanismes dans la génération des clés secrètes utilisées en IPSEC qui garantissent la qualité de ces clés ?

12) Quels sont les avantages et les inconvénients de ce mode de base d'échange de clés pré partagées (échange manuel de clés) du point de vue de l'administration de réseau (travail de l'ingénieur système réseau) ?

Exercice 21 : Sécurisation des communications en Internet avec SSL-TLS

SSL ('Secure Sockets Layer'), est une norme de réseau qui permet de sécuriser les communications pour des applications Internet utilisant TCP/IP. SSL offre un service de communication analogue à celui des sockets mais SSL ajoute aux communications standards, des fonctions de sécurité (authentification du client par le serveur, du serveur par le client, intégrité, confidentialité des données échangées) et éventuellement aussi des fonctions de compression.

Développé par Netscape jusqu'à la version 3.0 (novembre 1996), l'IETF a alors adopté SSL et a présenté sa version baptisée TLS ('Transport Layer Security' RFC 2246 en 1998) compatible avec SSL 3.0. Par rapport à SSL, TLS offre quelques extensions mineures comme une amélioration des signatures, différents traitements d'erreurs supplémentaires, ... En ce sens TLS 1.0 est parfois désigné comme SSL 3.1.

SSL/TLS est découpé en deux grandes parties. La partie 'Handshake' assure les fonctions initiales d'un échange sécurisé. La partie 'Record' assure les fonctions de sécurité sur les données utilisateur en appliquant des approches de cryptographie et de signatures définies pendant la phase de Handshake. Ce problème étudie plus particulièrement la partie Handshake.

La partie Handshake de SSL/TLS permet d'établir le contexte de sécurisation utilisé ensuite dans la phase d'échange de données. La partie Handshake permet l'authentification des entités communicantes. Elle permet également l'échange de clés de session.

Un contexte de sécurisation comporte :

- Un identifiant de session sécurisée choisi par le serveur.
- Un certificat d'entité distante (optionnel).
- Une méthode de compression (si la compression est appliquée).
- La suite cryptographique utilisée (voir plus loin).
- Une clé secrète ('master secret' 48 octets partagés entre client et serveur).
- Une variable indiquant si la session peut couvrir plusieurs connexions TCP.

Les opérations principales du protocole Handshake sont :

1. Négocier la suite cryptographique utilisée pendant le transfert des données.
2. Etablir une clé de session partagée entre le client et le serveur
3. Authentifier le serveur par le client (optionnel).
4. Authentifier le client par le serveur (optionnel).

En SSL/TLS une suite cryptographique est un choix relatif aux éléments suivants :

- La méthode d'échange de clés.
- La méthode de chiffrement utilisée pendant le transfert des données.
- La méthode de hachage utilisée pour la création d'une signature.

La méthode d'échange de clés peut se faire de deux façons. L'une utilise les algorithmes à clé publique et la notion de certificat. Une autre méthode est prévue en l'absence de certificats : la méthode de Diffie-Hellman.

Le chiffrement est réalisé au moyen d'un algorithme à clé secrète. Neuf algorithmes à clé secrètes avec des variantes sur les longueurs de clés sont possibles (DES, Triple-DES, IDEA, etc…)

La fonction de hachage à sens unique (Digest Function) peut également être sélectionnée (MD5, SHA-1)

En combinant les différents choix possibles dans les trois domaines précédents, la norme définit 31 suites de chiffrement cohérentes qui peuvent être adoptées après négociation.

1) Du point de vue du modèle OSI on considère que SSL-TLS est de niveau session alors que l'interface socket standard n'est pas de niveau session. Pourquoi ?

2) La plupart des utilisations de SSL/TLS comporte l'authentification du serveur par le client (bien que les fonctions d'authentification soient optionnelles). Citez une application de cette authentification. De manière générale pourquoi est il important d'authentifier un serveur ?

3) Pour un serveur, il est également possible avec SSL-TLS d'authentifier le client. Citez une application de cette authentification. De manière générale quelle est l'utilisation de cette authentification.

4) Le protocole SSL/TLS propose, dans l'une de ses modalités, d'utiliser la notion de certificat. Cette approche est d'ailleurs de loin la plus souvent retenue. A quoi sert un certificat. Rappelez les principaux champs d'un certificat ?

5) La vérification d'un certificat comporte différentes étapes. Quels sont les traitements à réaliser pour vérifier un certificat ?

6) Rappelez les principes d'une authentification en utilisant un algorithme à clés publiques ?

Les échanges du protocole Handshake sont assez complexes. En particulier ce protocole dépend des techniques de sécurisation utilisées (définies par le contexte de sécurisation négocié). On présente les principaux éléments du fonctionnement du protocole Handshake en omettant beaucoup de détails pour simplifier.

L'échange suivant est utilisé en cas d'authentification dans les deux sens au moyen de certificats. Ce protocole négocie le contexte de sécurisation, échange les certificats et les valide, construit un secret partagé sur 48 octets ('master secret') qui permet de fabriquer des clés de session et il échange des messages de terminaison.

Client		Serveur
ClientHello(protocolVersion, random, sessionID,Ciphersuite, compressMethod)	→	
	←	ServerHello(protocolVersion, random, sessionID,Ciphersuite, compressMethod)
	←	Certificate () CertificateRequest ()
Certificate () ClientKeyExchange () Finished()	→	
	←	Finished()

Explications des échanges

a) La première phase (messages ClientHello, ServerHello) correspond à la négociation du contexte de sécurisation. Le client envoie différentes informations proposant un contexte de sécurisation (premier message avec version du protocole SSL, un nombre aléatoire, un nonce, une suite cryptographique, une méthode de compression). Le serveur choisit les valeurs définitives du contexte de sécurisation acceptable en fonction de la proposition client (second message). Il fournit également un nombre aléatoire.

b) Dans le cas d'une authentification du serveur, le serveur fournit son certificat (message Certificate). Il demande le certificat du client s'il y a aussi authentification du client (message CertificateRequest).

c) Le client valide le certificat du serveur.

d) Le client envoie son certificat au serveur. Il créée un secret au moyen d'un générateur de nombres aléatoires ('pre master secret'). Le client l'envoie au serveur encrypté avec la clé publique du serveur (message ClientKeyExchange). Le client génère le secret partagé (master secret) à partir du 'pre master secret' et des deux nombres aléatoires échangés dans les deux premiers messages.

e) Le serveur valide le certificat client. Il décrypte le secret (pre master secret) envoyé par le client au moyen de sa clé privée. Il génère selon le même algorithme que le client le même secret partagé (master secret).

f) En utilisant le secret maintenant partagé (master secret), le client et le serveur génèrent chacun de leur coté la même clé secrète de session utilisable pour des algorithmes à clés privées (comme le DES).

g) Le client et le serveur échangent des messages de terminaison (Finished) chiffrés au moyen de la clé secrète de session. Ces messages indiquent que le protocole de Handshake est terminé et que les échanges auront lieu à partir de maintenant en utilisant le contexte de sécurité négocié. Ces messages doivent être déchiffrés et vérifiés.

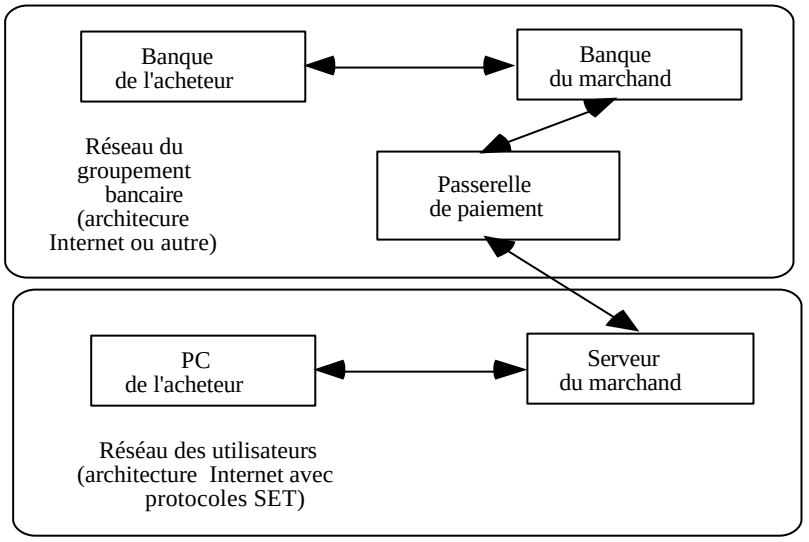
7) Quels sont les mécanismes du protocole Handshake qui assurent l'authentification du serveur vis à vis du client?

8) Le protocole SSL utilise une combinaison de méthodes de cryptographie qui comprend des algorithmes à clés publiques et des algorithmes à clés privées. Pourquoi utiliser concurremment les deux techniques?

Exercice 22 : Commerce électronique sur Internet (SET "Secure Electronic Transactions")

Pour sécuriser le commerce électronique sur Internet le standard SET a été proposé en 1996/1997. Il est soutenu par les principaux groupements de cartes bancaires Visa et Master Card ainsi que par de nombreux fournisseurs (IBM, Microsoft, ...). SET propose un ensemble de protocoles de sécurité pour le paiement par carte bancaire sur Internet. Il utilise pour cela des techniques de cryptographie (à clés publiques RSA, à clés secrètes DES, de fonction de hachage SHA-1). C'est une solution purement logicielle.

Une vision simplifiée de l'architecture de SET est donnée par la figure suivante ou apparaissent les différents calculateurs de l'acheteur, du marchand, des banques ainsi qu'une passerelle faisant interface entre le monde Internet (utilisant le protocole SET) et le réseau bancaire.



