Partie XI

La sécurité

La sécurité est une fonction incontournable des réseaux. Puisqu'on ne voit pas son correspondant directement, il faut l'authentifier. Puisqu'on ne sait pas par où passent les données, il faut les chiffrer. Puisqu'on ne sait pas si quelqu'un ne va pas modifier les informations émises, il faut vérifier leur intégrité. Nous pourrions ajouter une longue suite de requêtes du même genre qui doivent être prises en charge par les réseaux.

Globalement, on peut diviser la sécurité en deux parties : la sécurité à l'ouverture de la session et la sécurité lors du transport de l'information. Les techniques pour réaliser ces deux formes de sécurité sont extrêmement diverses, et il s'en invente de nouvelles tous les jours. De même, les pirates, à chaque attaque contrée, vont un peu plus loin pour contourner les défenses. Ce jeu de poursuite n'est pas de nature à faciliter la présentation des mécanismes de sécurité. Dans cet ouvrage, nous nous limitons à l'environnement réseau, sans nous pencher sur la sécurité physique des terminaux ou des logiciels.

Le chapitre 38 propose une vue générale des éléments de sécurité dans un réseau, en suivant la proposition de l'ISO en matière de sécurité. Cette proposition a été effectuée en même temps que le modèle de référence. Nous présentons ensuite les mécanismes de sécurité les plus classiques, tels que l'autorisation, l'authentification, le chiffrement, la signature, etc. Le chapitre 39 s'intéresse plus particulièrement au monde IP et aux protocoles qui implémentent les mécanismes décrits au chapitre 38.

38

Vue générale des mécanismes de sécurité

La sécurité du transport de l'information est une préoccupation primordiale dans le domaine des réseaux. Pendant de longues années, la sécurité d'un équipement demandait une isolation complète de l'environnement extérieur, et aucune communication avec une machine externe n'était possible. C'est encore très souvent le cas aujourd'hui.

Ce chapitre examine les mécanismes fondamentaux de sécurité mis en œuvre dans les réseaux, ainsi que les algorithmes de chiffrement et d'authentification.

Les services de sécurité

En informatique, le terme sécurité recouvre tout ce qui concerne la protection des informations. L'ISO s'est attachée à prendre toutes les mesures nécessaires à la sécurité des données durant leur transmission. Ces travaux ont donné naissance à un standard d'architecture international, ISO 7498-2 *(OSI Basic Reference Model-Part 2: Security Architecture)*. Cette architecture est très utile pour tous ceux qui veulent implémenter des éléments de sécurité dans un réseau car elle décrit en détail les grandes fonctionnalités et leur emplacement par rapport au modèle de référence.

Trois grands concepts ont été définis :

- Les fonctions de sécurité, qui sont déterminées par les actions pouvant compromettre la sécurité d'un établissement.
- Les mécanismes de sécurité, qui définissent les algorithmes à mettre en œuvre.
- Les services de sécurité, qui représentent les logiciels et les matériels mettant en œuvre des mécanismes dans le but de mettre à la disposition des utilisateurs les fonctions de sécurité dont ils ont besoin.

La figure 38.1 recense les services de sécurité et les niveaux de l'architecture OSI où ils doivent être mis en œuvre.

Cinq types de service de sécurité ont été définis :

- La confidentialité, qui doit assurer la protection des données contre les attaques non autorisées.
- L'authentification, qui doit permettre de s'assurer que celui qui se connecte est bien celui qui correspond au nom indiqué.
- L'intégrité, qui garantit que les données reçues sont exactement celles qui ont été émises par l'émetteur autorisé.
- La non-répudiation, qui assure qu'un message a bien été envoyé par une source spécifiée et reçu par un récepteur spécifié.
- Le contrôle d'accès, qui a pour fonction de prévenir l'accès à des ressources sous des conditions définies et par des utilisateurs spécifiés.

Dans chacun de ces services, il peut exister des conditions particulières, indiquées à la figure 38.1.

Figure 38.1
Sécurité et niveaux de l'architecture OSI

	Niveau 1	2	3	4	5	6	7
Confidentialité							
avec connexion	Oui	Oui	Oui	Oui		Oui	Oui
sans connexion		Oui	Oui	Oui		Oui	Oui
d'un champ particulier						Oui	Oui
Authentification			Oui	Oui			Oui
Intégrité							
avec reprise			Oui	Oui			Oui
sans reprise				Oui			Oui
díun champ particulier							Oui
Non-répudiation							Oui
Contrôle d'accès			Oui	Oui			Oui

Si l'on reprend les cinq services de sécurité présentés précédemment en étudiant les besoins de l'émetteur et du récepteur et en les répertoriant, on obtient le processus suivant :

1. Le message ne doit parvenir qu'au destinataire.
2. Le message doit parvenir au bon destinataire.
3. L'émetteur du message doit pouvoir être connu avec certitude.
4. Il doit y avoir identité entre le message reçu et le message émis.
5. Le destinataire ne peut contester la réception du message.
6. L'émetteur ne peut contester l'émission du message.
7. L'émetteur ne peut accéder à certaines ressources que s'il en a l'autorisation.

Le besoin 1 correspond à un service de confidentialité, les besoins 2 et 3 à un service d'authentification, le besoin 4 à un service d'intégrité des données, les besoins 5 et 6 à un service de non-répudiation, et le besoin 7 au contrôle d'accès.

Les mécanismes de chiffrement

Le chiffrement est un mécanisme issu d'une transformation cryptographique. Le mécanisme inverse du chiffrement est le déchiffrement. La normalisation dans ce domaine est quelque peu complexe, pour des raisons essentiellement politiques. De ce fait, l'ISO a supprimé ce type de normalisation de son cadre de travail à la suite de la publication des algorithmes DES (Data Encryption Standard). L'ISO est alors devenue une simple chambre d'enregistrement des algorithmes de chiffrement.

La première norme ISO du domaine, ISO 9979, se préoccupe du problème relatif aux « procédures pour l'enregistrement des algorithmes cryptographiques ». Une vingtaine d'algorithmes sont aujourd'hui déposés à l'ISO ou chez d'autres organismes de normalisation. Des normes complémentaires, comme les procédures de chiffrement de messages (ISO 10126), les modes opératoires d'un algorithme de chiffrement par blocs de *n* bits (ISO 10116) ou les caractéristiques d'interfonctionnement avec la couche physique (ISO 9160) ont été publiés par l'ISO.

Les principaux mécanismes de chiffrement normalisés par l'ISO sont les suivants :

- Le mécanisme de bourrage de trafic consiste à envoyer de l'information en permanence en complément de celle déjà utilisée de façon à empêcher les fraudeurs de repérer si une communication entre deux utilisateurs est en cours ou non.

- L'authentification utilise un mécanisme de cryptographie normalisé par la série de normes ISO 9798 à partir d'un cadre conceptuel défini dans la norme ISO 10181-2. Dans cette normalisation, des techniques de chiffrement symétrique et à clés publiques sont utilisées.

- L'intégrité est également prise en charge par l'ISO. Après avoir défini les spécifications liées à la normalisation de l'authentification dans la norme ISO 8730, cet organisme a décrit le principal mécanisme d'intégrité, le CBC (Cipher Block Chaining), dans la norme ISO 8731. La norme ISO 9797 en donne une généralisation. La norme ISO 8731 décrit un second algorithme, le MAA (Message Authenticator Algorithm).

- La signature numérique est un mécanisme appelé à se développer de plus en plus. Pour le moment, la normalisation s'adapte aux messages courts, de 320 bits ou moins. C'est l'algorithme RSA, du nom de ses inventeurs (Rivest, Shamir, Adleman), qui est utilisé dans ce cadre (ISO 9796). Le gouvernement américain possède son propre algorithme de signature numérique, le DSS (Digital Signature Standard), qui lui a été délivré par son organisme de normalisation, le NIST (National Institute for Standards and Technology).

- La gestion des clés peut également être mise en œuvre dans les mécanismes de sécurité. Elle comprend la création, la distribution, l'échange, le maintien, la validation et la mise à jour de clés publiques ou secrètes. En matière d'algorithmes symétriques, la norme ISO 8732 fait référence. De même, la norme ISO 11166 fait référence pour les algorithmes asymétriques.

Les mécanismes de sécurité pour la messagerie électronique ont été définis par l'UIT-T, dans la série de recommandations X.400. Cette série fournit la description des menaces et les clés d'utilisation de l'algorithme cryptographique RSA pour résoudre ces problèmes.

Le second apport de l'UIT-T en matière de sécurité concerne les annuaires et fait l'objet de la recommandation X.509. Les annuaires électroniques peuvent également être le lieu de dépôt des clés publiques, et l'UIT-T a introduit des concepts de certificats de clés publiques et des mécanismes de gestion de ces certificats.

Les algorithmes de chiffrement

Les algorithmes de chiffrement permettent de transformer un message écrit en clair en un message chiffré, appelé cryptogramme. Cette transformation se fonde sur une ou plusieurs clés. Le cas le plus simple est celui d'une clé unique et secrète, que seuls l'émetteur et le récepteur connaissent.

Les systèmes à clés secrètes sont caractérisés par une transformation f et une transformation inverse f^{-1}, qui s'effectuent à l'aide de la même clé. C'est la raison pour laquelle on appelle ce système « à chiffrement symétrique ». Cet algorithme est illustré à la figure 38.2.

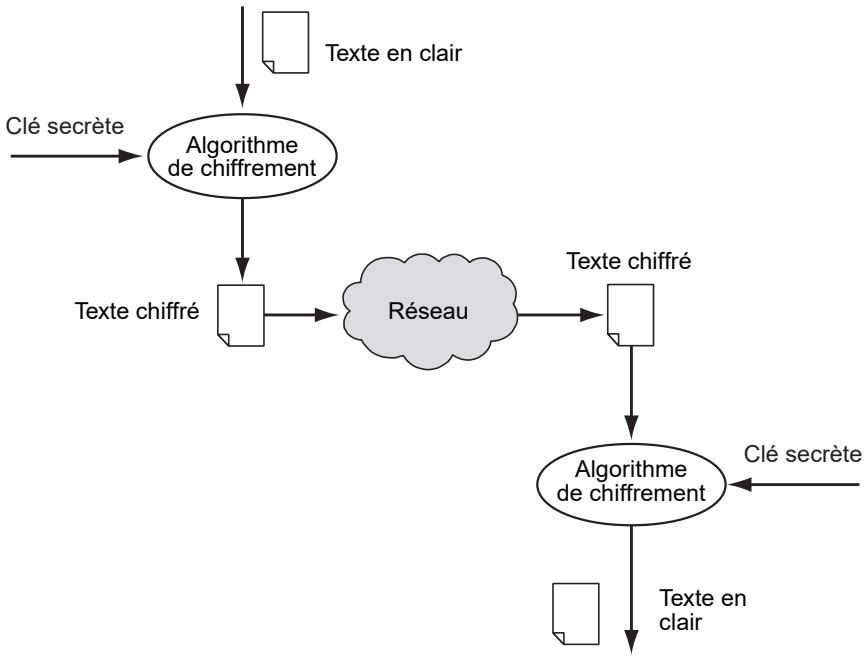


Figure 38.2

Algorithme de chiffrement symétrique

Le plus connu des algorithmes de chiffrement est le DES. Pour chaque bloc de 64 bits, le DES produit un bloc chiffré de 64 bits. La clé, d'une longueur de 56 bits, est complétée par un octet de détection d'erreur. De cette clé de 56 bits, on extrait de manière déterministe 16 sous-clés de 48 bits chacune. À partir de là, la transformation s'effectue par des sommes modulo 2 du bloc à coder et de la sous-clé correspondante, avec des couplages entre les blocs à coder.

Les algorithmes à sens unique sont ceux dont la transformation en sens inverse est quasiment impossible à effectuer dans un laps de temps admissible. Diffie-Hellman constitue

un premier exemple de ce type d'algorithme. Soit X et Y un émetteur et un récepteur qui veulent communiquer. Ils se mettent d'accord sur deux valeurs non secrètes, μ et p. L'émetteur X choisit une valeur a secrète et envoie à Y la valeur $x = \mu_a \bmod p$. De même, Y choisit une valeur b secrète et envoie à X une valeur $y = \mu_b \bmod p$. Si les valeurs μ et p sont suffisamment grandes, le fait de retrouver a ou b à partir de x ou y est à peu près impossible. X et Y décident que la clé commune est le produit ab et que le message chiffré est obtenu par $\mu_{ab} \bmod p$.

Les algorithmes de chiffrement à clé publique sont des algorithmes asymétriques. Le destinataire est le seul à connaître la clé de déchiffrement. La sécurité s'en trouve accrue puisque même l'émetteur ne connaît pas cette clé. L'algorithme le plus classique et le plus utilisé est RSA, qui utilise la quasi-impossibilité d'effectuer la fonction d'inversion d'une fonction puissance. La clé permettant de déchiffrer le message et que seul le destinataire connaît est constituée de deux nombres, p et q, d'environ 250 bits chacun. La clé publique est $n = pq$. Comme n est très grand, il est quasiment impossible de trouver toutes les factorisations possibles. La connaissance de n ne permet pas d'en déduire p et q. À partir de p et de q, on peut choisir deux nombres, e et d, tels que $ed = 1 \bmod (p-1)(q-1)$. De même, la connaissance de e ne permet pas de déduire la valeur de d.

L'algorithme de chiffrement s'effectue de la façon suivante : si M est le message à chiffrer, le message chiffré est obtenu par $M_e \bmod n$ et l'algorithme de déchiffrement par $(M_e)_d$.

La figure 38.3 illustre le fonctionnement de l'algorithme asymétrique.

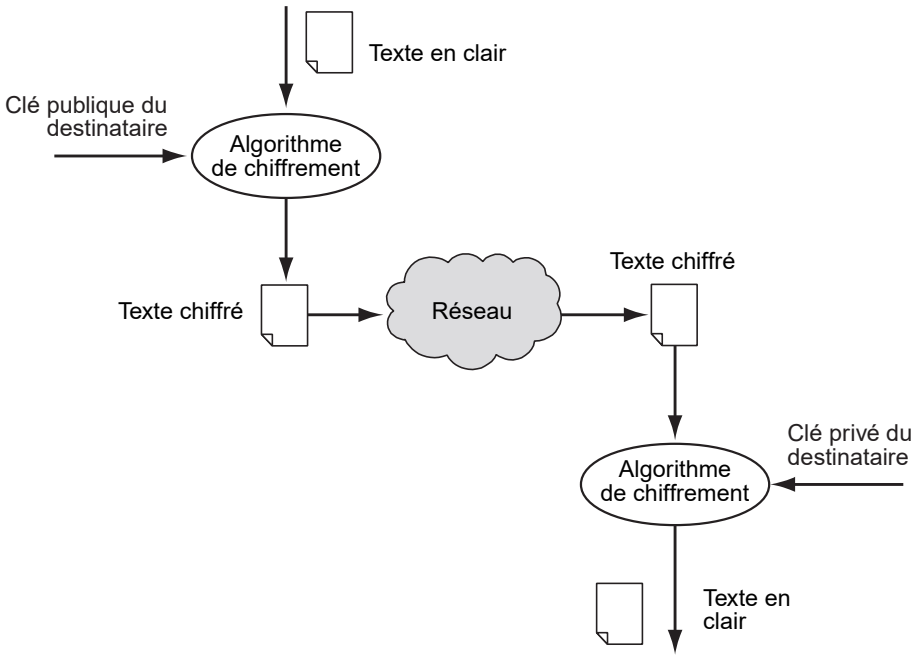


Figure 38.3

Algorithme de chiffrement asymétrique

Les signatures électroniques font également partie de la panoplie des mécanismes indispensables à la transmission de documents dans un réseau. La signature a pour fonction d'authentifier l'émetteur. Celui-ci code le message de signature par une clé qu'il est le seul à connaître. La vérification d'une signature s'effectue par le biais d'une clé publique. En utilisant l'algorithme RSA, l'émetteur signe le message M par M_e mod n, et le récepteur porte cette valeur à la puissance d pour vérifier que $(M_e)_d = M$. Si cette égalité se vérifie, la signature est authentifiée.

Solutions de chiffrement

Le chiffrement représente la méthode suivie pour que l'information ne puisse pas être lue par une autre personne que le destinataire. Les techniques de chiffrement que l'on utilise sont toutes *a priori* violables, mais il faudrait pour cela utiliser une machine de calcul extrêmement puissante et la faire tourner pendant plusieurs années.

Nous allons introduire les principaux algorithmes qui permettent de chiffrer une suite d'éléments binaires en la transformant en une nouvelle suite d'éléments binaires, qui, elle, ne peut être lue sans la clé de déchiffrement :

- DES, de 1977, à clés symétriques. Les données sont codées par blocs de 64 bits avec une clé de 56 bits. Cet algorithme est très utilisé dans les applications financières. Il est également utilisé dans un chaînage dit par bloc CBC (Cipher Block Chaining). Il existe de nombreuses variantes de l'algorithme DES, comme 3DES, qui utilise trois niveaux de chiffrement, ce qui implique une clé de chiffrement sur 168 bits.
- RC4, RC5 (Rivest's Code #4, #5), de 1987, à clés symétriques. Ce sont des algorithmes propriétaires, diffusés par la société RSA Security Inc. Ils utilisent des clés de longueur variable pouvant atteindre 2 048 bits. Ils sont surtout utilisés au niveau applicatif lorsqu'une application a besoin d'être fortement sécurisée. Ils demandent une puissance de calcul importante, qui ne pourrait être maintenue sur un flot continu à haut débit à des niveaux inférieurs de l'architecture.
- IDEA (International Data Encryption Algorithm), de 1992, à clés symétriques. Cet algorithme développé en Suisse est surtout utilisé pour la messagerie sécurisée PGP, que nous détaillons un peu plus loin dans ce chapitre.
- Blowfish, de 1993, à clés symétriques.
- AES (Advanced Encryption Standard), de 1997, à clés symétriques.
- RSA (Rivest, Shamir, Adleman), de 1978, à clés asymétriques (RFC 2437).
- Diffie-Hellman, à clés asymétriques.
- El Gamal, à clés asymétriques.

L'ensemble des techniques que nous venons d'énumérer est difficile à mettre en œuvre dès que le débit d'une application, d'un flot ou d'un lien augmente. C'est la raison pour laquelle, les techniques symétriques et asymétriques sont utilisées conjointement. Pour cela, on recourt à des clés de session, qui ne sont valables que pour une communication déterminée. Les informations de la session sont codées grâce à une clé secrète permettant de réaliser un chiffrement avec beaucoup moins de puissance qu'une clé asymétrique. Seule la clé secrète est codée par un algorithme de chiffrement asymétrique pour être envoyée au destinataire.

Les certificats

Une difficulté qui s'impose à la station d'un réseau qui communique avec beaucoup d'interlocuteurs consiste à se rappeler de toutes les clés publiques dont elle a besoin pour récupérer les clés secrètes de session. Pour cela, il faut utiliser un service sécurisé et fiable, qui délivre des certificats. Un organisme offrant un service de gestion de clés publiques est une autorité de certification, appelée tiers de confiance. Cet organisme émet des certificats au sujet de clés permettant à une entreprise de les utiliser avec confiance.

Un certificat est constitué d'une suite de symboles (document M) et d'une signature. Le format de certificat le plus courant provient du standard X.509 v2 ou v3. La syntaxe utilisée est l'ASN.1.

L'authentification

L'authentification a pour objectif de vérifier l'identité des processus communicants. Plusieurs solutions simples sont mises en œuvre pour cela, comme l'utilisation d'un identifiant (login) et d'un mot de passe (password). L'authentification peut s'effectuer par un numéro d'identification personnel, comme le numéro inscrit dans une carte à puce, ou code PIN (Personal Identification Number).

Des techniques beaucoup plus sophistiquées, comme les empreintes digitales ou rétiniennes, se développent de façon industrielle en ce début des années 2000. Cependant, leur utilisation est assez complexe et ne peut être mise en place que dans un contexte particulier, comme un centre de recherche de l'armée.

L'authentification peut être simple ou mutuelle. Elle consiste essentiellement à comparer les données provenant de l'utilisateur qui se connecte à des informations stockées dans un site protégé. Des attaques sur les sites mémorisant les mots de passe forment une classe importante de piratage.

L'intégrité des données

L'intégrité des données consiste à prouver que les données n'ont pas été modifiées. Elles ont éventuellement pu être copiées, mais aucun bit ne doit avoir été changé.

Une première possibilité pour garantir l'intégrité des données transportées dans un paquet est de les chiffrer. En effet, si les données sont impossibles à déchiffrer par le récepteur, c'est qu'elles ont été modifiées. Cette solution permet à la fois de garantir la confidentialité et l'intégrité.

Une seconde possibilité provient des techniques de signature. Une signature, déterminée par l'ensemble des éléments binaires composant un message, est nécessaire pour en assurer l'intégrité. Le chiffrement joue le rôle de signature dans la première possibilité. Une signature plus simple que le chiffrement est suffisante dans le cas d'une demande d'intégrité uniquement. Pour cela, on utilise des fonctions de hachage, qui calculent une empreinte digitale qu'il suffit de vérifier au récepteur pour prouver que la suite d'éléments binaires n'a pas été modifiée. Pour que l'empreinte ne puisse être modifiée par hasard lors de la transmission, c'est-à-dire pour que le pirate ne puisse à la fois déterminer l'algorithme de hachage utilisé et recalculer une nouvelle valeur de l'empreinte sur la suite d'éléments binaires modifiés, une fonction de chiffrement doit être appliquée à la signature.

Les plus célèbres techniques de signature sont les suivantes :

- MD5 (Message Digest #5), de 1992, défini dans la RFC 1321. Ce sont des fonctions conçues par Ron Rivest qui produisent des empreintes de 128 bits.
- SHA-1 (Secure Hash Algorithm), de 1993, pour les fonctions de hachage. Cette technique permet de réaliser une empreinte de 160 bits.

La non-répudiation

Les services de non-répudiation consistent à empêcher le démenti qu'une information a été reçue par une station qui l'a réclamée. Ce service permet de donner des preuves, comme on peut le faire par télex. De manière équivalente, on peut retrouver la trace d'un appel téléphonique, de telle sorte que le récepteur de l'appel ne puisse répudier cet appel.

La fonction de non-répudiation peut s'effectuer à l'aide d'une signature à clé privée ou publique ou par un tiers de confiance qui peut certifier que la communication a bien eu lieu.

Caractéristiques des algorithmes de sécurité

Cette section présente quelques caractéristiques des algorithmes de sécurité, en commençant par les algorithmes de chiffrement, puis analyse leurs performances temporelles.

Les algorithmes de chiffrement

Les algorithmes de chiffrement les plus classiques sont DES et 3DES. Le nouveau standard AES est intégré dans cette partie car il est attendu comme le remplaçant des algorithmes précités.

Nombre de rounds

La plupart des algorithmes de chiffrement par clés symétriques utilisent le chiffrement en utilisant deux transformations : une substitution et une permutation. Un « round » est complété lorsque ces deux transformations sont effectuées une fois.

En règle générale, on considère que plus le nombre de round est élevé, plus la sécurité apportée est grande. En revanche, plus le nombre de round est élevé, plus on consomme de ressources, si bien que les temps de réponse peuvent s'en ressentir. C'est la raison pour laquelle AES est devenu très prisé. Cet algorithme ne demande pas un effort de calcul aussi important que les algorithmes DES, tout en étant plus sûr.

Le nombre de round n'est cependant pas toujours le critère le plus important. Certains algorithmes sont beaucoup plus puissants que d'autres, tout en ayant beaucoup moins de rounds.

DES et 3DES

DES applique 16 rounds. 3DES, qui est une succession de trois algorithmes DES, applique $3 \times 16 = 48$ rounds.

Algorithme	Nombre de round
DES	16 rounds
3DES	48 rounds

AES

AES applique 10 rounds avec une clé de 128 bits, 12 rounds avec une clé de 192 bits et 14 rounds avec une clé de 256 bits.

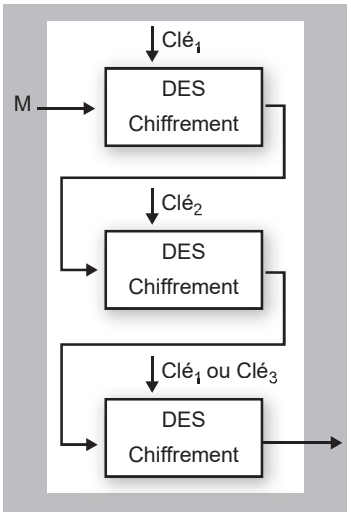
Algorithme	Nombre de round
AES	10 rounds pour une clé de 128 bits 12 rounds pour une clé de 192 bits 14 rounds pour une clé de 256 bits

Longueur de la clé

En règle générale, plus la clé est longue, plus l'algorithme est résistant à une attaque exhaustive. La force d'un algorithme symétrique, dédié à la confidentialité ou à l'intégrité, peut être mesurée en terme de *work factor* associé à une attaque « exhaustive », ou attaque de force brute. Cette attaque consiste à tester toutes les clés possibles. Si une clé a une longueur de n bits, il existe 2^n possibilités de clés différentes, et il faudrait en moyenne 2^{n-1} essais pour trouver la valeur de la bonne clé. On considère que plus la clé est grande, plus la probabilité de la trouver est faible et donc plus grande est la sécurité. La sécurité correspond ici à la résistance de l'algorithme symétrique à une attaque exhaustive.

DES et 3DES

Figure 38.4
Principe de fonctionnement de 3DES



DES utilise une clé de 56 bits. 3DES, qui est illustré à la figure 38.4, étant une succession de trois DES, si K1 = K3, 3DES utilise une clé de $2 \times 56 = 112$ bits. Si K1 ? K3, c'est une clé de $3 \times 56 = 168$ bits.

Algorithme	Longueur de la clé secrète
DES	56 bits
3DES	112 ou 168 bits

AES

L'algorithme AES peut utiliser une clé sécrète de 128, 192 ou 256 bits.

Algorithme	Longueur de la clé secrète
AES	128, 192 ou 256 bits

La performance temporelle

Le temps de traitement de l'algorithme de chiffrement influe sur les caractéristiques temporelles du flux à sécuriser. Un algorithme lent ne permet pas de chiffrer le flux d'une application temps réel sur certaines machines, alors même qu'il peut apporter davantage de sécurité qu'un algorithme plus rapide mais moins sûr. On peut utiliser le nombre de round de l'algorithme pour lui associer un facteur de performance temporelle. Si l'on considère une confidentialité utilisant DES et 3DES, nous pouvons décrire le niveau de sécurité et les performances comme indiqué au tableau 38.1.

Nombre de round	Longueur de la clé secrète	Niveau de sécurité	Performance (rapidité pour QoS)
16 (DES)	56 bits	★	★★
48 (3DES)	112 bits (2 clés)	★★	★
48 (3DES)	168 bits (3 clés)	★★★	★

TABLEAU 38.1 • Performance des algorithmes DES

Pour une confidentialité utilisant uniquement l'AES, à la fois plus sûr et plus rapide que le DES, nous obtenons les performances décrites au tableau 38.2.

Nombre de round	Longueur de la clé secrète	Niveau de sécurité	Performance (rapidité pour QoS)
10	128 bits	★	★★★
12	192 bits	★★	★★
14	256 bits	★★★	★

TABLEAU 38.2 • Performance de l'algorithme AES

Si nous tentons de classifier les niveaux de sécurité des algorithmes DES, 3DES et AES, nous obtenons les valeurs décrites au tableau 38.3.

Nombre de round	Niveau de sécurité	Performance (rapidité pour QoS)
DES	★	★★
3DES (112)	★★	★
3DES (168)	★★★	★
AES (128)	★★★★	★★★★★
AES (192)	★★★★★	★★★★
AES (256)	★★★★★★	★★★

TABLEAU 38.3 • Classification des niveaux de sécurité des algorithmes DES et AES

Des compromis sont nécessaires pour décider du niveau de confidentialité à apporter et de la rapidité de l'algorithme à utiliser. Ce choix peut dépendre de la puissance des machines qui chiffrent et déchiffrent.

Les algorithmes d'authenticité

Les services d'intégrité des données et d'authentification de leur origine sont deux services inséparables. C'est pourquoi on les rassemble sous le terme d'authenticité. La solution la plus classique est d'utiliser un chiffrement permettant à la fois de vérifier l'intégrité et de chiffrer les données. Par exemple, l'authenticité apportée par IPsec s'effectue par l'ajout d'un code d'authentification de message, ou HMAC (Hashing Message Authentication Code).

Dans le cas d'IPsec, qui est le protocole de sécurité le plus utilisé avec SLL, les algorithmes que doit fournir par défaut toute implémentation sont HMAC-MD5 et HMAC-SHA-1. La sécurité fournie par l'algorithme HMAC est directement liée à l'algorithme de hachage sous-jacent, tels MD5 pour HMAC-MD5, SHA-1 pour HMAC-SHA-1, etc.

Dans SHA-1, la fonction de compression demande 4 rounds, chacun constitué de 20 étapes, soit 80 étapes. Dans MD5, la fonction de compression demande 4 rounds, chacun constitué de 16 étapes, soit 64 étapes. On peut dire que pour le chiffrement d'un texte donné, SHA-1 est un peu plus lent que MD5. Chacune de ces fonctions prend en entrée une clé secrète en plus des données auxquelles on souhaite appliquer le hachage. La taille de ces clés est différente selon l'algorithme utilisé.

Pour établir un niveau entre ces différents algorithmes, on peut se fonder sur la force de chacun. La force de l'algorithme est mesurée en terme de facteur de travail associé à une force d'attaque brute, donc liée à la longueur de la clé secrète. Les attaques par force brute sont plus difficiles si SHA-1 est utilisé. Pour cela, SHA-1 est considéré comme étant plus sûr que MD5.

Les niveaux de service d'authenticité

Les niveaux d'authenticité peuvent être classés en fonction de la longueur de la clé secrète et des performances de l'algorithme. Cette classification est récapitulée au tableau 38.4.