Le fonctionnement de base est analogue à celui des cartes de crédits habituelles. L'acheteur connecte son poste de travail sur le serveur du marchand. Il consulte le catalogue des produits proposés, passe commande et autorise le paiement. Le marchand accepte la commande et la réalise. Le marchand pour se faire payer adresse à sa banque l'autorisation de paiement de l'acheteur via la passerelle de paiement. On trouve donc trois protocoles essentiels dans SET:

- Le protocole d'achat.
- Le protocole d'autorisation de paiement.
- Le protocole de paiement

Voici (parmi de nombreuses autres) quelques règles de sécurité de base que les protocoles SET doivent respecter :

a) L'acheteur et la passerelle de paiement doivent pouvoir vérifier que le marchand est bien celui qu'il prétend être. Le marchand doit pouvoir vérifier que l'acheteur et la passerelle de paiement sont bien ceux qu'ils prétendent être.

b) Une personne non autorisée ne doit pas pouvoir modifier les messages échangés entre le marchand et la passerelle de paiement.

c) Le marchand ne doit pas pouvoir accéder au numéro de la carte de l'acheteur.

d) Le banquier n'a pas à connaître la nature de la commande passée par l'acheteur au marchand.

1) Les problèmes de sécurité associés aux règles a) b) c) précédentes sont des problèmes généraux traités en cours. Donnez pour les règles a) puis b) puis c) les noms des problèmes associés. Rappelez de manière succincte la définition de ces problèmes (en une phrase).

SET est un protocole applicatif, qui définit une politique de sécurité. A partir des éléments précédents on cherche à spécifier cette politique

2) Quels sont les quatre rôles qui doivent être considérés ?

On définit les objets suivants

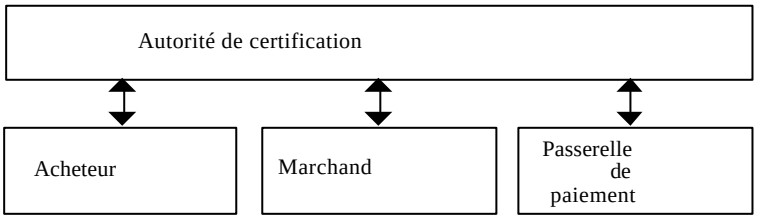
a) La carte bleue
b) Le PIN code de la carte bleue ("Personal Identification Number", le code secret)
c) Le numéro de la carte bleue
d) L'identifiant de la commande
e) Le contenu qualitatif de la commande (sa nature)
f) Le prix de la commande

Selon le cas on définit pour les objets précédents une ou plusieurs des méthodes suivantes:
A. Créer
B. Lire (connaître la donnée)
C. Accepter (signer la donnée)

3) Pour chaque objet donnez les méthodes applicables (A, B ou C)

4) Etablir la matrice des droits: Il s'agit d'une matrice ayant en colonne (au nombre de 13) les méthodes et en ligne (au nombre de 4) les rôles. Si un rôle X a le droit d'utiliser la méthode y l'élément X,y est marqué à 1 et n'est pas marqué sinon.

Pour mettre en oeuvre les protocoles de sécurité utilisant la cryptographie SET définit une phase d'accréditation préalable des acteurs par une autorité de certification (en fait une hiérarchie d'autorités).



Vue globalement, l'autorité de certification délivre des certificats aux différents acteurs des protocoles SET.

5) Qu'est ce qu'un certificat? Quelles en sont les propriétés principales?

6) On étudie maintenant le processus d'achat. Il se déroule en deux échanges requêtes réponses successifs.

Premier échange (l'échange initial)

La requête initiale de l'acheteur vers le marchand indique simplement en clair l'intention par l'acheteur de passer commande.

La réponse initiale du marchand comporte trois éléments:

- un identifiant de commande plus sa signature numérique
- le certificat du marchand avec sa clé publique
- le certificat de la passerelle de paiement avec sa clé publique.

A la suite de ce premier échange, quelles vérifications peuvent être effectuées par l'acheteur ?

7) Le second échange du processus d'achat comporte l'envoi de la requête d'achat et une réponse d'accusé de réception de commande.

Envoi de la requête d'achat

L'acheteur construit la structure de donnée commande qui a vocation a être communiquée au marchand (produits, quantités, prix avec l'identification de la commande fournie par le marchand pendant l'échange initial...). Elle est baptisée par la suite OI ("Order Information").

L'acheteur construit la structure de données de paiement qui a vocation a être communiquée à la passerelle de paiement (informations concernant la carte bancaire de l'acheteur et identification de la commande à payer fournie par le marchand pendant l'échange initial). Elle est baptisée dans la suite PI ("Payment Information").

En fait les deux structures de données sont liées. Le paiement ne concerne que la commande identifiée. Il doit être effectué que si la commande est acceptée par le marchand. La commande n'est effective que si la banque approuve le paiement. De plus le contenu de la commande doit être caché à la banque et le contenu des instructions de paiement doit être caché au marchand.

Pour lier les deux structures de données, l'acheteur calcule par l'algorithme SHA-1 la fonction de hachage de chacune des structures de données SHA-1(OI) et SHA-1(PI). Il applique à nouveau la fonction de hachage SHA_1 à l'ensemble (SHA-1(OI), SHA-1(PI)) des fonctions de hachage concaténées. Il chiffre cette dernière empreinte en RSA avec sa clé privée. C'est en fait une signature numérique double qui est réalisée. Elle est baptisée dans la norme SET signature duale.

$$\text{Signature duale} = \left\{\left\{\{OI\}_{SHA1}, \{PI\}_{SHA1}\right\}_{SHA1}\right\}_{RSA}^{clef\_privée\_acheteur}$$

Le message suivant est préparé pour la passerelle de paiement:

PI, Signature duale

L'acheteur choisit une clé aléatoire clé_aléa pour le DES. Le message a destination de la passerelle de paiement est chiffré en DES au moyen de cette clé.

$$\{PI, Signatureduale\}_{DES}^{clé\_aléa}$$

La clé DES est chiffrée au moyen de la clé publique de la passerelle arrivée avec le certificat de la passerelle.

$$\{clé\_aléa\}_{RSA}^{CLEF\_PUBLIQUE\_PASSERELLE}$$

Finalement le message de requête d'achat envoyé au marchand contient toutes les informations suivantes:

$\{PI, Signatureduale\}_{DES}^{clé\_aléa}$,
$\{PI\}_{SHA1}$,
$\{clé\_aléa\}_{RSA}^{CLEF\_PUBLIQUE\_PASSERELLE}$,
OI,
Signature duale,
Certificat de l'acheteur.

Envoi de la réponse du marchand à la requête d'achat

Le marchand construit un message de réponse qui a comme unique signification d'être un accusé de réception de la commande. Le marchand signe numériquement ce message (fonction SHA_1 et chiffre RSA avec sa clé privée). Il ajoute à l'ensemble son propre certificat.

7.1) Comment le marchand vérifie t'il l'intégrité de la commande OI?

7.2) Comment est réalisée la confidentialité des informations concernant la carte de crédit vis à vis du marchand?

7.3) Comment le marchand vérifie t'il que l'acheteur est bien celui qu'il prétend être?

Exercice 23 : Authentification des usagers et autorisation des requêtes dans le WEB

Le World Wide Web a tout d'abord été conçu comme un outil de diffusion de documents publics de sorte que peu d'efforts ont été effectués au départ pour contrôler l'accès aux informations offertes. Comme de plus en plus d'informations et surtout de services ont été distribués au moyen du WEB, des outils de sécurité ont été proposés pour satisfaire les besoins qui sont apparus. Ce sujet examine des solutions successives qui ont été développées dans le cadre du protocole HTTP pour répondre aux besoins d'authentification des usagers et d'autorisation des requêtes qu'ils émettent.

La version 1.0 du protocole HTTP (HTTP/1.0 RFC 1945 mai 1996) propose un mécanisme de base de contrôle d'accès utilisant une authentification à mot de passe (mécanisme 'basic').

Si un utilisateur client requiert une page protégée d'un serveur WEB sans fournir de couple usager, mot de passe, il reçoit en réponse un code d'erreur 401 ('unauthenticated'). Sur réception de ce diagnostic le navigateur du client demande alors à l'utilisateur un nom d'usager autorisé et son mot de passe au moyen d'un dialogue interactif dans une fenêtre. Lorsqu'une réponse est fournie le navigateur client réémet la requête vers le serveur avec les informations usager:mot_de_passe.
Lorsque l'on émet sur le réseau une requête avec le couple nom d'usager et mot de passe, ces informations sont codées mais non cryptées (enregistrement de la requête baptisé 'Authorization'). La méthode de codage en format texte ascii employée est baptisée Base64. Elle consiste essentiellement à découper les informations par groupes de 6 bits et à représenter les groupes de 6 bits par un caractère ASCII. Par exemple si on souhaite transmettre le couple usager:mot_de_passe "Aladdin:open sesame", il apparaît dans la requête une ligne de la forme :

Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==

De la sorte, si les mots de passe ne sont pas immédiatement lisibles dans les requêtes ('human readable'), il sont facilement décodables par un programme de décodage et peuvent donc être connus de tout le monde (les mots de passe ne sont pas cryptés).

1) Quels sont les avantages et les inconvénients en termes de sécurité et de coût de mise en œuvre d'une telle approche d'authentification ?