Algorithme	Longueur de la clé secrète	Niveau d'authenticité	Performance (rapidité pour QoS)
MD5	128 bits	★	★★
SHA-1	160 bits	★★	★

TABLEAU 38.4 • Niveaux des services d'authenticité

SHA-1 est considéré comme plus sûr que MD5, mais le choix de MD5 au lieu de SHA-1 dans un HMAC peut se justifier pour la performance.

Protection contre le rejeu

Malgré le chiffrement, il est possible d'effectuer une attaque par rejeu, c'est-à-dire en rejouant une séquence de paquets à laquelle l'attaquant ne comprend rien mais qui correspond à ce que le serveur attend. En d'autres termes, il est possible pour un

attaquant de recopier des trames pendant une authentification et de réutiliser les mêmes trames ultérieurement.

La protection contre le rejeu repose sur un numéro de séquence. Elle se présente souvent sous forme d'option : l'utilisateur peut, sur demande explicite, bénéficier d'une protection contre le rejeu en numérotant ses paquets dans le champ Chiffrer du paquet de telle sorte qu'une recopie ne puisse être utilisée puisque le récepteur attend un numéro de série parfaitement déterminé entre l'émetteur et le récepteur.

Les algorithmes d'authentification

Comme nous l'avons vu à plusieurs reprises, l'authentification est une fonction de sécurité essentielle. Le protocole d'authentification utilisé le plus souvent provient de la norme IEEE 802.1x. Cette norme est très générale et s'applique aussi bien aux réseaux terrestres qu'aux réseaux hertziens.

Nous détaillons ici la procédure IEEE 802.1x mais renvoyons au chapitre suivant pour la description des algorithmes les plus utilisés, qui appartiennent au monde IP.

IEEE 802.1x s'appuie essentiellement sur le protocole EAP, que nous examinons beaucoup plus en détail au chapitre 39. EAP est un protocole d'authentification général, qui supporte de multiples méthodes d'authentification, telles que Kerberos, TLS, MS-Chap, SIM, etc. De nouveaux mécanismes d'authentification peuvent ainsi être définis au-dessus d'EAP. Seul le champ type du paquet EAP, codé sur un octet, limite le nombre de mécanismes d'authentification. Le standard EAP a été conçu comme protocole générique pour le transport du trafic des protocoles d'authentification. Il est construit autour d'un modèle de communication demandant un défi (challenge), auquel une réponse doit être apportée pour qu'il y ait authentification.

Comme illustré à la figure 38.5, quatre types de messages EAP permettent de réaliser l'authentification d'un client sur un serveur :

- EAP REQUEST : demande d'authentification ;
- EAP RESPONSE : réponse à une requête d'authentification ;
- EAP SUCCESS : pour indiquer le succès de l'authentification ;
- EAP FAILURE : pour informer le client du résultat négatif de l'authentification.

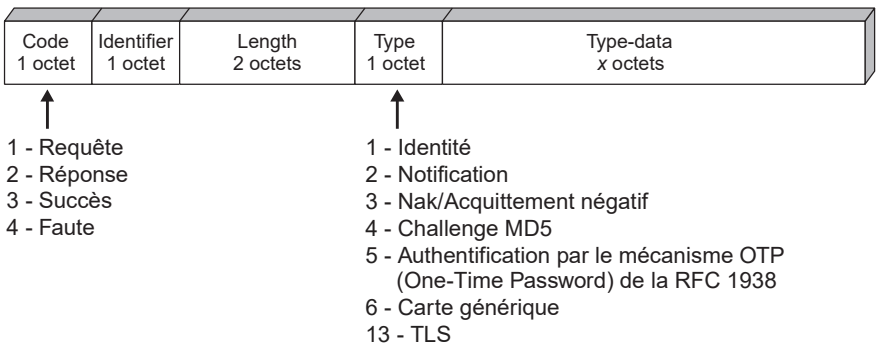


Figure 38.5

Format d'un paquet EAP

La norme IEEE 802.1x met en présence un supplicant, que l'on appelle client, un authenticator, qui est un contrôleur de communication, et un serveur d'authentification. Le client est un équipement terminal relié par une liaison filaire ou sans fil. Le contrôleur est un équipement intermédiaire qui peut être intégré dans un routeur ou dans un commutateur. Il peut aussi s'agir d'un contrôleur Wi-Fi ou même d'un point d'accès Wi-Fi compatible avec la norme IEEE 802.11. Ces points d'accès compatibles sont généralement haut de gamme et réservés au monde professionnel. Le contrôleur est un organe qui possède un port contrôlé, lequel peut être ouvert ou fermé en fonction de la réussite ou non de l'authentification. Le serveur d'authentification est le plus souvent de type RADIUS *(voir le chapitre 39).*

La figure 38.6 illustre cette configuration.

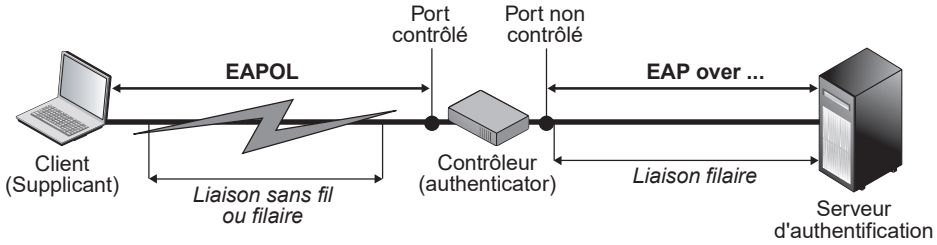


Figure 38.6

Configuration d'une authentification IEEE 802.1x

IEEE 802.1x utilise le protocole EAP pour mettre en communication le client et le serveur d'authentification *via* le contrôleur. Le protocole EAP est encapsulé dans une trame émise sur la connexion avec ou sans fil entre le client et le contrôleur et adaptée à la communication entre le contrôleur et le serveur d'authentification.

L'authentification EAP se déroule de la manière suivante :

1. Lorsque la phase d'établissement de la liaison est terminée, le serveur d'authentification envoie une demande d'identité.
2. Le client envoie un paquet EAP RESPONSE dans lequel il fournit son identité et les méthodes d'authentification qu'il supporte. La phase d'authentification débute à cet instant.
3. Le serveur envoie un défi au client.
4. Le client y répond par un message EAP RESPONSE, dans lequel il envoie le défi chiffré avec sa clé secrète.
5. Le serveur met fin à la phase d'authentification par l'intermédiaire d'un paquet de succès ou d'échec.

Si la phase d'authentification s'est bien déroulée, le serveur d'authentification peut transmettre une clé de chiffrement au client, lequel l'utilise pour chiffrer les données émises. Cette dernière phase est optionnelle pour le protocole EAP, car elle dépend du protocole d'authentification utilisé.

Le protocole EAP étant extensible, tout mécanisme d'authentification peut être encapsulé à l'intérieur des messages EAP, comme l'illustre la figure 38.7. Au niveau supérieur de la figure se trouvent les méthodes d'authentification, comme TLS, MS-Chap, SIM,

etc. Vient ensuite le niveau d'encapsulation de la méthode d'authentification de la trame EAP. La trame EAP elle-même est encapsulée dans une trame de transport. Cette encapsulation peut s'effectuer soit dans une trame EAP over RADIUS, soit dans une trame EAPoL (EAP over LAN), qui est utilisée dans les réseaux locaux, en particulier les réseaux locaux sans fil de type Wi-Fi.

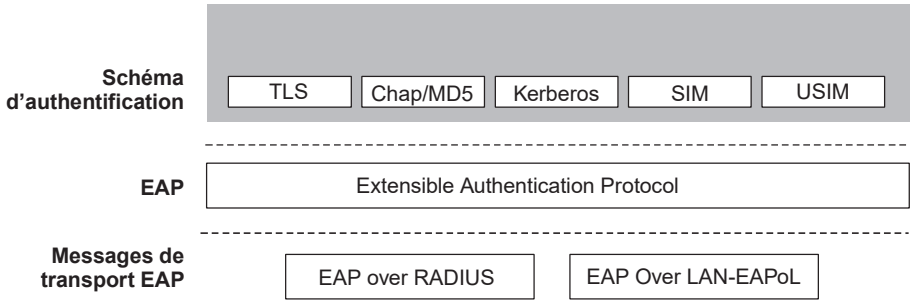


Figure 38.7

Architecture d'EAP

Un autre atout d'EAP est qu'il est conçu pour fonctionner au-dessus de la couche liaison. Il ne nécessite donc pas le niveau IP mais inclut son propre support pour la livraison et la retransmission. Développé pour être utilisé avec PPP, c'est un protocole générique qui supporte la majorité des normes de niveau liaison.

Autres mécanismes d'authentification

D'autres mécanismes d'authentification, plus simples dans leur conception et liés à des environnements différents, sont couramment utilisés. Les sections suivantes présentent brièvement les deux plus connus, OTP et Kerberos. OTP (One-Time Password) introduit un changement du mot de passe à chaque tentative d'authentification. C'est pourquoi on l'appelle mécanisme à mot de passe valable une seule fois. Kerberos est fondé sur une architecture client serveur à chiffrement symétrique.

OTP (One-Time Password)

Fondé sur un algorithme de hachage répété récursivement afin de générer des mots de passe à usage unique, OTP a fait l'objet de plusieurs tentatives de sécurisation par carte à puce.

L'algorithme OTP fonctionne de la façon suivante :

1. Dans le cas des mots de passe défi/réponse, le serveur envoie un défi aléatoire (random challenge) au client.
2. Le client manipule ce défi en introduisant son secret et envoie la réponse au serveur.
3. Le serveur effectue le même travail et compare les résultats.

L'inconvénient de ce mécanisme est qu'il est possible pour un attaquant de rejouer le même mot de passe une fois que celui-ci a été intercepté. Pour y remédier, on utilise une liste de mots de passe, chacun d'eux n'étant valable que pour une tentative d'authentification. Une telle liste est constituée de mots de passe générés et utilisés séquentiellement

par le client à chaque tentative d'authentification. Les mots de passe étant indépendants, il est impossible de prévoir un mot de passe à partir de ceux utilisés précédemment.

La procédure S/KEY

Dans la procédure S/KEY, qui est un cas particulier de la méthode OTN, une liste de mots de passe à usage unique de 64 bits de longueur est générée par le serveur à partir du mot de passe de l'utilisateur. Cela permet au client d'utiliser le même mot de passe sur des machines différentes. La taille des mots de passe OTP est un bon compromis entre sécurité et facilité d'emploi pour l'utilisateur.

Les 8 octets du mot de passe de l'utilisateur sont concaténés avec une séquence aléatoire, ou *seed*. Une fonction de hachage MD4 est appliquée au résultat de la concaténation, puis le résultat obtenu est réduit à 8 octets par un XOR des deux moitiés. Ce résultat, appelé *s*, est fourni en entrée de l'étape suivante.

Le premier mot de passe à usage unique est produit en exécutant *n* fois la fonction de hachage sur *s*. Le mot de passe OTP de rang *i* (p_i) est produit en appliquant la fonction de hachage $n - i$ fois. Les deux équations suivantes indiquent le calcul du premier mot de passe (exécution de *n* fois la fonction de hachage sur *s)* et du i-ème mot de passe (exécution de *n-i* fois la fonction de hachage sur *s)* :

$$p_0 = f_n(s)$$

$$p_i = f_{n-i}(s)$$

Un attaquant qui a surveillé l'utilisation du mot de passe p_i n'est pas en mesure de générer le prochain mot de passe de la séquence (p_{i+1}) puisque la fonction de hachage est irréversible.

Quand un client tente d'être authentifié, la séquence d'octets aléatoires et la valeur courante de *i* sont passées au client. Le client retourne le prochain mot de passe. Le serveur sauvegarde dans un premier temps une copie de ce mot de passe puis lui applique la fonction de hachage :

$$p_i = f(f_{n-i-1}(s)) = f(p_{i+1})$$

Si l'égalité ci-dessus n'est pas vérifiée, la requête échoue. Dans le cas contraire, le fichier de mots de passe est mis à jour avec la copie du mot de passe OTP qui a été sauvegardé. Cette mise à jour avance la séquence de mots de passe.

Ce mécanisme empêche les attaques par rejeu mais n'a malheureusement aucun effet sur les attaques actives.

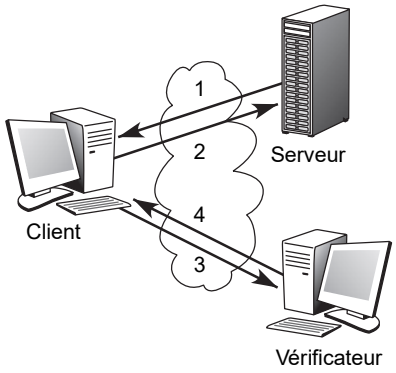
Kerberos

Kerberos est un algorithme d'authentification fondé sur une cryptographie. L'utilisation de cet algorithme ne permet pas à une personne qui écoute le dialogue d'un client à l'insu de ce dernier de se faire passer pour lui plus tard. Ce système permet à un processus client travaillant pour un utilisateur donné de prouver son identité à un serveur sans avoir à envoyer de données dans le réseau.

Kerberos a été développé à partir des années 1980 et en est aujourd'hui à la version v5, qui peut être considérée comme le standard Kerberos. L'idée sous-jacente est que le processus client doit prouver qu'il possède la clé de chiffrement qui est connue des seuls utilisateurs de base et du serveur.

La suite simplifiée des requêtes qui vont être échangées est illustrée à la figure 38.8.

Figure 38.8
Suite simplifiée des requêtes Kerberos



Le client et le processus client ne connaissent pas la clé de chiffrement de base utilisée pour l'authentification. Quand un client doit répondre à une demande d'authentification, il se met en contact avec le serveur pour générer une nouvelle clé de chiffrement et se la faire envoyer de façon sécurisée. Cette nouvelle clé est appelée clé de session. Le serveur utilise un ticket Kerberos pour envoyer la clé de session au vérificateur par l'intermédiaire du client.

Le ticket Kerberos est un certificat provenant, comme nous venons de le voir, du serveur d'authentification. Il est chiffré avec la clé de base que ne connaît pas le client. Le ticket contient des informations, parmi lesquelles la clé de session qui est utilisée pour l'authentification, le nom de l'utilisateur de base et un temps d'expiration après lequel la clé de session n'est plus valide. Puisque le ticket est chiffré avec la clé de base, connue seulement du serveur et du vérificateur, il est impossible au client de modifier le ticket sans que cela passe inaperçu.

À la figure 38.8, la demande 1 de vérification génère la réponse 2, qui délivre un ticket de vérification et la clé de session. Les messages 3 et 4 permettent l'authentification en prouvant que le client connaît la clé de session incluse dans le ticket. À la réception du ticket Kerberos, le vérificateur le déchiffre, extrait la clé de session et utilise celle-ci pour déchiffrer le nom du serveur. Si la même clé a été utilisée pour chiffrer le nom du serveur et le déchiffrer, la vérification de la zone de détection d'erreur, ou checksum, donne un résultat positif.