Le serveur WEB autorise une requête en consultant des fichiers créés par l'administrateur du serveur. Nous reprenons ici la protection de pages WEB par répertoires telle qu'elle est définie dans le cadre de l'outil NCSA Mosaic et reprise en version très voisine dans le serveur Apache. Le système d'exploitation est le système UNIX.

Supposons que l'administrateur d'un serveur WEB souhaite protéger un répertoire (baptisons le /mydir/turkey) contenant des pages WEB. Il doit créer dans ce répertoire un fichier dont le nom par défaut est .htaccess. Un exemple de fichier .htaccess est le suivant:

```
AuthUserFile /otherdir/.htpasswd
AuthGroupFile /dev/null
AuthName ByPassword
AuthType Basic

<Limit GET PUT>
require user daniel
</Limit>
```

Dans la première ligne de ce fichier on décrit où se trouve le fichier des mots de passe. Il est ici baptisé /otherdir/.htpasswd (c'est un fichier qui a pour nom .htpasswd et qui se trouve dans le répertoire otherdir). La seconde ligne indique qu'il n'existe pas de protection d'accès au niveau groupe d'utilisateurs, ce qui est indiqué par le fait que le fichier des protections de groupe est un flot vide (/dev/null). La chaîne associée à la troisième ligne (mot clé AuthName) peut être arbitrairement choisie par l'administrateur. Elle définit le nom du domaine auquel s'applique la politique de protection. Ce nom est transmis par le serveur au navigateur et il est affiché lors des demandes de mots de passe. Il permet à l'usager de savoir quel nom d'usager et quel mot de passe fournir (en fonction du domaine accédé). Ici le nom du domaine est un nom passe partout 'Bypassword'. La ligne 'AuthType' (type d'authentification) définit le protocole d'authentification comme étant 'Basic' c'est-à-dire celui que nous étudions ici. D'autres authentifications sont utilisables (PGP, KerberosV4, KerberosV5, ou Digest que nous examinons plus loin). On voit ensuite que dans ce fichier exemple seules les requêtes GET et PUT sont autorisées (enregistrement Limit GET PUT). On aurait pu définir ici une authentification pour d'autres opérations du protocole HTTP. Ici, l'autorisation est prévue uniquement pour l'usager daniel.

Il faut bien sur créer le fichier /otherdir/.htpasswd de mots de passe. Il contient des enregistrements de la forme usager:mot_de_passe. Les serveurs WEB disposent d'outils pour créer simplement par des dialogues interactifs ces fichiers de protection.

2) On suppose que l'administrateur du serveur WEB doit protèger d'autres pages WEB qui ont été créées dans un autre répertoire /mydir/goose. Ces pages appartiennent à un nouveau domaine baptisé realm. L'administrateur souhaite protéger l'accès à ces pages WEB en autorisant l'usager daniel pour les opérations GET, PUT et POST , et l'usager laurent uniquement pour les requêtes GET sur ces nouvelles pages. Quels sont les fichiers qui doivent être créés et à quel endroit doivent t'il se trouver?

Quand on réalise un service d'accès distant en réseau comme telnet ou de transfert de fichiers comme ftp, on commence par une ouverture de connexion ('login process') pendant laquelle on réalise une authentification de l'usager. Cette authentification demeure effective pendant toute une période considérée comme formant un tout du point de vue de la protection. Pendant une telle session un usager peut réaliser un ensemble d'opérations puis fermer la session quand bon lui semble.

3) Le protocole HTTP est il conçu selon le principe précédent, c'est à dire que l'accès à un ensemble de pages, d'images … d'un serveur forme un tout ou bien chaque accès est-il considéré comme indépendant des autres comme c'est le cas pour un serveur sans état (le serveur est il avec ou sans état) ?

4) Quelles sont les conséquences du choix précédent relativement au problème d'authentification (quelle technique peut-on proposer coté navigateur client pour simplifier la vie de l'usager à sa console) ?

A la version 1.1 du protocole HTTP (HTTP/1.1 RFC 2068 janvier 1997) est associée un autre protocole d'authentification baptisé « message digest authentication » défini par la RFC 2069 janvier 1997. Des améliorations ont encore été apportées au protocole de la version 'digest' par la RFC 2617 juin 1999.

On définit ici les principes généraux de la solution sans entrer dans les détails. Lorsqu'une réponse d'un serveur WEB à une requête d'un navigateur client est un rejet (code 401 'unauthenticated'), la réponse du serveur contient un champ particulier baptisé nonce qui contient

une valeur aléatoire à la discrétion du serveur. Ce message de rejet devient alors ce que l'on appelle un challenge. Au lieu de répondre par le nom d'usager et le mot de passe, le navigateur client doit alors répondre par un nom d'usager et à la place du mot de passe la valeur :

MD5(concat (nom d'usager, mot de passe, nonce))

Dans l'expression précédente, concat désigne l'opérateur de concaténation. On voit donc que le navigateur ayant obtenu un nom d'usager et un mot de passe doit fabriquer un texte qui concatène le nom d'usager, le mot de passe et le nonce. Ensuite il lui applique la fonction MD5 qui définit une fonction de hachage à sens unique. Le résultat est la valeur transmise (à la place du mot de passe).

La RFC 2069 propose d'utiliser par défaut dans le protocole d'authentification de type 'digest' la fonction MD5 ('Message Digest version 5). D'autres fonctions de hachage à sens unique peuvent être négociées.

5) Comment le serveur WEB réalise la vérification de l'autorisation d'accès à une page WEB protégée (le navigateur client ayant envoyé une requête avec nom d'usager, MD5(concat (nom d'usager, mot de passe, nonce)) comme expliqué plus haut) ?

6) Pourquoi avoir appliqué la fonction MD5 au mot de passe et en même temps au nonce (qu'est ce que cela apporte en termes de sécurité) ?

7) La méthode de vérification permet-elle de stocker les mots de passe dans le fichier des mots de passe sous une forme cryptée (comme dans les fichiers de mot de passe d'accès à un système UNIX) ou bien doit-on avoir sur le serveur le fichier des mots de passe en clair ? Quelle est la conséquence relativement à la sécurité de la méthode?

La valeur du nonce dépend de l'implantation du serveur ('implementation dependent'). La norme suggère néanmoins pour assister les implanteurs de serveurs WEB une valeur de nonce qui peut-être recalculée, de manière à ce que la valeur obtenue soit identique (presque toujours) entre un challenge et sa réponse.

Nonce = MD5 (concat (adresse_IP, ":", estampille , ":", clé_privée))

Adresse_ip est l'adresse IP de la machine du client.

Estampille est une valeur dépendante du temps qui ne change pas très souvent.

Clé_privée est une valeur secrète du serveur.

8) Pourquoi choisir une estampille temporelle qui ne change pas très souvent ? Quel est le risque encouru par cette méthode?

9) Pourquoi avoir choisi d'introduire dans le nonce l'adresse IP du client, une estampille temporelle, un secret (en quoi le choix des valeurs utilisées dans le nonce limite t'il le risque encouru) ?

Exercice 24 : Sécurisation du courrier avec S/MIME

S/MIME ('Secure/Multipurpose Internet Mail Extensions') définit un ensemble de standards pour transmettre de manière sécurisée des documents au format MIME.

1) Rappelez l'objectif poursuivi par la définition du format MIME. Citez deux grandes applications de l'Internet qui utilisent le format MIME.

S/MIME offre les grandes catégories de services de sécurité suivantes: authentification, intégrité, non-répudiation, confidentialité. Pour cela il permet de transférer de nouveaux types d'attachements qui sont des parties sécurisées. Une partie sécurisée est elle même une partie MIME (En tête + corps). On distingue trois types de parties dans le domaine de la sécurité pour lesquelles le standard définit un format précis (CMS 'Cryptographic Message Syntax' en S/Mime version 3):

Partie à signer ou à chiffrer ('data')
Partie chiffrée ('envelopped data')
Partie composant la signature d'un texte en clair ('siagned data')

2) A quoi sert chaque nouveau type ?

Un document signé est décrit en ASN1 par la définition suivante :

SignedData ::= SEQUENCE {
version **Version**,
digestAlgorithms **DigestAlgorithmIdentifiers**,
contentInfo **ContentInfo**,
certificates [0] IMPLICIT **ExtendedCertificatesAndCertificates** OPTIONAL,
crls [1] IMPLICIT **CertificateRevocationLists** OPTIONAL,
signerInfos **SignerInfos** }

3) Que peuvent contenir et quels sont les usages des variables dont les types sont:

DigestAlgorithmIdentifiers
ExtendedCertificatesAndCertificate
CertificateRevocationLists

4) Lorsque l'on transmet un document chiffré il est possible de le compresser également. Dans quel ordre effectue t'on les opérations: chiffrer, compresser, encoder en ASCII? (Justifiez votre réponse).

Exercice 25 : Sécurisation du DNS : les normes DNSSEC-1

De nombreuses attaques dans le réseau Internet sont conduites en utilisant les faiblesses en matière de sécurité du DNS dans sa version de base (non sécurisée). En effet, cette version qui est encore pratiquement la seule utilisée ne possède aucun mécanisme de contrôle d'intégrité ou d'authentification. Par exemple, si un attaquant arrive à modifier les informations fournies par un serveur DNS ou par un cache DNS, il peut remplacer l'adresse d'une machine rendant un service normal par l'adresse d'une autre machine contrôlée par lui. Avec cette nouvelle machine qui va recevoir toutes les requêtes, un pirate peut réaliser différents types d'attaques (espionnage, déguisement, attaque par le milieu).

L'un des problèmes essentiels à tous les niveaux dans le DNS est donc le maintien de l'intégrité des informations gérées et transmises par les différentes entités DNS. C'est l'objet de propositions réunies sous le nom d'ensemble DNSSEC (pour DNS SECurity extensions). Les normes DNSSEC sont encore en évolution. Nous étudions dans ce problème leur première version.

1) Dans le DNS rappelez le rôle des différentes entités communicantes suivantes: serveur primaire, serveur secondaire, cache et résolveur ?

2) Rappelez les grandes lignes des différents protocoles de communications que le DNS définit entre les différentes entités précédentes : serveur primaire, serveur secondaire, cache et résolveur.

TSIG ('Transaction Signature' RFC 2845)

L'une des solutions, proposée pour la sécurisation du DNS dans le cadre DNSSEC, est baptisée TSIG. TSIG s'applique aux messages échangés par les protocoles DNS en définissant une signature ajoutée à ces messages. TSIG permet d'authentifier l'entité DNS émettrice et permet de signer en intégrité des données fournies par cette entité DNS en utilisant une fonction de hachage et une méthode de cryptographie symétrique (à clé secrète).
Pour les besoins du protocole TSIG, on rajoute deux informations dans chaque message émis. Tout d'abord on rajoute la date et l'heure de l'émission du message (chaque message est horodaté au moyen de l'horloge temps réel de la machine émettrice). On rajoute surtout dans chaque message protégé, une signature calculée au moyen de la clé secrète et d'une fonction de hachage sécuritaire selon la méthode HMAC sur laquelle nous revenons plus loin.

La RFC recommande d'utiliser une clé secrète par paire de communicants. Pour distinguer les différentes clés utilisées, chaque clé possède un nom logique qui doit être connu de l'émetteur et du destinataire. Ce nom est transmis dans chaque message sécurisé afin de déterminer la clé utilisée à l'émission et à utiliser par le récepteur.

3) Compte tenu des principes généraux de fonctionnement de TSIG qui viennent d'être décrits, entre quelles entités communicantes dans le DNS le protocole TSG vous semble t'il le mieux adapté (justifiez votre réponse) ?

4) Pourquoi le protocole TSIG a-t-il prévu un horodatage des messages au moyen de la date et de l'heure d'émission de chaque message? Quel contrôle est effectué par un récepteur de message? Expliquez pourquoi le champ d'horodatage doit être signé ?

5) Quelles propriétés doivent vérifier les horloges de chaque entité DNS pour que les contrôles d'horodatage soient efficaces? Quel protocole complémentaire à TSIG doit on absolument mettre en œuvre pour que l'horodatage fonctionne correctement ?

De manière générale on appelle en sécurité MAC (Message Authentication Code), une information rajoutée à un message qui sert de signature en intégrité en utilisant une fonction de hachage et une clé secrète. HMAC signifie Keyed-**H**ashing **M**essage **A**uthentication **C**ode. HMAC est une méthode particulière de construction d'un MAC. HMAC est défini par la RFC 2104. C'est une méthode de signature assez largement utilisée (IPSEC, SSL, DNSSEC).
Pour construire une signature HMAC on utilise une clé secrète notée K et une fonction de hachage de sécurité notée H. Il peut s'agir par exemple de MD5 ou de SHA-1. De la sorte on a des HMAC-MD5 ou des HMAC-SHA-1.
Comme H n'est pas unique on note B le nombre d'octets généré pour chaque hachage (par exemple avec des condensés de 128 bits on a donc B=16 octets).
Si la clé K est trop courte (elle fait moins de B octets) on la complète avec des 0. Si la clé K est trop longue elle est tronquée à B octets par application de la fonction H.

0n définit deux chaînes de B octets utilisées pour faire du bourrage ('padding'):

Ipad = 0x36 répété B fois (l'octet 36 en hexadécimal répété B fois).

Opad = 0x5C répété B fois (l'octet 5C en hexadécimal répété B fois).

La signature HMAC pour un message M est donnée par:

HMAC(M) = H (K XOR Opad || H (K XOR Ipad || M))

Dans l'expression précédente, XOR est l'opérateur de ou exclusif. La double barre || indique la concaténation.

De manière à expliciter autrement la signature HMAC on peut encore dire qu'elle est obtenue au moyen des étapes suivantes de calcul:

A. Si K fait plus de B octets calculer K = H(K).
B. Si K fait moins de B octets ajouter des 0 à K de façon à ce que K soit de longueur B.
C. Calculer K XOR Ipad.
D. Concaténer le message M au résultat de 3.
E. Appliquer H au résultat de 4.
F. Calculer K XOR Opad.
G. Concaténer le résultat de 5 au résultat de 6.
H. Appliquer H au résultat de 7.

6) La clé K est une clé secrète. Quelles précautions doit-on prendre concernant sa longueur, sa génération et son stockage?

7) Sur quelles propriétés de H reposent la sécurité du mécanisme de signature en intégrité HMAC?

8) Rappelez la méthode classique de signature vue en cours? Quel est l'avantage de HMAC en terme de performance par rapport à la méthode classique de signature?

Nouveaux enregistrements ressources KEY, SIG (RFC 2535 à 2539)

Une autre solution proposée dans le cadre DNSSEC pour l'authentification et l'intégrité repose sur la création de deux nouveaux types d'enregistrements ressources (KEY et SIG) qui peuvent être stockés dans une base de données DNS.

L'enregistrement KEY a pour objectif de stocker des clés selon un format propre au DNS. Comme l'utilisation de clés ne concerne pas uniquement le DNS, l'enregistrement KEY peut aussi

servir pour d'autres protocoles de sécurité (au niveau réseau avec IPSEC, au niveau transport avec TLS ou au niveau application avec SMIME).

La structure de l'enregistrement KEY est la suivante :

{Nom , TTL, Classe=IN, Type=KEY, (Indicateurs, Protocole, Algorithme, Valeur de clé)}

- Un nom d'enregistrement KEY peut appartenir à différents types permettant différentes utilisation de la clé. Dans le cas le plus usuel (sécurisation du DNS), il s'agit d'un nom de zone DNS pour lequel on définit une clé (par exemple cnam.fr). Un nom d'enregistrement KEY peut également être un nom de machine ou un nom d'utilisateur.
- Les indicateurs ('flags') définissent différents champs précisant l'utilisation de la clé (quel est le type du champ nom, …)
- Le champ protocole définit le protocole de sécurité utilisant la clé (par exemple 3 : DNSSEC, …)
- Le champ algorithme définit l'algorithme de chiffrement auquel est destiné la clé (par exemple 3 : DSA …).
- La valeur de la clé (codée au format base 64).

L'enregistrement ressource SIG est placé après un enregistrement ressource donné. Il contient une signature pour cet enregistrement ressource. La technique de signature est la technique habituelle (ce n'est pas une signature HMAC). De la sorte, on peut interroger un serveur DNS sur un enregistrement ressource (mode de fonctionnement de base) mais on peut aussi demander la signature de cet enregistrement ressource. Pour améliorer les performances on peut aussi enregistrer une signature pour un ensemble d'enregistrements ressources.

La structure de l'enregistrement SIG est la suivante :

{Nom , TTL, Classe=IN, Type=SIG, (Type de l'enregistrement signé, Algorithme, TTL d'origine, Date d'échéance, Date de signature, Empreinte de la clé, Autorité signatrice, Signature)}

- Type de l'enregistrement signé (par exemple SOA).
- Champ algorithme (définit les algorithmes utilisés pour générer une signature).
- TTL d'origine (définit le TTL de l'enregistrement signé).
- Date d'échéance (la date limite d'utilisation d'une signature).
- Date de signature (définit la date de création de la signature).
- Empreinte de la clé (définit un résumé, une empreinte de la clé utilisée).
- Autorité ayant signé (définit le nom de l'autorité qui a généré la signature).
- Valeur de la signature (codée au format base 64).

9) A quel type d'algorithme de chiffrement est selon vous destiné l'enregistrement KEY.

10) Avec les nouveaux enregistrements KEY et SIG comment se passe une requête d'accès sécurisée dans le DNS?

11) Les enregistrements KEY sont des enregistrements ressources qui peuvent être protégés aussi par un enregistrement SIG ? A quoi sert l'enregistrement SIG dans ce cas ?

12) Une idée qui a été proposée pour vérifier une clé est d'utiliser la hiérarchie de nommage du DNS comme hiérarchie des autorités de certification. Selon cette approche comment fonctionnerait alors la vérification d'une clé (par exemple si un enregistrement est sous l'autorité d'une zone iie.cnam.fr, quels sont les niveaux d'autorités impliqués dans le contrôle de cet enregistrement, quels sont les contrôles réalisés par le destinataire)?

13) Les champs date d'échéance de signature et date de signature de l'enregistrement montrent que la validité d'une signature dépend du temps. Pourquoi? Quel protocole est indispensable pour valider correctement les signatures ?

Exercice 26 : Sécurisation DNS : Difficultés en DNSSEC-1, les normes DNSSEC-2

Les normes DNSSEC (Domain Name System SECurity') définissent un ensemble d'extensions au DNS pour en assurer la sécurité. Actuellement DNSSEC agit essentiellement en garantissant l'intégrité des données obtenues en réponse à une requête. DNSSEC sécurise également certaines opérations systèmes attachées à l'architecture DNS comme les transactions de mise à jour dynamique ou le transfert, entre serveurs primaires et secondaires, de la base de données relative à une zone. On rappelle qu'une zone est une partie de l'arborescence du DNS. La notion de zone correspond à l'idée de délégation de la responsabilité dans la fourniture des réponses à une requête. La responsabilité pour une zone est déléguée par la zone parente à une autorité qui administre une zone fille c'est-à-dire un ensemble de noms de domaines ayant comme préfixe le nom de domaine de la zone parente. La zone fille met en œuvre un ensemble de serveurs fournissant au reste de l'Internet les informations DNS relatives à cette zone. Les zones sont donc disjointes.