Cette solution peut paraître insuffisante, puisqu'un utilisateur malveillant peut intercepter les requêtes et les rejouer à la place du vérificateur. Pour cette raison, le vérificateur doit s'assurer que le temporisateur d'utilisation du ticket n'est pas dépassé. À ce stade, il est supposé que l'identité du client a été authentifiée par le vérificateur. Le client peut aussi demander à vérifier l'identité du serveur. Une authentification mutuelle est alors effectuée.

Exemples d'environnements de sécurité

Un environnement de sécurité fait référence à un ensemble d'équipements qui peuvent se faire confiance grâce à des mécanismes de sécurité qui leur permettent d'entrer dans le même cercle de confiance. Les sections suivantes présentent quelques exemples de mécanismes permettant d'introduire des éléments de sécurité.

PGP (Pretty Good Privacy)

PGP est un algorithme permettant de sécuriser la messagerie électronique en lui apportant authentification et confidentialité. PGP permet le chiffrement et la signature de documents par le biais d'un chiffrement symétrique avec IDEA ou DES, d'une fonction de hachage MD5 ou SHA-1 et d'une clé RSA.

Le texte à envoyer est chiffré par l'algorithme IDEA. La signature utilise MD5. L'algorithme RSA est utilisé pour l'échange de la clé privée nécessaire à IDEA. L'authentification par signature implémente une fonction de hachage SHA-1, et la confidentialité un des nombreux algorithmes disponibles, comme IDEA, 3DES, Diffie-Helman, RSA, etc.

L'infrastructure PKI

La distribution sécurisée de clés publiques est une question cruciale pour un système global sécurisé, à laquelle l'infrastructure PKI (Public Key Infrastructure) offre une solution.

Les principales fonctions réalisées par une infrastructure PKI pour la gestion des certificats sont les suivantes :

- Enregistrement de demandes et vérification des critères pour l'attribution d'un certificat. L'identité du demandeur est vérifiée ainsi que le fait qu'il soit bien en possession de la clé privée associée.
- Création des certificats.
- Diffusion des certificats entraînant la publication des clés publiques.
- Archivage des certificats pour assurer la sécurité et la pérennité.
- Renouvellement des certificats en fin de période de validité.
- Suspension des certificats : cette fonction peut être utile si le propriétaire estime ne pas avoir besoin temporairement de son certificat. Cependant, n'étant pas aisée à mettre en œuvre, elle est essentiellement administrative, et il n'existe pas de standard d'implémentation.
- Révocation de certificats pour péremption, perte, vol ou compromission de clés.
- Création et publication des listes de révocation des certificats. Il y a révocation du certificat en cas de fin de validité ou de clé privée divulguée, perdue ou compromise. Il n'existe aucun protocole standard qui permette d'effectuer une révocation automatiquement. Il faut donc recourir à des moyens administratifs. Ceux-ci doivent être implémentés avec un maximum de sécurité, le demandeur de la révocation devant en particulier prouver qu'il est bien le propriétaire de la clé publique ou privée devenue inutilisable. Les listes de révocation doivent, d'une part, être protégées afin d'éviter toute corruption et, d'autre part, être accessibles en permanence et le plus à jour possible. Pour un fonctionnement correct, les listes de révocation nécessitent une synchronisation des horloges de tous les acteurs concernés.
- Délégation de pouvoir à d'autres entités reconnues de confiance. Toute communauté de personnes peut créer sa propre infrastructure PKI. Dans ce cas, une étude de faisabilité est nécessaire en s'appuyant sur de nombreux critères.

Un critère important lors du déploiement d'une PKI est le format des certificats numériques utilisés. Le format le plus largement accepté est le X.509 de l'UIT-T. En plus d'une clé publique, un certificat contient généralement un nom, une adresse et d'autres informations décrivant le porteur de la clé secrète.

Tous les certificats sont signés par la banque de données qui enregistre les clés publiques des membres de la communauté. Pour devenir membre de la communauté, un abonné doit réaliser deux choses :

- Fournir au service d'annuaire une clé publique et des informations d'identification, de telle sorte que les autres personnes soient capables de vérifier la signature de son certificat.
- Obtenir la clé publique du service d'annuaire de façon à permettre une vérification de la signature des autres personnes.

Un certificat étant signé, il est non falsifiable. Son authenticité ne dépend pas du canal par lequel il a été reçu mais est intrinsèque.

Une autorité de certification (AC) émet, gère et révoque les certificats. La clé publique du certificat de l'AC doit être reconnue de confiance par tous les utilisateurs finals. Les certificats émis aux utilisateurs finals sont appelés certificats utilisateur (end-user certificates), et ceux émis pour validation entre les différents AC certificats d'AC (CA-certificates).

Une seule autorité de certification pour le monde entier ne serait pas appropriée. Il est nécessaire que l'architecture PKI soit distribuée, les AC étant autorisées à certifier d'autres AC. L'AC peut déléguer son autorité à une autorité subordonnée en émettant un certificat d'AC, créant ainsi une hiérarchie de certificats. La séquence ordonnée de certificats, de la dernière branche à la racine, est appelée chemin ou chaîne de certification. Chaque certificat contient le nom de l'émetteur du certificat, c'est-à-dire le nom du certificat directement supérieur dans la chaîne. En règle générale, il peut y avoir un nombre arbitraire d'AC sur le chemin entre deux utilisateurs. Pour obtenir la clé publique de son correspondant, un utilisateur doit vérifier le certificat de chaque AC. Ce procédé est appelé validation du chemin de certification.

Quand plusieurs AC sont utilisées, la manière dont les AC sont organisées est très importante pour construire l'architecture PKI. Certaines PKI utilisent un modèle hiérarchique, appelé hiérarchie générale, où chaque AC certifie ses pères et ses fils. D'autres PKI utilisent une variante de la hiérarchie générale dans laquelle les AC certifient seulement leurs fils et l'AC racine dans tous les chemins de certification.

Dans une architecture « top-down », tous les utilisateurs doivent utiliser la plus haute AC comme racine. Cela nécessite que tous les utilisateurs obtiennent une copie de la clé publique de l'AC la plus haute avant d'utiliser la PKI. Tous les utilisateurs doivent pleinement avoir confiance dans l'AC racine, ce qui la rend impraticable pour une PKI globale. La certification croisée (cross-certification) aide à réduire la longueur du chemin, au risque d'en compliquer la découverte. La figure 38.9 illustre un exemple de chemin de certification.

Dans l'optique d'une communication extérieure à l'entreprise, l'interopérabilité des PKI est essentielle. Les principaux efforts de normalisation en ce sens émanent des laboratoires RSA, avec les normes PKCS (Public-Key Cryptography Standards). Ces normes font office de standard et sont unanimement adoptées, notamment pour la cryptographie et l'échange de clés. Parallèlement, l'IETF produit des normes plus générales, avec les RFC

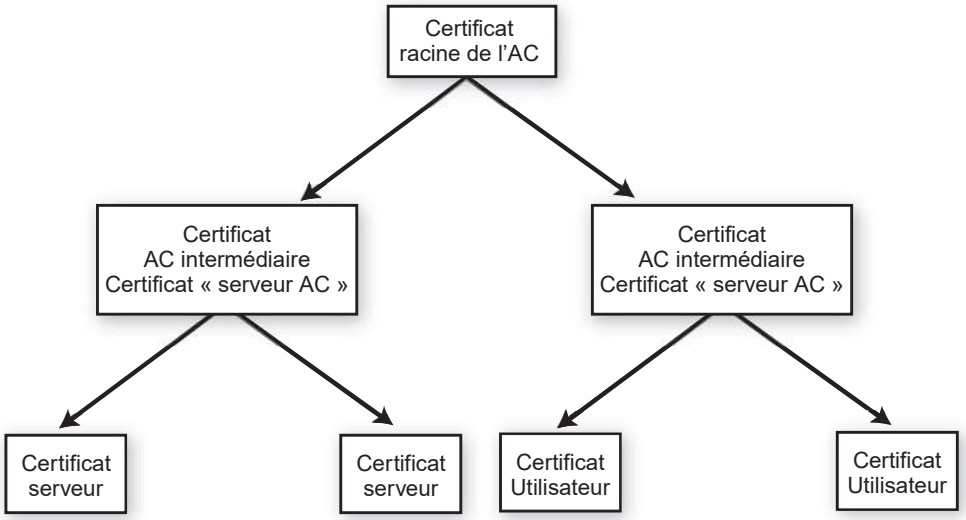


Figure 38.9

Exemple de chemin de certification

PKIX (Public Key Cryptography eXtension). Cependant, certains aspects demeurent encore insuffisamment normalisés, comme les politiques et les pratiques de certification ou les paramètres des certificats.

PKCS (Public-Key Cryptography Standards)

PKCS est un ensemble de standards pour la mise en place des IGC (infrastructure de gestion des clés). Coordonnés par RSA, ces standards définissent les formats des éléments de cryptographie :

- PKCS#1, RSA Cryptography Specifications Version 2, RFC 2437 ;
- PKCS#2, inclus dans PKCS#1 ;
- PKCS#3, Diffie-Hellman Key Agreement Standard Version 1.4 ;
- PKCS#4, inclus dans PKCS#1 ;
- PKCS#5, Password-Based Cryptography Standard Version 2 ;
- PKCS#6, Extended-Certificate Syntax Standard Version 1.5 ;
- PKCS#7, Cryptographic Message Syntax Standard Version 1.5, RFC 2315 ;
- PKCS#8, Private-Key Information Syntax Standard Version 1.2 ;
- PKCS#9, Selected Attribute Types Version 2.0 ;
- PKCS#10, Certification Request Syntax Version 1.7 ou CSR (Certificate Signing Request), RFC 2314 ;
- PKCS#11, Cryptographic Token Interface Standard Version 2.10 ;
- PKCS#12, Personal Information Exchange Syntax Standard Version 1.0 ;

- PKCS#13, Elliptic Curve Cryptography Standard Version 1.0 ;
- PKCS#14, Pseudorandom Number Generation Standard Version 1.0 ;
- PKCS#15, Cryptographic Token Information Format Standard Version 1.1.

Les virus

Les virus sont des programmes, généralement écrits en langage machine, susceptibles de s'introduire dans un ordinateur et de s'y exécuter. L'exécution peut produire de nombreux effets, allant du blocage d'une fonction à la destruction des ressources de l'ordinateur, comme l'effacement de la mémoire ou du disque dur, en passant par l'émission de messages incontrôlés.

Les logiciels antivirus ont pour fonction de détecter la présence de virus sur une machine et de les détruire. Cependant, comme nous allons le voir, certains virus sont résistants, et les logiciels antivirus peuvent avoir du mal à les détecter.

On dénombre un très grand nombre de techniques de virus, notamment les suivantes :

- *Boot sector virus,* ou virus travaillant sur le programme de démarrage. Ce programme se met en route au moment de la mise en marche de la station terminale. Le virus se trouve sur le disque dur et peut se dupliquer sur les disquettes ou les CD. Suivant son origine, le virus bloque un certain nombre de fonctions, parfois de façon aléatoire afin de ne pas se faire détecter. Il peut aussi empêcher tout démarrage de la machine en ne permettant pas à une instruction importante du programme de démarrage de se dérouler.
- *File infected virus.* Ce sont les plus courants. Ils s'attachent à un programme exécutable particulier et, en s'exécutant, bloquent la mise en route du programme, tout en s'attachant à d'autres programmes.
- *Polymorphic virus.* Le rôle de ces virus est de ne pas se faire détecter, tout en causant un certain nombre d'ennuis à l'utilisateur. Ils se modifient en passant à un autre programme, de telle sorte qu'ils sont parfois très difficiles à détecter puisque non répertoriés dans une forme spécifique à un programme.
- *Stealth virus,* que l'on peut traduire par virus furtifs. Comme les précédents, ils tentent de ne pas se faire détecter facilement tout en occasionnant des dégâts aux programmes auxquels ils s'accrochent. Une des méthodes qu'ils emploient le plus fréquemment consiste à s'incruster dans les programmes en prenant la place de quelques lignes de code de telle sorte que la taille exacte du programme reste inchangée.
- *Encrypted virus.* Ces virus forment une famille très délicate à repérer puisqu'ils sont chiffrés et que les antivirus n'ont pas la possibilité de les déchiffrer pour les détecter. Ces virus doivent pouvoir être déchiffrés pour être mis en œuvre. Ils nécessitent donc un environnement qui leur est adapté. Ils utilisent généralement les techniques de chiffrement utilisées classiquement dans les systèmes d'exploitation qu'ils attaquent.
- *Worms,* ou vers. Ces virus sont de nature différente. Ce sont eux-mêmes des programmes qui transportent des virus. Beaucoup d'attaques sur les messageries s'effectuent en attachant un vers au message. L'utilisateur à qui l'on a fait croire à l'utilité de ce programme l'ouvre et l'exécute. Le virus attaché peut alors commencer à infecter la machine.

- *Trojan horses*, ou chevaux de Troie. Ces virus bien connus sont des programmes qui s'introduisent à l'intérieur de l'ordinateur et donnent des renseignements à l'attaquant externe. Le code du cheval de Troie est généralement encapsulé dans un programme système nécessaire au fonctionnement de l'ordinateur.
- *Time bomb virus.* Ces virus sont liés à l'horloge du système et se déclenchent à une heure déterminée à l'avance.
- *Logical bombs,* ou bombes logiques. Ces virus se déclenchent lorsqu'un certain nombre de conditions logiques sont vérifiées.

Il est de plus en plus difficile de détecter les virus, les pirates essayant de les encapsuler dans des programmes innocents. Les parades à ces attaques sont nombreuses, quoique jamais complètement efficaces. La solution la plus simple consiste à se doter d'un antivirus mis à jour régulièrement.

Conclusion

Pour définir la sécurité, on peut partir de la couche application en disant qu'une application est sécurisée si l'utilisateur qui s'en sert a été identifié et authentifié, si les données transportées n'ont pas été modifiées, si les données n'ont pu être interceptées et si elles ont une valeur juridique.

À partir de cette définition, on peut considérer que la sécurité consiste en cinq types d'opération :

- identification d'un utilisateur ;
- authentification d'un utilisateur ;
- intégrité des données ;
- confidentialité des données ;
- non-répudiation.

Les outils à mettre en œuvre pour assurer ces opérations proviennent de différents horizons et progressent rapidement pour tenter de rattraper le retard sur les attaquants, qui ont toujours une longueur d'avance. Le chapitre suivant examine les algorithmes de sécurité les plus utilisés dans le monde IP.

39

La sécurité dans l'environnement IP

Les échanges de données s'effectuent généralement entre un client et un serveur, mais les applications peer-to-peer, qui vont directement d'un client à un autre client, deviennent de plus en plus courantes. En règle générale, le client se sert d'un identifiant déterminé par un login et un mot de passe mis en clair sur le réseau. Le client est donc identifié, et il obtient l'autorisation d'accéder au serveur grâce à son mot de passe. Cependant, cette solution peut se révéler bien faible face à des pirates. L'absence d'authentification de la provenance des paquets IP rend possible de nombreuses attaques, comme les dénis de service *(denial of service),* c'est-à-dire le refus ou l'impossibilité pour un serveur de fournir l'information demandée.