Nous nous intéressons dans ce texte à la garantie de l'intégrité. DNSSEC utilise pour cela uniquement la cryptographie à clefs publiques RSA dans le cadre de signatures RSA des données gérées par le DNS. Chaque zone va donc devoir disposer d'une paire de clefs (clés privées et clés publiques). De sorte que DNSSEC doit résoudre également les problèmes du stockage des clés et de la distribution des clefs publiques.

Rappels DNSSEC - 1

La première version DNSSEC - 1 (RFC 2535) adopte une approche assez naturelle pour la gestion des clés. En DNSSEC - 1, un couple clé privée, clé publique permet de signer puis de vérifier les informations relatives à une zone. On associe une signature à chaque RR ('Resource Record' ou enregistrement ressource) mais pour être plus efficace on signe aussi des 'RRsets' c'est à dire des ensemble d'enregistrements ressources qui ont le même nom et le même type mais des valeurs différentes. Ces RR regroupés sont habituellement transférés ensemble comme par exemple la liste de tous les serveurs de courriers dans un domaine. Les signatures sont stockées comme les enregistrements ressources dans la base de données de zone au moyen d'un nouveau type d'enregistrement ressource créé pour DNSSEC: le RR SIG. Outre la valeur d'une signature, l'enregistrement SIG comporte le nom du domaine possesseur de la clé ayant servi à construire la signature.

La clé publique utilisée qui permet de vérifier une signature est stockée dans un autre nouveau type d'enregistrement ressource créé pour DNSSEC : le RR KEY. Le RR KEY est lui aussi signé par un RR SIG.

Pour vérifier l'intégrité d'un enregistrement DNS, les deux enregistrements précédents (SIG et KEY) le concernant peuvent être obtenus par des requêtes DNS habituelles.

1) Le problème qui se pose alors est celui de valider la clé publique qui permet le contrôle d'une signature. On suppose tout d'abord qu'une zone choisit elle-même son couple clé publique, clé privée et qu'elle signe elle-même au moyen de sa clé ses enregistrements ressources et sa clé publique (solution dite d'auto signature pouvant être adoptée au début du déploiement de DNSSEC). Comment un client peut-il valider une signature dans ce mode d'auto signature ? Vous rappellerez les solutions que vous connaissez de distribution des clés publiques. Vous les évaluerez ?

2) Pour la mise en œuvre des signatures DNSSEC, on suggère quelquefois qu'il est préférable d'utiliser des clés plutôt courtes. Pourquoi cette proposition (discutez en les avantages et les inconvénients) ?

3) Quand des clés sont courtes quelle précaution doit-on prendre ?

Il est apparu très naturel, en raison de l'existence de la hiérarchie de nommage du DNS, d'associer une chaîne de certification à l'arbre du DNS. L'idée de base pour le fonctionnement des signatures devient la suivante. La zone parente doit signer la clé publique d'une zone fille.

4) En cas de renouvellement fréquent des clés, quel problème se pose alors ?

Pour certaines requêtes DNS la réponse est négative en ce sens que l'information demandée n'existe pas. Il n'y a pas de réponse quand une requête porte sur des noms inexistants ou sur des types d'enregistrements qui n'ont pas été définis pour un nom de domaine qui existe par ailleurs. Une réponse négative est significative et usurper une telle réponse peut permettre des attaques. Un nouveau type d'enregistrement dans le DNS baptisé NXT a été créé. Les enregistrements NXT sont insérés dans un fichier de zone entre les enregistrements correspondants à deux noms (d'où le sigle de 'next', le prochain). Un enregistrement NXT est associé à un nom de domaine. Il contient la liste des différents types de RR qui sont renseignés pour ce nom de domaine et il pointe sur le prochain nom de domaine figurant dans le fichier de zone. Le dernier nom de domaine de la zone pointe sur le premier nom de sorte qu'une base de données de zone est aussi une liste circulaire.

5) Quelle précaution doit-on prendre relativement à l'ordre des enregistrements ressource dans une base de données de zone.

6) Avec l'enregistrement NXT quels sont les problèmes de confidentialité qui se posent du point de vue de la diffusion des informations concernant l'architecture informatique d'une entreprise ?

DNSSEC - bis (DNSSEC - 2)

A la suite de l'expérimentation de DNSSEC version 1, différents problèmes ayant été relevés. Une nouvelle version a été proposée et adoptée (RFC 4033), de sorte que pour éviter toute ambiguïté les enregistrements ressources utilisés par la version 2 portent des noms différents de ceux de la version 1: SIG est devenu RRSIG, KEY est devenu DNSKEY, NXT est devenu NSEC. Nous étudions ici une seule modification importante. On propose en version 2 que chaque zone dispose de deux sortes de clés pour les chiffrements RSA. Ces clés seront donc a priori différentes à moins de leur donner la même valeur mais on ne gagne plus rien.

Les clés baptisées ZSK (pour 'Zone Signing Key') désignent les clefs utilisées pour la signature de pratiquement tous les types d'enregistrements (à une exception qu'on va voir). En ce sens une clé ZSK est analogue à la clé utilisée en signature dans la version DNSSEC -1 (une clé stockée dans l'enregistrement KEY).

Les clefs baptisées KSK ('Key Signing Key') servent dans la construction de la chaîne de confiance. Elles ne signent que d'autres clés. La KSK d'une zone est signée par la ZSK de la zone parente. Elle sert principalement à signer la ZSK de la zone.

La chaîne de confiance est alors assurée par la création d'un nouvel enregistrement ressource baptisé DS pour 'Delegation Signer'. Un enregistrement DS est localisé dans une zone parente pour créer un lien de sécurité avec une zone fille. DS contient une signature de la KSK de la zone fille (c'est-à-dire un hachage de la KSK de la zone fille chiffré par la ZSK de la zone mère).

7) Avec cette solution pourquoi devient-il beaucoup plus facile de changer de clé ZSK dans une zone ?

8) On suggère qu'avec DNSSEC bis les clés KSK doivent être longues. Pourquoi ?

9) Quels sont les inconvénients de cette solution ? Indication : listez les opérations à réaliser.

Exercice 27 : DNS et Messagerie Internet SMTP, lutte contre le courrier non sollicité

Différentes méthodes de lutte contre le courrier non sollicté ('spam') ont été proposées. Dans ce problème, nous étudions deux de ces méthodes qui sont basées sur les relations entre la messagerie SMTP et le service de nom de domaines DNS.

Dans ce texte, une entreprise connectée à l'Internet possède un nom de domaine (comme par exemple fai.com..). Cette entreprise gère un service de courrier électronique au moyen de différents serveurs de messagerie SMTP. Un hôte de l'Internet, dont l'adresse IP est a.b.c.d et dont le nom de domaine DNS est hote.domaine.. , utilise le protocole SMTP pour envoyer un courrier électronique à un destinataire d'adresse usager@fai.com.

1) Dans le DNS rappelez la définition de la notion d'enregistrement ressource (RR Resource record') ?

2) A quoi servent les types d'enregistrements A et MX ?

3) Quelles sont les actions que doit réaliser un client SMTP (qui tourne par exemple ici sur hote.domaine..) pour localiser un serveur de courrier avant de commencer les échanges SMTP (ces actions impliquent essentiellement le DNS) ?

Pour éviter une partie des courriers non sollicités ('spam') certaines entreprises pour effectuer une première catégorie de vérifications simples imposent les deux règles suivantes.

a) Tout hôte de l'Internet qui expédie des courriers électroniques à un serveur de courrier doit avoir obligatoirement un enregistrement ressource de type PTR dans sa base de donnée DNS permettant d'effectuer une recherche inverse.

b) Le nom de domaine d'un hôte émetteur de courrier doit posséder un enregistrement ressource de type A.

4) Comment l'enregistrement ressource de type PTR est il utilisé dans le DNS pour réaliser une recherche inverse ?

5) Quelles sont les informations dont dispose un serveur SMTP pour identifier l'émetteur d'un courrier électronique qu'il vient de recevoir (trois informations)?

6) Quelle sont les vérifications que veut effectuer un serveur SMTP d'une entreprise comme fai.com qui impose aux émetteurs de courriers les deux règles a) et b) (quelles sont les requêtes DNS utilisées) ?

7) L'administrateur d'un site émetteur de courrier, souhaite vérifier que son DNS est correctement configuré pour que ses courriers ne soient pas rejetés par une entreprise comme fai.com.. qui applique les règles a) et b). Il utilise l'outil nslookup disponible sous Unix ou Windows.

Quelles commandes doit-il taper pour vérifier qu'il satisfait les deux règles a) et b) (expliquez la formation des noms de domaines utilisés pour les lignes de commande) ?

Pour exprimer votre réponse vous pouvez utiliser aussi dig ou host si vous les connaissez mieux.

Cependant les deux règles a) et b) n'évitent plus beaucoup de courriers non sollicités car les entreprises qui font du spam le font, non pas à partir de machines dans leur domaine, mais à partir de machines appartenant à d'autres domaines de l'Internet qui ont été infestées par des virus chargés d'émettre les courriers indésirables. Une proposition de vérification qui améliore les règles a) et b) a été récemment acceptée par l'IETF. L'ensemble des nouveaux mécanismes est baptisé SPF pour 'Sender Policy Framework'. La nouvelle vérification est basée sur la création dans le DNS d'un nouveau type d'enregistrement ressource qui contient pour un domaine les machines habilitées à émettre du courrier.