Il existe de nombreuses familles d'attaques dans le réseau Internet. Ce chapitre commence par donner un certain nombre d'exemples de ces attaques avant d'examiner les parades possibles.

Les attaques par Internet

Les attaques concernent deux grands champs : celles qui visent les équipements terminaux et celles qui visent le réseau Internet lui-même. Elles ne sont pas totalement décorrélées puisque les attaques des machines terminales par Internet utilisent souvent des défauts d'Internet.

Les attaques du réseau Internet lui-même consistent à essayer de dérégler un équipement de routage ou un serveur, comme les serveurs DNS, ou à obstruer les lignes de communication. Les attaques des machines terminales consistent à prendre le contrôle de la machine pour effectuer des opérations non conformes. Très souvent, ces attaques s'effectuent par le biais des logiciels réseau qui se trouvent dans la machine terminale. Cette section explicite quelques attaques parmi les plus classiques.

Les attaques par ICMP

Le protocole ICMP (Internet Control Message Protocol) est utilisé par les routeurs pour transmettre des messages de supervision permettant, par exemple, d'indiquer à un utilisateur la raison d'un problème. Un premier type d'attaque contre un routeur ou un serveur réseau consiste à générer des messages ICMP en grande quantité et à les envoyer vers la machine à attaquer à partir d'un nombre de sites important.

Pour inonder un équipement de réseau, le moyen le plus simple est de lui envoyer des messages de type ping lui demandant de renvoyer une réponse. On peut également inonder un serveur par des messages de contrôle ICMP d'autres types.

Les attaques par TCP

Le protocole TCP travaille avec des numéros de port qui permettent de déterminer une adresse de socket, c'est-à-dire d'un point d'accès au réseau. Cette adresse de socket est formée par la concaténation de l'adresse IP et de l'adresse de port. À chaque application correspond un numéro de port, par exemple 80 pour une application HTTP.

Une attaque par TCP revient à utiliser un point d'accès pour faire autre chose que ce pour quoi il a été défini. En particulier, un pirate peut utiliser un port classique pour entrer dans un ordinateur ou dans le réseau d'une entreprise. La figure 39.1 illustre une telle attaque. L'utilisateur ouvre une connexion TCP sur un port correspondant à l'application qu'il projette de dérouler. Le pirate commence à utiliser le même port en se faisant passer pour l'utilisateur et se fait envoyer les réponses. Éventuellement, le pirate peut prolonger les réponses vers l'utilisateur de telle sorte que celui-ci reçoive bien l'information demandée et ne puisse se douter de quelque chose.

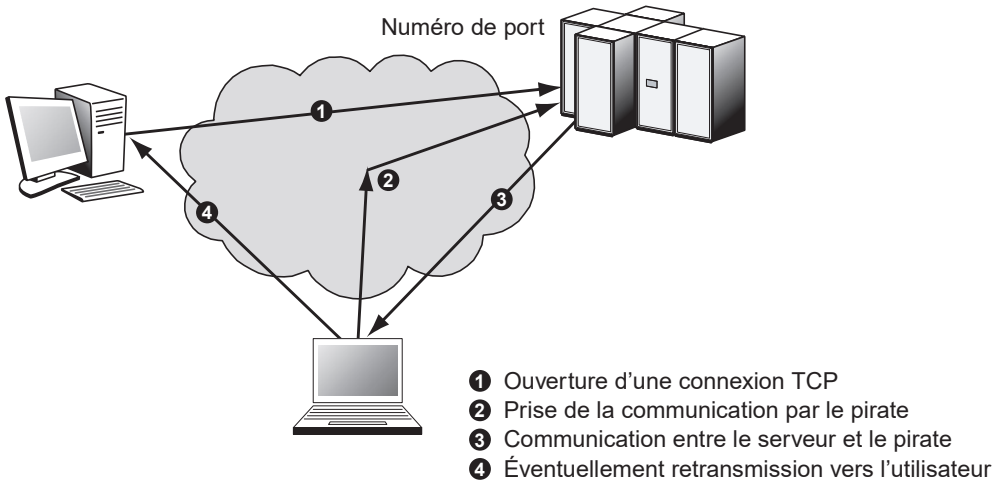


Figure 39.1

Attaque par le protocole TCP

Nous verrons en fin de chapitre comment les pare-feu, ou firewalls, essaient de parer ce genre d'attaque en bloquant certains ports.

Les attaques par cheval de Troie

Dans l'attaque par cheval de Troie, le pirate introduit dans la station terminale un programme qui permet de mémoriser le login et le mot de passe de l'utilisateur. Ces informations sont envoyées à l'extérieur par le biais d'un message vers une boîte aux lettres anonyme.

Diverses techniques peuvent être utilisées pour cela, allant d'un programme qui remplace le gestionnaire de login jusqu'à un programme pirate qui espionne ce qui se passe dans le terminal.

Les attaques par dictionnaire

Beaucoup de mots de passe étant choisis dans le dictionnaire, il est très simple pour un automate de les essayer tous. De nombreuses expériences ont démontré la facilité de cette attaque et ont mesuré que la découverte de la moitié des mots de passe des employés d'une grande entreprise s'effectuait en moins de deux heures.

Une solution simple pour remédier à cette attaque est de complexifier les mots de passe en leur ajoutant des lettres majuscules, des chiffres et des signes comme !, ?, &, etc.

Les autres attaques

Le nombre d'attaques possibles est bien trop grand pour que nous puissions les citer toutes. De plus, de nouvelles procédures d'attaque s'inventent chaque jour.

Les attaques par écoute consistent, pour un pirate, à écouter une ligne de communication et à interpréter les éléments binaires qu'il intercepte. Les attaques par fragmentation utilisent le fait que les informations de supervision se trouvent dans la première partie du paquet à un emplacement parfaitement déterminé. Un pirate peut modifier la valeur du bit de fragmentation, ce qui a pour effet de faire croire que le message se continue alors qu'il aurait dû se terminer. Le pare-feu voit donc arriver une succession de fragments qui suivent les fragments de l'utilisateur sans se douter que ces fragments complémentaires ont été ajoutés par le pirate.

Les algorithmes de routage sont à la base de nombreuses attaques. En effectuant des modifications sur les tables de routage, le pirate peut récupérer de nombreuses informations qui ne lui sont pas destinées ou dérouter les paquets, lesquels, par exemple, vont effectuer des boucles et saturer le réseau.

De la même façon, de nombreuses attaques sont possibles en perturbant un protocole comme ARP (Address Resolution Protocol), soit pour prendre la place d'un utilisateur, soit en captant des données destinées à un autre.

Les parades

Les parades aux attaques sont nombreuses. Elles relèvent autant du comportement humain que de techniques spécifiques. Nous allons examiner les principales : l'authentification, l'intégrité du flux, la non-répudiation, la confidentialité du flux et la confidentialité au niveau de l'application.

L'authentification

Une première parade visant à empêcher qu'un autre terminal que celui prévu ne se connecte ou bien qu'un terminal ne se connecte sur un serveur pirate est offerte par les méthodes d'authentification. L'authentification peut être simple, et ne concerner que l'utilisateur, ou mutuelle, et impliquer à la fois le client et le serveur.

Dans des applications de type Telnet, e-mail ou LDAP, le client s'authentifie avec un mot de passe auprès du serveur pour établir ses droits. Dans une application de commerce électronique (HTTP), il est nécessaire d'authentifier le serveur puis le client, généralement à l'aide d'un mot de passe. Le protocole HTTP ne possédant pas de moyen efficace d'authentification du client, la société Netscape a introduit vers 1995 la notion de cookie, destinée à identifier un flux de requêtes HTTP disjointes.

L'intégrité du flux de données

L'intégrité d'un flux de données demande qu'il ne puisse y avoir une altération des informations transportées. Un pirate pourrait en effet modifier une information pour tromper le récepteur. Il est à noter qu'intégrité ne signifie pas confidentialité. En effet, il est possible que l'information ne soit pas confidentielle et qu'elle puisse être recopiée, sans que cela pose de problème à l'utilisateur. Cependant, l'utilisateur veut que son information arrive intègre au récepteur.

La solution classique à ce genre de problème consiste à utiliser une empreinte. À partir de l'ensemble des éléments binaires dont on souhaite assurer l'intégrité, on calcule une valeur, qui ne peut être modifiée sans que le récepteur s'en rende compte. Les empreintes regroupent les solutions de type empreinte digitale, signature électronique, analyse rétinienne, reconnaissance faciale et, d'une manière générale, tout ce qui permet de signer de façon unique un document. Ces différentes techniques de signature proviennent de techniques d'authentification puisque, sous une signature, se cache une authentification.

Dans les réseaux IP, la pratique de la signature électronique est de plus en plus mise en œuvre pour faciliter le commerce et les transactions financières.

La non-répudiation

La non-répudiation consiste à empêcher l'éventuel refus d'un récepteur d'effectuer une tâche suite à un démenti de réception. Si la valeur juridique d'un fax est reconnue, celle d'un message électronique ne l'est pas encore. Pour qu'elle le soit, il faut un système de non-répudiation. Les parades visant à éviter qu'un utilisateur répudie un message reçu proviennent essentiellement d'une signature unique sur le message et sur son accusé de réception, c'est-à-dire une signature qui ne serait valable qu'une seule fois et serait liée à la transmission du message qui a été répudié. Un système de chiffrement à clés publiques peut être utilisé dans ce contexte.

Une autre solution, qui se développe, consiste à passer par un notaire électronique, qui, par un degré de confiance qui lui est attribué, peut certifier que le message a bien été envoyé et reçu.

Une difficulté importante de la non-répudiation dans une messagerie électronique provient de la vérification que le récepteur en a pris possession et a lu le message.

Il n'existe pas de règle aujourd'hui sur Internet pour envoyer des messages de type lettre recommandée. Le récepteur peut, par exemple, recevoir le message dans sa boîte aux lettres électronique mais ne pas le récupérer. Il peut également recopier le message dans la boîte aux lettres de son terminal et le supprimer sans le lire.

Les techniques de non-répudiation ne sont pas encore vraiment développées dans le monde IP. En effet, cette fonction de sécurité est souvent jugée moins utile que les autres. Cependant, elle est loin d'être absente. En effet, dans le commerce électronique elle est capitale pour qu'un achat ne puisse être décommandé sans certaines conditions déterminées dans le contrat d'achat. Cette fonction serait également utile dans des applications telles que la messagerie électronique, où l'on aimerait être sûr qu'un message est bien arrivé.

Même si la non-répudiation n'est pas implémentée de façon automatique, elle est proposée dans de nombreuses applications qui en ont besoin.

La confidentialité

La confidentialité désigne la capacité de garder une information secrète. Le flux, même s'il est intercepté, ne doit pas pouvoir être interprété. La principale solution permettant d'assurer la confidentialité d'un flux consiste à le chiffrer. Les systèmes de chiffrement ont été présentés au chapitre 38.

Aujourd'hui, étant donné la puissance des machines qui peuvent être mises en jeu pour casser un code, il faut utiliser de très longues clés. Les clés de 40 bits peuvent être percées en quelques secondes et celles de 128 bits en quelques minutes sur une très grosse machine. Une clé RSA de 128 bits a été cassée en quelques heures par un ensemble de machines certes important mais accessible à une entreprise.

Pour casser une clé, il faut récupérer des données chiffrées, parfois en quantité importante, ce qui peut nécessiter plusieurs heures d'écoute, voire plusieurs jours si la ligne est à faible débit. Une solution à ce problème de plus en plus souvent utilisée consiste à changer de clé régulièrement de telle sorte que l'attaquant n'ait jamais assez de données disponibles pour casser la clé.

Dans la réalité, il est plus facile de pirater une clé que d'effectuer son déchiffrement. Une parade pour contrer les pirates réside dans ce cas dans un contrôle d'accès sophistiqué des bases de données de clés.

La confidentialité est aujourd'hui un service fortement utilisé dans le monde IP. IPsec en est un très bon exemple, et nous le détaillons un peu plus loin dans ce chapitre. De nouvelles méthodes, comme le chiffrement quantique, sont à l'étude et pourraient déboucher sur des méthodes encore plus sûres.

La sécurité dans les protocoles

Conçus avant les années 2000, les protocoles du monde IP n'ont pas intégré de fonctions de sécurité. De nombreuses failles de sécurité existent donc, qui sont comblées régulièrement par des RFC spécifiques.

Les attaques sur les protocoles de gestion ou de contrôle peuvent facilement arrêter le fonctionnement d'un réseau. Il suffit, par exemple, de faire croire aux accès que le réseau est saturé ou que les nœuds sont en panne pour que les performances du réseau s'effondrent totalement.

La sécurité dans SNMP

La RFC 2274 définit le modèle USM (User-based Security Model) de sécurité de SNMP, qui offre à la fois une authentification et un service de sécurité.

Les principales attaques dont SNMP peut être l'objet sont les suivantes :

- Modification de l'information : une entité peut altérer un message en transit généré par une entité autorisée pour modifier une opération de type comptabilité, configuration ou opération.
- Mascarade : une entité prend l'identité d'une entité autorisée.
- Modification à l'intérieur d'un flot de messages : SNMP est construit pour gérer un protocole de transport en mode sans connexion. Les messages peuvent être réordonnés d'une façon différente de celle d'origine et détruits ou rejoués d'une autre manière. Par exemple, un message qui redémarre une machine peut être copié puis rejoué ultérieurement.
- Ordre de secret : une entité peut observer les échanges entre un manager et son agent et apprendre les valeurs des objets gérés. Par exemple, l'observation d'un ensemble de commandes capables de modifier un mot de passe permettrait à un utilisateur de modifier le mot de passe et d'attaquer le site.

Le modèle de sécurité USM ne prend pas en compte les deux fonctionnalités suivantes :

- Refus de service : un attaquant interdit l'échange d'informations entre un manager et son agent. Nous avons vu au chapitre 34, consacré à la gestion de réseau, que les échanges d'information de gestion s'effectuaient entre un manager de gestion et ses agents. Si le manager ne reçoit plus les informations du réseau et *vice versa,* les agents ne reçoivent plus les commandes du manager, et le processus de gestion du réseau ne peut plus s'effectuer. On appelle cette attaque un refus de service, puisque le service de gestion refuse de travailler.
- Analyse de trafic : un attaquant observe le type de trafic qui s'effectue entre un manager et son agent. L'analyse permet de détecter les ordres qui sont passés et les remontées d'information. Après analyse du trafic, le pirate peut faire croire au manager que le trafic est totalement différent de ce qu'il est effectivement dans le réseau.

Pour contrer ces différentes attaques, deux fonctions cryptographiques ont été définies dans USM : l'authentification et le chiffrement. Pour les réaliser, le moteur SNMP requiert deux valeurs : une clé privée et une clé d'authentification. Ces valeurs sont des attributs de l'utilisateur qui ne sont pas accessibles par des primitives SNMP.

Deux algorithmes d'authentification sont disponibles : HMAC-MD5-96 et HMAC-SHA-96. L'algorithme HMAC utilise une fonction de hachage sécurisée et une clé secrète pour produire un code d'authentification du message. Ce protocole fortement utilisé dans Internet est décrit en détail dans la RFC 2104.