8) Le nouvel enregistrement ressource a d'abord été défini en utilisant le type existant TXT. Pourquoi avoir utilisé ce type existant ?

Il existe de nombreuses variantes en SPF pour spécifier une machine habilitée à émettre du courrier. Une façon basique mais quand même très significative d'appliquer la politique SPF consiste à créer l'enregistrement suivant :

domaine. IN TXT "v=spf1 mx –all"

domaine. : le nom de domaine concerné par l'enregistrement ressource

IN TXT : enregistrement ressource type TXT pour le réseau Internet

"v=spf1" : une valeur obligatoire pour indiquer une utilisation du RR TXT en spf version 1

"mx" : spécifie que tous les serveurs de courrier du domaine sont des émetteurs autorisés

"-all" : indique que tous les courriers qui ne satisfont pas la contrainte sont rejetés.

Il existe différentes variantes syntaxiques permettant de spécifier des émetteurs autorisés : directement par le nom de domaine d'un hôte ou par une plage d'adresses IP autorisées etc …

9) Quelles sont les vérifications qui sont effectuées par un serveur SMTP d'une entreprise qui applique la politique SPF vis-à-vis d'un émetteur ayant inséré dans son DNS l'enregistrement ressource précédent (quelles sont les requêtes DNS utilisées) ? En matière de sécurité, à quelle approche s'apparente la technique SPF ?

10) L'utilisation de SPF permet de contrôler mieux le volume des courriers non sollicités. Expliquez l'ensemble des mécanismes que vous proposez reposant sur SPF pour effectuer ce contrôle.

Exercice 28 : Politiques de sécurité : pare-feux et filtrage des paquets

Un pare-feu (« firewall ») peut rassembler différents dispositifs de sécurité mais le plus souvent il se résume à un dispositif essentiel : un filtre de paquets qui fonctionne dans l'Internet.

Un filtre de paquet (« packet filter ») s'applique donc le plus souvent à des datagrammes IP. Pour le configurer on doit définir une politique de sécurité basée sur des ensembles de capacités. On présente le plus souvent ces capacités sous la forme d'un tableau. L'exemple suivant montre introduit simplement un exemple d'école des différentes informations que l'on va préciser :

Adresse source	Adresse destination	Port source	Port destination	Protocole	Autres champs	Action
163.173.128.36	163.173.128.40	2840	6032	TCP	Bit SYN=1	Autoriser
163.173.128.44	163.173.129.76	1030	5040	UDP		Interdire

On voit que les principaux champs utilisés pour le filtrage sont les adresses IP source et destination, les ports source et destination, le protocole transporté par IP qui est soit TCP soit UDP. On voit ensuite que tous les autres champs d'un paquet IP ou d'un segment TCP ou UDP sont susceptibles d'être l'objet d'un filtrage et sont regroupés dans une seule colonne. Ainsi, l'exemple montre qu'on peut définir pour autoriser une transmission, la présence obligatoire du bit SYN dans le segment TCP. On trouve enfin dans la dernière colonne, une action associée à la capacité qui est soit d'autoriser la communication soit de l'interdire. En résumé on peut donc définir une politique de sécurité portant non seulement sur l'adressage mais aussi sur tous les paramètres d'appels ou de réponse relatifs aux protocoles IP/TCP/UDP.

Selon les choix effectués dans un pare-feux industriel, la forme syntaxique qui permet de configurer une ligne du tableau est assez variable. Par ailleurs, pour définir la politique de sécurité (remplir les lignes du tableau) on doit aussi connaître certains aspects sémantiques du fonctionnement du filtre. Par exemple, nous supposerons ici que le filtrage est appliqué en sortie c'est-à-dire juste avant d'émettre un paquet sur l'une des interfaces de sortie du dispositif matériel qui supporte le filtre (par exemple un routeur ou un hôte). Nous supposerons également que les lignes du tableau (les capacités) sont exploitées l'une après l'autre dans l'ordre ou elles apparaissent. La première ligne qui est satisfaite déclenche l'action associée qui est soit de transmettre soit d'interdire la transmission (c'est-à-dire de détruire le datagramme). D'autres choix sont possibles.

1) Les règles de filtrage associées aux lignes du tableau précédent sont des capacités dans une politique de sécurité. Quel est le sujet, quel est l'objet et quel est le droit associé à la capacité ?
2) Dans une politique de sécurité en matière de communications, on peut avoir une attitude qui consiste à autoriser un ensemble limité de communications permettant aux utilisateurs de travailler et à interdire tout les autres communications par défaut.
Comment s'appelle le principe de sécurité qui gouverne un tel choix de politique. Quels sont les avantages et quels sont les inconvénients de cette politique ?

3) Donnez la forme générale d'une liste de capacités qui définit une politique conforme au principe de la question 2 (tout ce qui n'est pas autorisé est interdit). On notera une zone qui peut prendre toutes les valeurs possibles par une *.
4) Une autre version pour concevoir une politique de sécurité est de permettre par défaut tout ce qui n'est pas interdit. Quels sont les avantages et les inconvénients d'un tel choix. Donnez la forme générale d'une liste de capacités quand on adopte cette seconde stratégie.

5) Dans une connexion TCP pratiquement tous les segments ont le bit ACK positionné (présence d'un champ d'acquittement inséré dans le segment). Quels sont les segments qui n'ont pas le bit ACK positionné ?

6) Déduire de la question précédente, la forme d'une politique de sécurité (avec deux règles de filtrage) qui interdit à un processus (avec une adresse source port source) d'ouvrir une connexion TCP sur une adresse destination port destination mais qui lui permet de poursuivre une communication (une connexion) qui a été ouverte par l'autre extrémité (le processus peut fonctionner en serveur passif mais pas en ouvreur actif)?

7) Une entreprise possède un réseau d'entreprise connecté à l'Internet mondial au moyen d'un filtre de paquets. L'entreprise souhaite définir une politique de sécurité ou l'on interdit par défaut tout ce qui n'est pas autorisé. Pour la politique de sécurité relative aux accès à distance en Telnet on souhaite interdire toute connexion en provenance de l'extérieur du réseau d'entreprise (de l'Internet mondial) vers le réseau d'entreprise. Par contre on souhaite autoriser toutes les connexions en Telnet à partir d'un client (un poste de travail) connecté au réseau d'entreprise vers n'importe quel serveur. Donnez la liste de filtrage qui autorise le trafic Telnet en sortie et qui l'interdit en entrée (il est indispensable d'expliquer les choix effectués pour les différents champs d'une règle de filtrage) ?

Indications :

a) On considérera que l'adresse IP du réseau de l'entreprise est 163.173/16
b) Le port Telnet est le port 23.
c) On s'attachera à ce que les ports alloués aux services bien connus ('well known ports' des serveurs comme le port 23 de telnet) soient distingués des ports alloués dynamiquement aux clients qui demandent une connexion. On pourra considérer que les ports alloués dynamiquement ont toujours un numéro supérieur à 1023.

8) L'entreprise ne possède qu'un seul serveur de messagerie (MTA Mail Transfer Agent) qui fonctionne avec le protocole Internet SMTP. Le protocole SMTP est un protocole client-serveur. Le MTA de l'entreprise est un client SMTP pour d'autres serveurs dans l'Internet lorsqu'un membre du personnel émet un courrier. Le MTA est un serveur SMTP d'un client distant lorsqu'un membre du personnel reçoit un courrier. On cherche tout d'abord à définir en français les principaux échanges qui doivent être autorisés entre le serveur de messagerie de l'entreprise et les autres serveurs de messagerie distants dans l'Internet mondial. Quelles sont les quatre catégories de messages qui vont devoir être autorisés entre le serveur de messagerie de l'entreprise et ces machines distantes?

9) Quelle est la liste des capacités (des règles de filtrage) qui doivent être définies pour autoriser la circulation des courriers électroniques entre le serveur de messagerie de l'entreprise et le reste de l'Internet (il est indispensable d'expliquer les choix effectués pour les différents champs d'une règle de filtrage)?

Indications :

a) On considérera que l'adresse IP du serveur de l'entreprise est 163.173.128.20.
b) Le port SMTP est le port 25.
c) On s'attachera à ce que les ports alloués aux services bien connus ('well known ports' des serveurs comme le port 25) soient distingués des ports alloués dynamiquement qui ont toujours un numéro supérieur à 1023.

Exercice 29 : Environnement de sécurité SSH 'Secure Shell'

Sécurité des commandes à distance (commandes r 'remote')

Dans le système UNIX on peut utiliser les trois commandes suivantes : cp (copie d'un fichier dans un autre), sh (lancement de l'interpréteur de commande 'shell' pour exécuter une commande), login (ouverture d'une nouvelle session interactive sur un autre compte).

Avec l'arrivée de l'architecture de réseaux TCP/IP, l'université californienne de Berkeley a fourni en 1983 dans sa livraison Unix (Unix BSD 'Berkeley Software Distribution') un paquetage de trois commandes nouvelles permettant l'exécution à distance des trois commandes précédentes qui n'étaient possibles jusque là que localement.

Le nom qui a été donné aux nouvelles commandes adaptées à l'univers réseau, a été obtenu en préfixant par r (pour 'remote') le nom de la commande en mode centralisé soit rsh, rcp, rlogin. Ces commandes sont réalisées par un processus lancé sur le site client par un usager désireux d'exécuter la commande. Le client dialogue avec un processus serveur qui doit être lancé en permanence sur le site distant. Ce serveur est donc constamment en attente des demandes d'exécution des commandes client pour les traiter.

Par exemple, pour exécuter à distance la liste des fichiers dans le répertoire par défaut (commande UNIX ls), de l'usager 'gerard' (paramètre dans la commande –l pour login) qui possède un compte sur la machine ulysse.cnam.fr on exécute :

```
% rsh -l gerard ulysse.cnam.fr "ls"
```

L'émetteur de la requête doit alors fournir le mot de passe associé au compte gerard sur la machine ulysse.cnam.fr. Dans la version de base, qui reste le fonctionnement de nombreuses implantations des commandes rsh, rcp, rlogin, tous les échanges se font en clair.

1) Donnez la liste la plus large d'attaques que vous pouvez imaginer sur l'exécution des commandes distantes en mode de base. Il est obligatoire de justifier pour une attaque citée comme possible, le mode opératoire de l'attaque (décrire le fonctionnement de l'attaque et sa conséquence sur l'exécution des commandes distantes qui vient d'être décrit).

Afin d'éviter aux utilisateurs de devoir saisir, pour chaque commande distante, un mot de passe, on permet pour chaque compte utilisateur de créer dans le répertoire principal (le répertoire par défaut), un fichier dont le nom est par convention '.rhosts'. Ce fichier contient une liste des machines, qui sont autorisées à exécuter des commandes sans vérification du mot de passe, sur le compte associé au répertoire qui contient le fichier .rhosts. Les hôtes listés dans ce fichier sont baptisés hôtes de confiance pour un compte utilisateur donné (« trusted hosts »). Cette méthode des hôtes de confiance est également appliquée globalement pour l'ensemble des utilisateurs d'une machine au moyen d'un fichier système dont le nom est par convention /etc/hosts.equiv. Ce fichier contient une liste d'autorisation ou d'interdiction de l'utilisation des commandes r pour des ordinateurs ou des utilisateurs.

2) Comment appelle t'on en sécurité le contenu d'un tel fichier (définissez cette approche de la sécurité par comparaison aux différentes approches de sécurité vues en cours) ?

3) Quelles sont les failles en sécurité de la mise en œuvre des fichiers '.rhosts' ou hosts.equiv (donnez la liste la plus large d'attaques que vous pouvez imaginer) ?

Généralités SSH

La version initiale de l'environnement de sécurité SSH ('Secure SHell') a été proposée en 1995 par Tatu Ylönen (chercheur finlandais) pour permettre l'exécution sécurisée des commandes à distance Unix que nous venons de voir. T. Ylönen a donc introduit une nouvelle série de commandes préfixées par s pour sécurité. La principale nouvelle commande a été baptisée ssh ('secure shell') pour réaliser l'exécution sécurisée de toute commande à distance. Pour sécuriser des transferts de fichiers il a introduit des commandes comme scp ('ssh copy') et sftp ('ssh file transfer protocol'). Dans ce texte nous ne considérons que la sécurisation des commandes d'UNIX mais le protocole SSH peut-être utilisé aussi pour sécuriser des commandes Windows ou Mac OS. SSH est devenu avec le temps un logiciel de sécurité très répandu dans la sécurisation des applications réseaux.

Ce texte s'intéresse à la version 2 (SSH V2), normalisée à l'IETF après de longs débats en janvier 2006. Les RFC principales sont : SSH Protocol Architecture (SSH-ARCH RFC 4251), SSH Authentication Protocol (SSH-AUTH RFC 4252), SSH Transport Layer Protocol (SSH-TRANS RFC 4253), SSH Connection Protocol (SSH-CONN RFC 4254).

La version 2 de SSH hérite par de nombreux aspects de mécanismes développés pour la version 1. Mais la version 2 corrige des faiblesses de sécurité de certains mécanismes de la version 1. Par exemple on y introduit l'utilisation du protocole d'échange de clés de Diffie-Hellman, l'utilisation de macs …. La version 2 ajoute de nouvelles fonctions. Pour toutes ces raisons, elle est incompatible avec la version 1 qui est donc destinée à disparaître.

Au total, SSH version 2 offre la possibilité de construire plusieurs sessions concurrentes sécurisées (construire plusieurs VPN SSH) avec authentification forte, contrôle d'intégrité et chiffrement des sessions. Chaque session permet l'une des applications sécurisée suivante:

- Sessions de travail à distance ('remote login') : ssh, slogin remplaçant rsh, login
- Sessions de transfert de fichiers: scp ou sftp qui remplacent rcp ou ftp.
- Sessions de déport d'affichage multi fenêtre : pour le protocole X11.
- Sessions réalisant des tunnels chiffrés : pour acheminer des informations échangées par d'autres protocoles (SSH 'Port forwarding' soit relayage ou transfert de port).

SSH V2 est structurée en modules séparés alors que la version 1 était monolithique. L'organisation modulaire en couches de SSH V2 (au dessus de TCP) est la suivante :

<table>
<tr><td colspan="2">Applications sécurisées utilisant des tunnels SSH
Commandes à distance sécurisées SCP, SFTP, SSH, …</td></tr>
<tr><td>SSH-CONN
Sessions sécurisées SSH</td><td>SSH-AUTH
Authentification client SSH</td></tr>
<tr><td colspan="2">SSH-TRANS
Authentification serveur, Echanges confidentiels de données SSH</td></tr>
<tr><td colspan="2">TCP
Transmission Control Protocol</td></tr>
</table>

La couche SSH-TRANS réalise l'authentification du serveur, négocie les algorithmes utilisés (chiffrement, échange de clés, intégrité, compression), échange les clés de session, assure les fonctions de confidentialité et d'intégrité sur les données, réalise des fonctions de compression.

La couche SSH-AUTH réalise l'authentification du client avec différentes possibilités (par mot de passe, clé publique, hôte de confiance). Elle assure les changements de clé ou de mot de passe.

La couche SSH-CONN utilise la couche de transport pour multiplexer plusieurs sessions interactives sécurisées. On peut ainsi exécuter des commandes shell à distance, ou faire communiquer des applications multi-fenêtre (avec X11), ou effectuer des transferts de port TCP ce qui permet de sécuriser par SSH des applications non sécurisées en appliquant de manière transparente à ces applications les protocoles de sécurité SSH.

Mécanismes d'authentification du serveur

L'authentification, dans l'exécution de commandes à distance, est un problème majeur. En SSH, l'authentification est vue à deux niveaux. On authentifie tout d'abord un hôte serveur SSH i.e. une machine qui exécute un processus serveur SSH. Cette authentification réussie, ouvre un échange de clés de session qui permettent à toute la suite des communications SSH d'être chiffrées en confidentialité et contrôlées en intégrité. On peut alors commencer à traiter des commandes usager. On réalise alors un second type d'authentification : on authentifie individuellement les usagers qui demandent des accès à des comptes sur un hôte serveur.

Nous commençons par l'authentification des machines. Elle utilise uniquement une approche à clés publiques. Donc, au moment de l'installation de SSH sur un serveur, un administrateur système doit commencer par générer un couple clé publique/clé privée et l'attribuer à cette machine. Pour ensuite établir une relation SSH entre un client et un serveur, la machine cliente doit posséder la clé publique de la machine servante.

SSH prévoit que cette clé publique soit contenue dans un certificat, auquel cas sa protection et sa vérification sont définies dans le cadre d'une infrastructure à clés publiques. Cette solution, très répandue, présente les difficultés de mise en œuvre inhérentes aux PKI.

SSH propose comme alternative plus légère à une PKI, de pouvoir gérer les clés publiques sans structure de certificat. Dans ce cas, une clé publique est associée à un seul enregistrement dans un fichier ayant comme champs: le nom de domaine de la machine serveuse SSH possédant la clé, son adresse IP, l'algorithme à clé publique pour lequel la clé est définie, la valeur de la clé. Pour information voici un enregistrement de clé publique:

```
serveur_ssh.cnam.fr, 163.173.29.211 ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAIEA06
jFqviLMMJ/GaJNhGx/P6Z7+4aJIfUqcVjTGQasS1daDYejcfOAWK0juoD+zS3BsGKKYKPA5Gc5
M8v+3NHLbPn1yTpDBgl6UzA0iiMPCbwnOLx61MrBTk+/qJI9kyDaJf4LEY6Chx4IJP0ZN5NmAl
CtXQsca3jwFAF72mqPbF8=
```

a) Une première méthode recommandée pour faire connaître la clé publique d'un site serveur est de l'installer manuellement dans un fichier sur chaque site client.

Une clé publique de serveur est connue, pour tous les utilisateurs d'un système client, si elle a été installée dans un fichier système global dont le nom est par convention en Open-SSH /etc/ssh_known_hosts. Dans ce texte, parmi les différentes implantations des normes SSH, nous utilisons pour nos exemples la version libre Open-SSH.

La clé publique peut être connue d'un seul utilisateur si elle se trouve dans le fichier known_hosts qui est par convention localisé dans le sous répertoire .ssh du répertoire par défaut de cet utilisateur (fichier ~/.ssh/known_hosts ou ~ désigne de manière générale le répertoire par défaut d'un usager soit par exemple /users/deptinfo/gerard).

b) Dans certains cas, un usager client peut vouloir utiliser un serveur distant sans que la clé publique de cette machine ne soit connue du client. Dans ce cas, le client SSH demande et obtient du serveur sa clé publique par un échange de messages réalisée à l'ouverture de la connexion SSH. On propose alors à l'usager demandeur de mettre cette clé publique dans son fichier ~/.ssh/known_hosts.