IPsec (IP sécurisé)

Le monde TCP/IP permet d'interconnecter plusieurs millions d'utilisateurs, lesquels peuvent souhaiter que leur communication reste secrète. Internet transporte de plus un grand nombre de transactions de commerce électronique, pour lesquelles une certaine confidentialité est nécessaire, par exemple pour prendre en charge la transmission de numéros de carte bancaire.

L'idée développée dans les groupes de travail sur la sécurité du commerce électronique dans le monde IP consiste à définir un environnement contenant un ensemble de mécanismes de sécurité. Les mécanismes de sécurité appropriés sont choisis par une association de sécurité *(voir ci-après).* En effet, toutes les communications n'ont pas les mêmes caractéristiques, et leur sécurité ne demande pas les mêmes algorithmes.

Chaque communication se définit par sa propre association de sécurité. Les principaux éléments d'une association de sécurité sont les suivants :

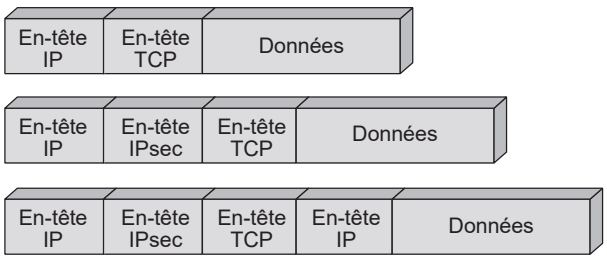
- algorithme d'authentification ou de chiffrement utilisé ;
- clés globales ou spécifiques à prendre en compte ;
- autres paramètres de l'algorithme, comme les données de synchronisation ou les valeurs d'initialisation ;
- durée de validité des clés ou des associations ;
- sensibilité de la protection apportée (secret, top secret, etc.).

La solution IPsec introduit des mécanismes de sécurité au niveau du protocole IP, de telle sorte qu'il y ait indépendance vis-à-vis du protocole de transport. Le rôle de ce protocole de sécurité est de garantir l'intégrité, l'authentification, la confidentialité et la protection contre les techniques rejouant des séquences précédentes. L'utilisation des propriétés d'IPsec est optionnelle dans IPv4 et obligatoire dans IPv6.

Une base de données de sécurité, appelée SAD (Security Association Database), regroupe les caractéristiques des associations par l'intermédiaire de paramètres de la communication. L'utilisation de ces paramètres est définie dans une autre base de données, la SPD (Security Policy Database). Une entrée de la base SPD regroupe les adresses IP de la source et de la destination, ainsi que l'identité de l'utilisateur, le niveau de sécurité requis, l'identification des protocoles de sécurité mis en œuvre, etc.

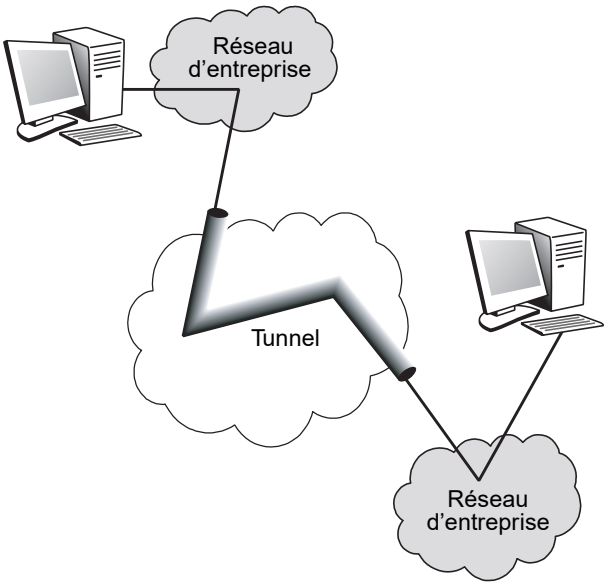
Le format des paquets IPsec est illustré à la figure 39.2. La partie la plus haute de la figure correspond au format d'un paquet IP dans lequel est encapsulé un paquet TCP. La partie du milieu illustre le paquet IPsec. On voit que l'en-tête IPsec vient se mettre entre l'en-tête IP et l'en-tête TCP. La partie basse de la figure montre le format d'un paquet dans un tunnel IPsec. La partie intérieure correspond à un paquet IP encapsulé dans un paquet IPsec de telle sorte que le paquet IP intérieur soit bien protégé.

Figure 39.2
Format des paquets IPsec



Dans un tunnel IPsec, tous les paquets IP d'un flot sont transportés de façon totalement chiffrée. Il est de la sorte impossible de voir les adresses IP ni même les valeurs du champ de supervision du paquet IP encapsulé. La figure 39.3 illustre un tunnel IPsec.

Figure 39.3
Tunnel IPsec



L'en-tête d'authentification

L'en-tête d'authentification est ajouté immédiatement derrière l'en-tête IP standard. À l'intérieur de l'en-tête IP, le champ indiquant le prochain protocole inclus dans le paquet IP (champ Next-Header) prend la valeur 51. Cette valeur précise que les champs IPsec et d'authentification sont mis en œuvre dans le paquet IP. L'en-tête IPsec possède lui-même un champ indiquant le protocole encapsulé dans le paquet IPsec. En d'autres termes, lorsqu'un paquet IP doit être sécurisé par IPsec, il repousse la valeur de l'en-tête suivant, qui était dans le paquet IP, dans le champ en-tête suivant de la zone d'authentification d'IPsec et met la valeur 51 dans l'en-tête de départ.

La figure 39.4 présente le détail l'en-tête d'authentification. Comme indiqué précédemment, cet en-tête commence par la valeur indiquant le protocole transporté. Le champ LG (Length), sur un octet, indique la taille de l'en-tête d'authentification. Vient ensuite une zone réservée, sur 2 octets, qui prend place avant le champ sur 4 octets, donnant un index des paramètres de sécurité, qui décrit le schéma de sécurité adopté pour la communication.

Le champ numéro de séquence, qui contient un numéro de séquence unique, est nécessaire pour éviter les attaques de type rejeu, dans lesquelles le pirate rejoue exactement la même séquence de messages que l'utilisateur par une copie pure et simple. Par exemple, si vous consultez votre compte en banque et qu'un pirate recopie vos messages, même chiffrés, c'est-à-dire sans les comprendre, il peut, à la fin de votre session, rejouer la même succession de messages, qui lui ouvrira les portes de votre compte.

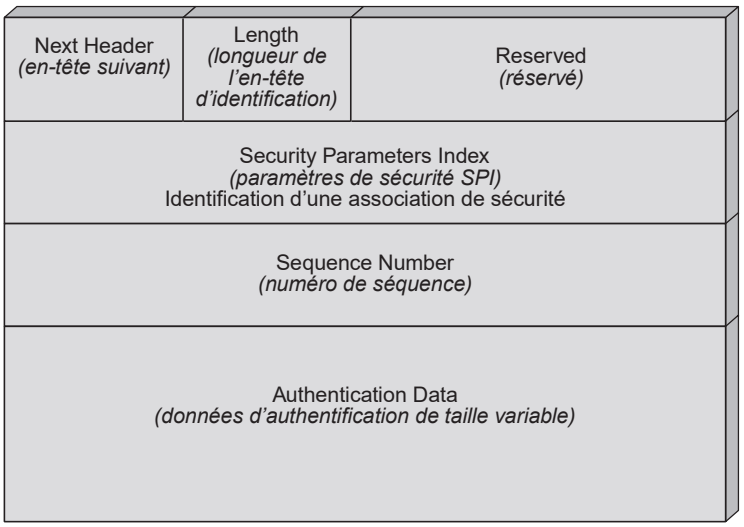


Figure 39.4

Format de l'en-tête d'authentification

L'en-tête d'authentification se termine par les données associées à ce schéma de sécurité. Il transporte le type d'algorithme de sécurité, les clés utilisées, la durée de vie de l'algorithme et des clés, une liste des adresses IP des émetteurs qui peuvent utiliser le schéma de sécurité, etc.

L'en-tête d'encapsulation de sécurité

Pour permettre une confidentialité des données, tout en garantissant une authentification, IPsec utilise une encapsulation dite ESP (Encapsulating Security Payload), c'est-à-dire une encapsulation de la charge utile de façon sécurisée. La valeur 50 est transportée dans le champ en-tête suivant (Next-Header) du paquet IP pour indiquer cette encapsulation ESP.

La figure 39.5 illustre ce processus d'encapsulation. On s'aperçoit que l'encapsulation ESP ajoute trois champs supplémentaires au paquet IPsec : l'en-tête ESP, qui suit l'en-tête IP de départ et porte la valeur 50, le Trailer, ou en-queue, ESP, qui est chiffré avec la charge utile, et le champ d'authentification ESP de taille variable, qui suit la partie chiffrée sans être lui-même chiffré.

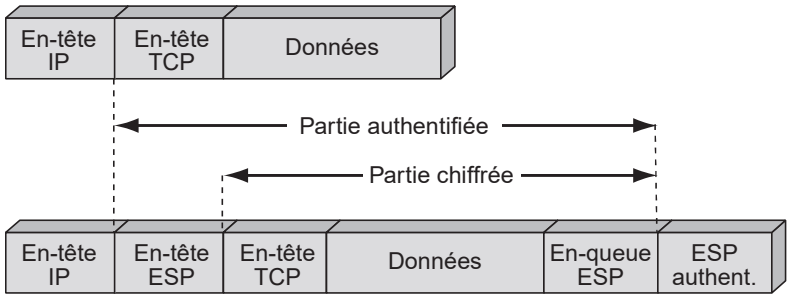


Figure 39.5

Processus d'encapsulation ESP

Le paquet ESP est repris à la figure 39.6 de façon un peu plus détaillée en ce qui concerne les champs internes, à partir du champ ESP d'en-tête.

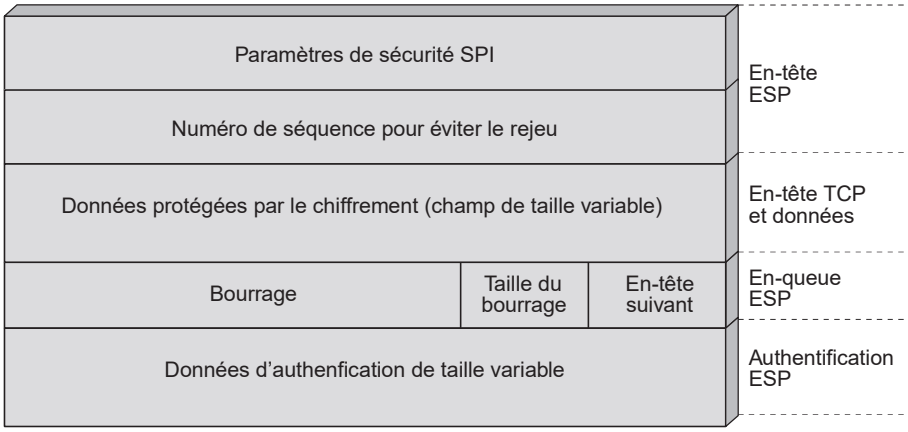


Figure 39.6

Format de l'en-tête ESP

La première partie de l'encapsulation reprend les paramètres SPI (Security Parameter Index) et numéro de séquence que nous avons déjà décrits dans l'en-tête d'authentification. Vient ensuite la partie transportée et chiffrée. L'en-queue ESP comporte une zone de bourrage optionnelle, allant de 0 à 255 octets, puis un champ longueur du bourrage (Length) et la valeur d'un en-tête suivant.

La zone de bourrage a plusieurs raisons d'être. La première provient de l'adoption d'algorithmes de chiffrement, qui exigent la présence d'un nombre de 0 déterminé après la zone chiffrée. La deuxième raison vient de la place de l'en-tête suivant, qui doit être aligné à droite, c'est-à-dire prendre une place en fin d'un mot de 4 octets. La dernière raison est que, pour contrer une attaque, il peut être intéressant d'ajouter de l'information sans signification susceptible de leurrer un pirate.

Les compléments d'IPsec

Dans IPsec, le chiffrement ne s'effectue pas sur l'ensemble des champs, car certains champs, que l'on appelle *mutable,* changent de valeur à la traversée des routeurs, comme le champ TTL (durée de vie). Dans le calcul du champ d'authentification, le processus ne tient pas compte de ces champs mutables.

Les algorithmes de sécurité qui peuvent être utilisés dans le cadre d'IPsec sont déterminés par un certain nombre de RFC :

- Pour l'en-tête d'authentification :
  - HMAC avec MD5 : RFC 2403 ;
  - HMAC avec SHA-1 : RFC 2403.
- Pour l'en-tête ESP :
  - DES en mode CBC : RFC 2405 ;
  - HMAC avec MD5 : RFC 2403 ;
  - HMAC avec SHA-1 : RFC 2404.

La sécurité dans IPv6

Le protocole IPv6 contient les mêmes fonctionnalités qu'IPsec. On peut donc dire qu'il n'existe pas d'équivalent d'IPsec dans le contexte de la nouvelle génération IP.

Les champs de sécurité sont optionnels. Leur existence est détectée par les valeurs 50 et 51 du champ en-tête suivant (Next-Header). Globalement, la sécurité offerte par IPv6 est donc exactement la même que celle offerte par IPsec. Elle est toutefois plus simple à mettre en œuvre puisque le protocole de sécurité est dans le protocole IPv6 lui-même. On peut en déduire que la sécurisation des communications sera beaucoup plus simple avec la nouvelle génération de réseau qui utilisera IPv6.

Cela pose toutefois d'autres problèmes. Si tous les flux sont chiffrés, par exemple, il n'y a plus moyen de reconnaître les numéros de port ou les adresses source et destination, et les applications deviennent transparentes. Toutes les appliances intermédiaires, comme les pare-feu ou les contrôleurs de qualité de service, deviennent inutilisables. L'information sur le type d'application véhiculé ne se trouve qu'à la source ou à la destination, et c'est là qu'il faut venir la rechercher. De nouvelles architectures de contrôle sont donc à prévoir avec l'arrivée d'IPv6, ce qui constitue une raison de repousser cette arrivée dans beaucoup d'entreprises.

SSL (Secure Sockets Layer)

SSL est un logiciel permettant de sécuriser les communications sous HTTP ou FTP. Ce logiciel a été développé par Netscape pour son navigateur et les serveurs Web.

Le rôle de SSL est de chiffrer les messages entre un navigateur et le serveur Web interrogé. Le niveau d'architecture où se place SSL est illustré à la figure 39.7. Il s'agit d'un niveau compris entre TCP et les applicatifs.