4) Quel problème de sécurité peut poser la transmission en réseau de la clé publique d'un serveur SSH distant sans la structure de certificat ?

Le dialogue console suivant est une partie de celui qu'on peut constater lorsqu'un client tente pour la première fois d'établir une communication en Open-SSH avec un site distant dont le nom de domaine est serveur_ssh.cnam.fr et dont la clé publique ne figure pas dans fichier known_hosts :

```
% ssh serveur_ssh.cnam.fr
The authenticity of host 'serveur_ssh.cnam.fr (163.173.29.211)' can't be
established.
RSA key fingerprint is 98:2e:d7:e0:de:9f:ac:67:28:c2:42:2d:37:16:58:4d.
Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added 'serveur_ssh.cnam.fr, 163.173.29.211' (RSA) to
the list of known hosts.
```

Dans la liste précédente on constate que la transmission de la clé publique du serveur distant (serveur_ssh.cnam.fr) au client local (le processus qui exécute la commande ssh) n'a pas été jugée suffisamment authentifiée (c'est parce que la clé n'existait pas dans l'un des fichiers /etc/ssh_known_hosts ou ~/.ssh/known_hosts). L'implantation du client SSH a alors proposé à l'usager une valeur d'empreinte de la clé publique. 'RSA key fingerprint' désigne littéralement une empreinte 'digitale' pour une clé RSA. La méthode de calcul de l'empreinte n'est pas précisée par la norme SSH. Elle est laissée au choix des implantations (ici on peut constater le résultat du calcul d'empreinte par Open-SSH sur une valeur de clé publique). L'empreinte est calculée une première fois lors de la génération de la clé publique avec comme objectif sa diffusion par un moyen différent de celui utilisé pour la clé publique. L'empreinte est recalculée sur la valeur de clé publique échangée en réseau et elle est éditée. L'utilisateur est censé connaître cette empreinte pour valider par sa réponse 'yes/no' l'entrée de cette clé dans le fichier ~/.ssh/known_hosts de son compte. La méthode de diffusion de l'empreinte n'est pas non plus normalisée. Elle peut être manuelle. L'empreinte peut figurer dans une liste d'empreintes sur une page Web ou se trouver dans un fichier d'un serveur protégé, dans un annuaire LDAP, il a été aussi proposé un enregistrement ressource DNS pour les empreintes de clés SSH ….

6) La solution constitue une forme d'infrastructure à clés publiques. Evaluez la sécurité de la solution proposée (quelles doivent être les qualités de l'empreinte et de sa diffusion).

7) Lorsque la clé publique du serveur est connue du client, SSH en version 1 applique une solution d'authentification standard du serveur basée sur un chiffre à clé publique. Cette solution est utilisable également en SSH V2. Rappelez les principes du protocole d'authentification utilisant un chiffre à clé publique.

Mécanismes de confidentialité

Après l'authentification du serveur, on réalise un échange de clé de session (de clé secrète). La version 2 de SSH permet au début d'une connexion SSH, la négociation (le choix) d'une méthode d'échange de clé de session (nous verrons une solution à clé publique et une solution Diffie-Hellman) et le choix d'un algorithme de chiffrement à clé secrète comme le 3-DES-CBC (implanté obligatoirement en SSH) ou l'AES-128-CBC (d'utilisation recommandée). A partir de l'échange de clé de session, tous les échanges SSH sont chiffrés en confidentialité au moyen d'un chiffre à clés secrètes et sont aussi protégés en intégrité.

8) On peut choisir comme méthode d'échange de clés, la méthode de base qui a été définie pour la version 1 de SSH. Elle s'appuie directement sur la disponibilité d'un chiffre à clé publique. Cette méthode d'échange de clé de base est à l'initiative du client. Rappelez le mode de fonctionnement d'un échange de clés de session au moyen d'un chiffre à clés publiques. Avec un chiffre à clé publique, peut-on combiner en un même protocole, l'échange de clé de session et l'authentification du serveur ?

SSH V2 n'impose comme méthode obligatoirement implantée pour l'échange de clés, que la méthode de Diffie-Hellman (toutes les autres méthodes sont optionnelles). En SSH V2, l'échange de Diffie_Hellman est classique dans la mesure ou le client commence par transmettre sa valeur de Diffie-Hellman, le serveur répond en transmettant la sienne. Les deux communicants peuvent alors déterminer la clé secrète partagée. Une variante apparaît en SSH car le serveur ajoute dans son message de réponse, la signature d'un ensemble d'informations comportant les identifications des communicants, des contenus de messages précédents et aussi les deux valeurs échangées dans le protocole de Diffie-Hellman puis la valeur de la clé secrète partagée. Le serveur a déjà pu la calculer après le premier message du client.

9) Pourquoi le serveur ajoute t'il la signature de ces données. Cette signature peut-elle aussi authentifier le serveur ? Justifiez vos réponses.

10) SSH V2 proclame qu'il satisfait la propriété de secret parfait des clés (PFS 'Perfect Forward Secrecy'). Qu'est ce que cela veut dire ? Pourquoi cette propriété est-elle satisfaite ?

Mécanismes d'authentification du client

On doit maintenant authentifier l'utilisateur client. SSH propose trois modes d'authentification : à hôte de confiance, à mot de passe, à clé publique.

11) La solution des hôtes de confiance est très similaire à celle des commandes r (avec les fichiers hosts) en particulier du point de vue de la sécurité. Elle a été conservée pour des raisons de compatibilité avec l'existant et pour des configurations avec des risques limités en sécurité. Dans des réseaux d'entreprise un tant soit peu attaqués, elle est en général interdite. Qu'aurait-il fallu faire pour que l'approche des hôtes de confiance soit sécurisée ?

12) Le plus souvent, on privilégie en SSH l'authentification du client basée sur un chiffre à clés publiques contre l'authentification par mot de passe.
Citez différentes faiblesses de l'authentification à mot de passe même si l'échange d'authentification est chiffré en confidentialité comme en SSH.

Lorsqu'un nouvel utilisateur de SSH se présente dans une approche à clé publique, la première opération qu'il doit accomplir est de se doter d'un couple clé publique clé privée. Avec SSH en version 2, la génération du couple est effectuée selon un paramètre de type qui a comme valeur soit RSA soit DSA (commandes ssh-keygen –t rsa ou ssh-keygen –t dsa).

13) Expliquez la différence entre ces deux options. Selon vous pourquoi a-t-on offert cette possibilité en SSH version 2 ?

Un exemple d'exécution de commande de génération de clés en SSH en version 2 est donné plus loin. Il existe des différences, principalement de syntaxe selon les implantations de SSH (usage des noms de fichiers, conventions de paramètrisation, ..).

```
% ssh-keygen
Generating RSA keys: Key generation complete.
Enter file in which to save the key (/users/deptinfo/gerard/.ssh/identity):
Created directory '/users/deptinfo/gerard/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /users/deptinfo/gerard /.ssh/identity.
Your public key has been saved in /users/deptinfo/gerard /.ssh/identity.pub.
The key fingerprint is:
1024 39:82:19:c3:3a:f4:14:3a:b8:8b:a8:5b:72:bd:c7:b9
```

On observe, dans l'exécution de la commande ssh-keygen, que les clés publique et privée sont conservées dans deux fichiers identity et identity.pub du répertoire baptisé .ssh. On observe également que l'utilisateur s'est vu demander une chaîne de caractère (une phrase de passe) au prompt "Enter passphrase" qu'il a du répéter au prompt "again". On voit aussi que cette chaîne est facultative. Quand elle est fournie, elle est utilisée comme clé secrète d'un algorithme de chiffrement à clé secrète pour chiffrer le fichier contenant la clé privée. En conséquence, à chaque fois qu'on devra utiliser la clé privée, l'usager se verra demander la phrase de passe pour déchiffrer la clé privée. Quand la phrase de mot de passe n'est pas fournie par l'utilisateur lors de la création, celui-ci doit assurer par lui-même la protection de sa clé privée. On a donc deux approches de sécurisation d'une clé privée.

14) Comparez les deux approches de sécurisation de la clé privée (niveau de sécurité, aisance du procédé,…).

L'utilisateur d'une authentification SSH à clé publique doit également enregistrer sa clé publique sur le serveur SSH sur lequel il va s'authentifier. Il doit placer cette clé dans un fichier dans le sous-répertoire .ssh du répertoire par défaut du compte sur lequel il s'authentifie. Cette transmission doit être sécurisée. Une solution de base est de la sécuriser au moyen du mot de passe habituel d'ouverture de session sur un compte sur le serveur distant. A partir de là on peut authentifier l'usager client au moyen d'une solution a clé publique.

Dans le cas de l'utilisation d'une phrase de passe, la demande fréquente de cette phrase à un usager est fastidieuse. SSH propose une solution au moyen d'un agent pour fournir cette phrase à la place de l'usager.

Mécanismes d'intégrité

15) La version 1 de SSH utilisait le CRC-32 comme contrôle de l'intégrité des messages transmis. C'est-à-dire qu'on calculait pour chaque message échangé en SSH une redondance selon la même méthode que celle qui est utilisée dans tous les réseaux locaux (par exemple en Ethernet ou en Wifi). Cette méthode a été abandonnée en SSH version 2. Pourquoi ?

16) SSH version 2 utilise comme contrôle de l'intégrité un HMAC. Rappelez le mode de fonctionnement de cette méthode de contrôle d'intégrité. Pourquoi est-elle de sécurité ?