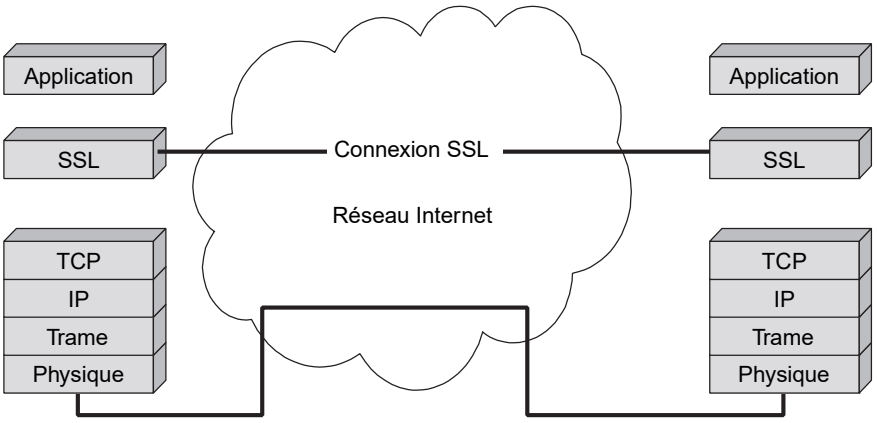


Figure 39.7

Architecture SSL

Les signatures électroniques sont utilisées pour l'authentification des deux extrémités de la communication et l'intégrité des données.

L'initialisation d'une communication SSL commence par un handshake, c'est-à-dire une poignée de main, qui permet l'authentification réciproque grâce à un tiers de confiance. La communication se continue par une négociation du niveau de sécurité à mettre en œuvre et peut se dérouler avec un chiffrement associé au niveau négocié à la phase précédente.

Les inconvénients du protocole SSL proviennent de l'utilisation d'un tiers de confiance et de la nécessité d'ouvrir le port associé à SSL dans les pare-feu. Nous verrons ultérieurement dans ce chapitre la signification exacte de l'expression « ouvrir un port ».

Le protocole SSL a vu son champ d'action dépasser la simple sécurisation d'une communication Web. Il est notamment utilisé dans le commerce électronique pour sécuriser la transmission du numéro de carte de crédit. Un autre protocole, S-HTTP (Secure HTTP), assez semblable à SSL, a été développé pour sécuriser les communications sous HTTP, mais il est beaucoup moins utilisé.

L'authentification de SSL s'appuie sur la cryptographie asymétrique par le biais de certificats à clé publique. Un client peut s'authentifier automatiquement auprès d'un serveur SSL utilisant la paire de clés publiques nécessaire. SSL peut utiliser différents mécanismes de chiffrement. En règle générale, la négociation entre le client et le serveur SSL permet de définir le meilleur algorithme commun. Classiquement, un serveur SSL utilise une clé publique RSA pour établir un ensemble de clés secrètes partagées utilisées avec un algorithme de chiffrement RC4 pour le chiffrement des données et MD5 pour l'intégrité.

Le protocole SSLv3 propose une architecture plus évoluée, qui contient un générateur de clés, des fonctions de hachage et des algorithmes de chiffrement et de gestion de certificats. Cette architecture est représentée à la figure 39.8.

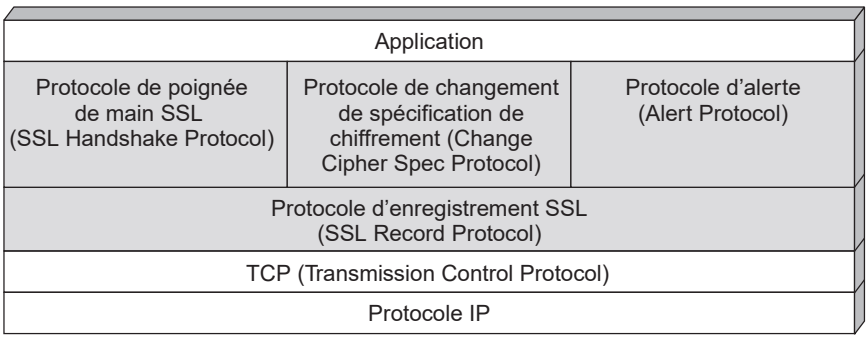


Figure 39.8

Architecture du protocole SSLv3

Le protocole de changement de spécification de chiffrement permet de modifier l'algorithme de chiffrement en cours de communication de sorte à garantir la confidentialité des données transportées. Le protocole d'alerte permet d'envoyer des alertes, accompagnées de leur importance. Ces alertes peuvent être un certificat inconnu, révoqué, expiré, etc. Les alertes de haut niveau entraînent l'arrêt de la communication.

Le protocole Handshake a pour objectif d'authentifier le serveur depuis le client, de négocier la version du protocole, de sélectionner les algorithmes de chiffrement, d'utiliser des techniques de chiffrement à clé publique pour générer et distribuer des clés secrètes et d'établir des connexions SSL chiffrées.

Messages du protocole Handshake

Les messages échangés pour réaliser le protocole Handshake sont les suivants :

- CLIENTHELLO : initialisation de la communication par l'envoi d'un hello du client vers le serveur.
- SERVERHELLO : en retour du message précédent, cette réponse peut contenir un certificat et demander une authentification de la part du client.
- SERVERKEYEXCHANGE : si les certificats ne sont pas pris en charge, ce message permet d'effectuer l'échange de clés publiques.
- SERVERHELLODONE : permet d'indiquer que la partie serveur du message hello est achevée.
- CERTIFICATEREQUEST : requête envoyée par le serveur au client lui demandant de s'authentifier. Le client répond soit avec un message envoyant le certificat, soit avec une alerte indiquant qu'il ne possède pas de certificat.
- CERTIFICATEMESSAGE : message qui envoie le certificat réclamé par le serveur.
- NOCERTFICATE : message d'alerte qui indique que le client ne possède aucun certificat susceptible de correspondre à la demande du serveur.
- CLIENTKEYEXCHANGE : échange de la clé du client avec le serveur.
- FINISHED : message qui conclut le handshake pour indiquer la fin de la mise en place de la communication.

Le protocole d'enregistrement SRP (SSL Record Protocol) n'est qu'une encapsulation des protocoles situés juste au-dessus, comme le protocole Handshake.

Le protocole SSLv3.0 et son successeur TLS1.0 ne présentent que des différences mineures entre eux mais ne sont cependant pas interopérables. La principale différence entre les deux concerne les méthodes de chiffrement puisque TLS n'impose aucune restriction.

Les protocoles d'authentification

Nous commencerons par introduire le protocole PPP et tous ses dérivés, puis nous examinerons l'extension EAP (Extensible Authentication Protocol), qui est devenue le standard de transport des informations d'authentification dans le monde des réseaux IP.

PPP (Point-to-Point Protocol)

PPP a été défini en juillet 1994 dans la RFC 1661. Protocole de niveau trame, il permet de transporter un paquet d'un nœud vers un autre nœud. Bien que conçu initialement pour transporter des paquets IP, il peut prendre en compte d'autres protocoles de contrôle, que nous détaillons également dans ce chapitre.

La structure de la trame PPP est illustrée à la figure 39.9.

Figure 39.9

Structure de la trame PPP

Le champ Protocol, sur 2 octets, identifie le type de paquet inclus dans la trame PPP. Les valeurs de ce champ sont indiquées au tableau 39.1.

Valeur	Protocole encapsulé
0x0021	IP
0xC021	LCP (Link Control Protocol)
0x8021	NCP (Network Control Protocol)
0xC023	PAP (Password Authentication Protocol)
0xC025	LQR (Link Quality Report)
0xC223	CHAP (Challenge Handshake Authentication Protocol)

TABLEAU 39.1 • Valeurs du champ Protocol de la trame PPP

Un diagramme des états du protocole PPP est illustré à la figure 39.10.

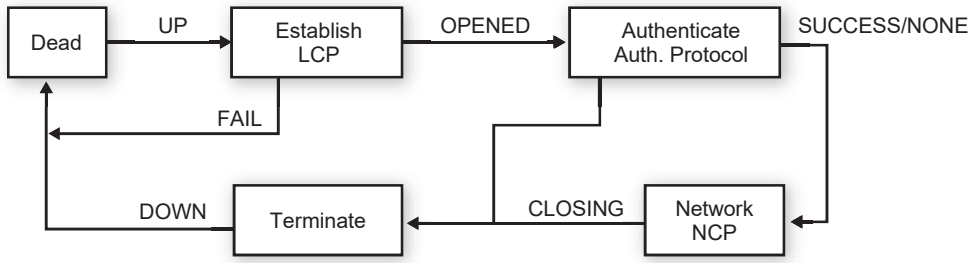


Figure 39.10

Diagramme des états de PPP

La liaison est d'abord mise en place par le protocole LCP (Link Control Protocol). Une fois l'ouverture effectuée, une authentification des extrémités a lieu pour sécuriser la liaison. Les protocoles de la famille CHAP (Challenge Handshake Authentication Protocol) ont été développés dans ce but. Une fois la liaison sécurisée, le protocole de contrôle NCP (Network Control Protocol) prend la suite pour déterminer les protocoles de niveau paquet qui vont utiliser la liaison.

LCP (Link Control Protocol)

Le protocole LCP permet d'ouvrir une liaison PPP et donne les moyens de négocier les options mises en œuvre par PPP, comme la taille des MTU.

Il existe onze types de paquets LCP, identifiés par un code sur 1 octet. Les options sont encodées sous la forme d'un code sur 1 octet, d'un identificateur sur 1 octet et d'une longueur sur 2 octets. Elles sont suivies par les données transportées, dont la longueur est précisée par le champ Length.

Les différents codes qui peuvent être utilisés dans le paquet LCP sont les suivants :

- (1) requête de configuration ;
- (2) accusé (ACK) de configuration ;

- (3) non accusé de configuration (NAK) ;
- (4) requête de terminaison ;
- (6) accusé de terminaison ;
- (7) rejet de code ;
- (8) rejet de protocole ;
- (9) requête d'écho ;
- (10) réponse d'écho ;
- (11) requête d'élimination.

L'identificateur du deuxième octet de LCP définit les options suivantes :

- (1) MRU (Maximum Receive Unit) ;
- (3) protocole d'authentification ;
- (4) protocole de qualité (mesure de la qualité de la ligne utilisée) ;
- (5) nombre magique (détection de boucles ; client et serveur sur le même système hôte) ;
- (7) compression du champ protocole (de 2 octets à 1-PFC) ;
- (8) compression des champs adresse et contrôle de la trame HDLC (ACFC).

NCP (Network Control Protocol)

Le protocole NCP est défini dans la RFC 1332. Il permet de configurer des types de réseaux différents susceptibles d'utiliser la liaison PPP, par exemple IP ou DECnet. Issu de l'architecture réseau de la société DEC, qui a disparu au cours des années 1990, DECnet n'est plus employé aujourd'hui. Dans le cadre des réseaux IP, le protocole utilisé est IPCP (Internet Protocol Control Protocol).

IPCP utilise seulement les sept premiers types de paquets de LCP. Une option de compression peut être utilisée. Dans ce cas, le code indiqué dans le champ protocole des trames PPP est 0x002D. Une adresse IP peut être attribuée par le serveur au client.

PAP (Password Authentication Protocol)

PAP (Password Authentication Protocol) est un protocole d'authentification par mot de passe défini dans la RFC 1334 en 1992.

La demande d'authentification du protocole PAP est indiquée par la présence de la valeur C023 dans le champ Protocol de la trame PPP. Les champs du paquet PAP sont illustrés à la figure 39.11. La longueur de la zone de données transportant le protocole d'authentification est de 4 octets.

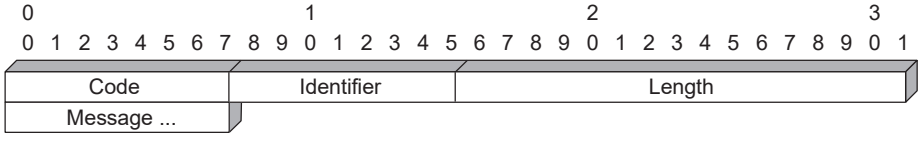


Figure 39.11

Format du paquet PAP

Le champ Code identifie la nature du paquet PAP. Il peut s'agir d'une requête d'authentification (Authentication Request) avec la valeur 1, d'un acquittement positif (Authenticate ACK) avec la valeur 2 ou d'un acquittement négatif de la demande (Authenticate NACK) avec la valeur 3. La structure des paquets correspondants est illustrée aux figures 39.12 et 39.13.

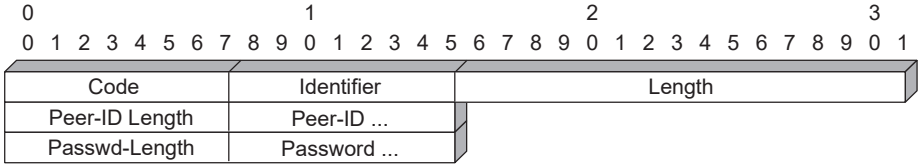


Figure 39.12

Structure du paquet de requête d'authentification

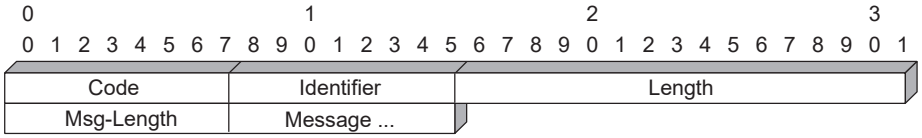


Figure 39.13

Structure du paquet d'authentification et de non-authentification

Le champ Identifier contient le numéro d'une requête et de la réponse associée. Le champ Length détermine la longueur totale du paquet PAP.

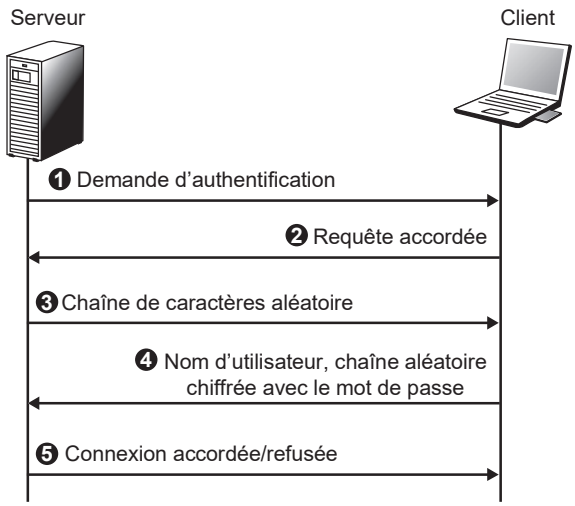
CHAP (Challenge Handshake Authentication Protocol)

Le protocole CHAP a été normalisé par la RFC 1334en 1994.

La demande d'authentification CHAP est indiquée par la valeur C223 dans le champ Protocol de la trame PPP. La figure 39.14 illustre le processus d'authentification du protocole CHAP.

Figure 39.14

Processus d'authentification du protocole CHAP



Le format du paquet CHAP est illustré à la figure 39.15.

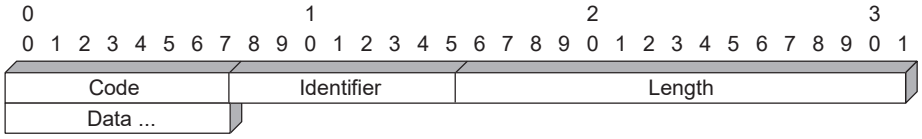


Figure 39.15

Format du paquet CHAP

Lorsque le champ code du paquet CHAP vaut 1-Challenge ou 2-Response, le paquet a la structure illustrée à la figure 39.16. Si le code vaut 3-Success ou 4-Failure, la structure du paquet prend la forme illustrée à la figure 39.17. Dans ces paquets, le champ Identifier indique le numéro d'une requête et de la réponse associée, et le champ *Length* la longueur totale du paquet PAP.

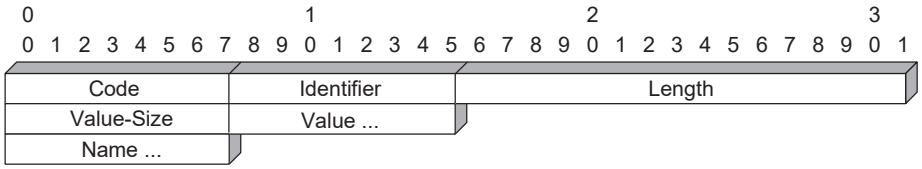


Figure 39.16

Paquet Challenge/Response

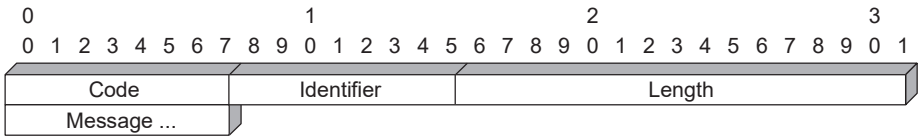


Figure 39.17

Paquet Success/Failure

MS-CHAP-V1

Le protocole MS-CHAP-V1 a été proposé par Microsoft et normalisé par l'IETF sous la RFC 2433 en 1998. Ce protocole est compatible avec la RFC de base datant de 1994.

Dans l'univers Microsoft, la sécurité d'un ordinateur personnel est fortement corrélée au mot de passe de son utilisateur. Ce dernier n'est jamais stocké en clair dans la mémoire de la machine. À partir d'un mot de passe, une empreinte MD4 de 16 octets est calculée puis mémorisée par le système hôte. Cette valeur, parfois nommée clé NT (NtPassword-Hash) est complétée par cinq octets nuls. On obtient ainsi 21 octets, interprétés comme une suite de trois clés DES de 56 bits chacune.

La méthode MS-CHAP-V1 est une authentification simple. Le serveur d'authentification produit un nombre aléatoire de 8 octets, et l'authentifié utilise ses trois clés DES pour chiffrer cet aléa, ce qui génère une réponse de 24 octets.

La demande d'authentification MS-CHAP-V1 est indiquée par la valeur C223 dans le champ Protocol de la trame PPP, complétée par un numéro d'algorithme prenant la valeur 0x80. Le format des paquets MS-CHAP-V1 est identique à celui des paquets CHAP. Les formats permettant de transporter les indications *Challenge/Response* et *Success/Failure* sont également les mêmes que dans le protocole CHAP. La différence provient de la taille du challenge, qui est de 8 octets. La taille de la réponse est de 25 octets, 24 octets pour les formats LAN manager et Windows NT et 1 octet (Use Windows NT compatible Challenge Response Flag) indiquant la disponibilité du format Windows NT. Le champ Name indique l'identifiant du compte utilisateur, c'est-à-dire nom de domaine plus le nom d'utilisateur.

Les indications 3-Success et 4-Failure comportent toujours une zone Identifier, qui donne le numéro d'une requête et de la réponse associée, et un champ *Length*, qui précise la longueur totale du paquet PAP.

Les messages portant les valeurs E, C et V indiquent :

- E : Error-Code ;
- R : retry allowed(1/0) ;
- C : new-challenge-value(16 hexadecimal value) ;
- V : decimal-version-code.

Deux nouvelles indications, les valeurs 5 et 6, permettent de modifier un mot de passe. Pour la valeur 5-Change Password Packet (version 1), les champs ont la valeur 5 pour le code et une longueur de 72 octets.

Ces 72 octets se décomposent de la façon suivante :

- 16 octets pour Encrypted LAN Manager Old password Hash ;
- 16 octets pour Encrypted LAN Manager New Password Hash ;
- 16 octets pour Encrypted Windows NT Old Password Hash ;
- 16 octets pour Encrypted Windows NT New Password Hash ;
- 2 octets pour Password Length ;
- 2 octets pour Flags.

Dans la deuxième version du changement de mot de passe, le code 6 est utilisé. La longueur du champ est de 1 114 octets, qui se décomposent de la façon suivante :

- 516 octets pour Password Encrypted with Old NT Hash ;
- 16 octets pour Old NT Hash Encrypted with New NT Hash ;
- 516 octets pour Password Encrypted with Old LM Hash ;
- 16 octets pour Old LM Hash Encrypted With New NT Hash ;
- 24 octets pour LAN Manager compatible challenge response ;
- 24 octets pour Windows NT compatible challenge response ;
- 2 octets pour Flags1.

Les mécanismes d'authentification de Windows NT utilisent un mot de passe. Ce dernier est constitué d'une chaîne Unicode de 256 caractères au plus. Le NTPasswordHash est le résultat d'un Hash MD4 produisant 16 octets (128 bits). Ce NTPasswordHash est complété par 5 octets nuls. On obtient ainsi 21 octets, décomposés en trois clés DES de 7 octets. Le challenge de 8 octets est chiffré par les trois clés DES, qui produisent une réponse de 24 octets : DES1(challenge), DES2(challenge), DES3(challenge).

Le PasswordHash (128 bits) est également utilisé comme clé de chiffrement RC4 dans les messages de modification de mot de passe. Un mot de passe est complété pour atteindre 256 caractères (512 octets) par une suite aléatoire. Cette valeur concaténée à la taille réelle (un entier de 4 octets) est chiffrée par la clé RC4, soit 516 octets.

MS-CHAP-V2

Le protocole MS-CHAP-V2 a été normalisé en 2000 par l'IETF sous la RFC 2759. MS-CHAP-V2 est une extension du protocole précédent, avec lequel il est compatible. L'objectif de cette nouvelle version est d'offrir une sécurité supérieure aux connexions d'accès distant en corrigeant certains problèmes de la précédente, comme la faiblesse des clés de chiffrement.

La demande d'authentification MS-CHAP-V2 est indiquée par la valeur 0x81 du champ Algorithm du protocole CHAP. Le format des paquets de la version 2 est similaire à celui de la version 1.

Le processus d'authentification est le suivant : le serveur d'authentification délivre un nombre aléatoire de 16 octets (AuthenticatorChallenge) ; le client 802.1x calcule un nombre de 8 octets à partir de cette valeur, d'un aléa (PeerChallenge) qu'il génère et du nom de l'utilisateur (login) ; ce paramètre est chiffré comme dans MS-CHAP-V1 par la clé NT pour obtenir une valeur de 24 octets.

Dans une plate-forme Microsoft, un annuaire stocke le nom des utilisateurs et leur mot de passe. La taille de la réponse est de 49 octets, qui se décomposent de la façon suivante :

- 16 octets pour le Peer-Challenge, qui porte un nombre aléatoire ;
- 8 octets réservés et codés à zéro ;
- 24 octets pour le format de réponse NT (NT-Response)
- 1 octet réservé et codé à zéro.

Le champ Name indique l'identifiant du compte utilisateur (nom-de-domaine\nom-utilisateur). Pour les codes 3-Success et 4-Failure, la longueur du champ Message est de 42 octets.

Le format de ce champ est S=<auth_string> M=<message>, auth_string étant une chaîne de 20 caractères ASCII et message un texte affichable compréhensible.

Lorsqu'un message d'erreur est envoyé en retour, il se présente sous la forme suivante :

```
E=Error-Code R=retry allowed(1/0) C=new-challenge-value(32 hexadecimal value)
➥V=decimal-version-code
```

La longueur du code 7, qui indique un changement de mot de passe, est de 586 octets, qui se décomposent de la façon suivante :

- 516 octets pour Encrypted-Password ;
- 16 octets pour Encrypted-Hash ;

- 16 octets pour Peer-Challenge ;
- 8 octets pour Reserved ;
- 24 octets pour NT-Response ;
- 2 octets pour Flags (réservé et codé à zéro).

Comme pour la version précédente, les mécanismes d'authentification que l'on trouve dans NT comprennent le mot de passe, qui est une chaîne Unicode de 256 caractères au plus. Pour la génération de la NT-Response, une procédure (ChallengeHash) fondée sur la fonction SHA-1 produit un nombre (challenge) de 8 octets à partir du nombre aléatoire AuthenticatorChallenge, d'un nombre aléatoire PeerChallenge et du UserName.

Le Password est associé à une empreinte MD4 de 16 octets (NTPasswordHash), étendue à 21 octets, et interprété comme une série de trois clés DES de 7 octets. Le champ Challenge (8 octets) est chiffré par les trois clés DES, DES1(challenge), DES2(challenge) et DES3(challenge). Ces 24 octets constituent la NT-Response. Le PasswordHash (128 bits) est utilisé comme clé RC4 pour le chiffrement d'un nouveau mot de passe. Les deux premières clés DES déduites d'un PasswordHash sont utilisées pour le chiffrement du NtPasswordHash associé au nouveau mot de passe.

EAP (Extensible Authentication Protocol)

Le problème de la gestion de la mobilité des utilisateurs est devenu critique dès lors que les internautes ont massivement utilisé des modems et le protocole PPP pour accéder aux ressources offertes par leurs fournisseurs de services. Les systèmes d'exploitation ont donc intégré les fonctionnalités afin de renforcer la sécurité des nomades :

- authentification des utilisateurs par des méthodes de défi telles que CHAP, MS-CHAP ou MS-CHAP-V2 ;
- chiffrement des trames PPP, par exemple à l'aide de l'algorithme MPPE (Microsoft Point-To-Point Encryption), défini par la RFC 3078 en mars 2001 ;
- méthodes de calcul des clés de chiffrement (MS-MPPE-Recv-Key et MS-MPPE-Send-Key) ;
- distribution des clés par le protocole RADIUS.

Le besoin de compatibilité avec des infrastructures d'authentification diversifiées et la nécessité de disposer de secrets partagés dans ces environnements multiples ont conduit à la genèse du protocole EAP, capable de transporter des méthodes d'authentification indépendamment de leurs particularités.

Le protocole EAP fournit un cadre peu complexe pour le transport de protocoles d'authentification. Un message comporte un en-tête de 5 octets et des données optionnelles, comme illustré à la figure 39.18.

Il existe quatre types de messages, identifiés par un code de 1 octet :

- Request : 1
- Response : 2
- Success : 3
- Failure : 4

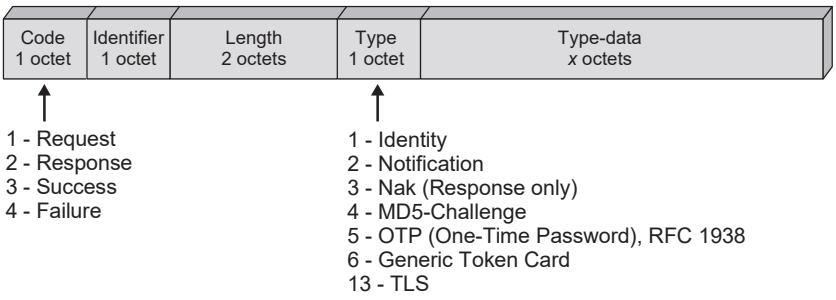


Figure 39.18

Format d'un message EAP

Chaque message est étiqueté à l'aide d'un nombre Identifier compris entre 0 et 255. L'étiquette d'une réponse est égale à celle de la requête correspondante. La longueur totale du message, codée sur deux octets, est comprise entre 4 et 65 535.

Le champ Type, compris entre 0 et 255, précise la nature des informations transportées. Les principales d'entre elles sont les suivantes :

- 1 : message relatif à l'identité (Identity).
- 2 : notification. Ce message contient une information affichable, et la réponse à une notification est obligatoirement une notification.
- 3 : notification d'une erreur (NAK).
- 4 : protocole d'authentification à base de défi MD5 (EAP-MD5).
- 6 : OTP.
- 13 : transport de TLS (EAP-TLS).
- 18 : méthode d'authentification fondée sur une carte SIM (EAP-SIM) pour le GSM 11.11, qui est la norme utilisée dans les réseaux GSM.
- 23 : EAP-AKA. Mise en œuvre des cartes USIM définies pour l'UMTS.
- 25 : PEAP. Méthode d'authentification du serveur, fondée sur TLS, et du client variable (MS-CHAP-V2, OTP, TLS, etc.).
- 26 : transport de MS-CHAP-v2.

Dans un réseau sécurisé avec 802.1x, une authentification EAP demande trois éléments : un supplicant que l'on appelle dans le langage courant un client, un authenticator que l'on appelle contrôleur et un serveur d'authentification. Un authenticator est un contrôleur de communication compatible 802.1x. La plupart des contrôleurs de communication dans les réseaux Ethernet sont compatibles 802.11x. Les points d'accès Wi-Fi professionnels sont souvent compatibles 802.1x et jouent le rôle d'authenticator. L'authentification se déroule de la manière suivante :

1. Lorsque la phase d'établissement de la liaison est terminée, le contrôleur envoie une requête d'identité.
2. Le client envoie un paquet EAP RESPONSE, dans lequel il fournit son identité et les méthodes d'authentification qu'il supporte. L'identité de l'utilisateur est indiquée par la valeur EAP-ID associée au message EAP-RESPONSE.IDENTITY.

3. Lorsque ce paramètre est similaire à une adresse de courrier électronique, ou NAI (Network Access Identifier), le contrôleur interprète la partie gauche, avant le caractère @, comme un login utilisateur et la partie droite comme le nom de domaine d'un serveur RADIUS. Une session d'authentification est initiée par le contrôleur grâce au message EAP-REQUEST.IDENTITY.
4. Le serveur d'authentification envoie alors un défi au client.
5. Le client y répond à nouveau par un message EAP RESPONSE.
6. L'authentification se poursuit par une suite de requêtes et de réponses (EAP-REQUEST.TYPE et EAP-RESPONSE.TYPE), relatives à un type, ou scénario d'authentification, particulier et échangés entre le serveur RADIUS et le client 802.1x.
7. Le contrôleur met fin à la phase d'authentification par l'intermédiaire d'un paquet de succès (EAP-SUCCESS) ou d'échec (EAP-FAILURE) qu'il aura reçu du serveur d'authentification AAA (Authentication, Authorization, Accounting). C'est ce dernier qui prend la décision d'accepter ou de refuser l'accès au réseau.
8. Si la phase d'authentification s'est bien déroulée, le serveur d'authentification peut transmettre une clé de chiffrement au contrôleur, qui l'utilisera pour chiffrer les données envoyées au client.

Cette dernière phase est optionnelle pour le protocole EAP, car elle dépend du protocole d'authentification utilisé.

Ce processus est illustré à la figure 39.19 où un point d'accès Wi-Fi compatible IEEE 802.1x joue le rôle du contrôleur, mais cette solution est évidemment générale et s'applique aussi bien à des réseaux hertziens qu'à des réseaux terrestres.

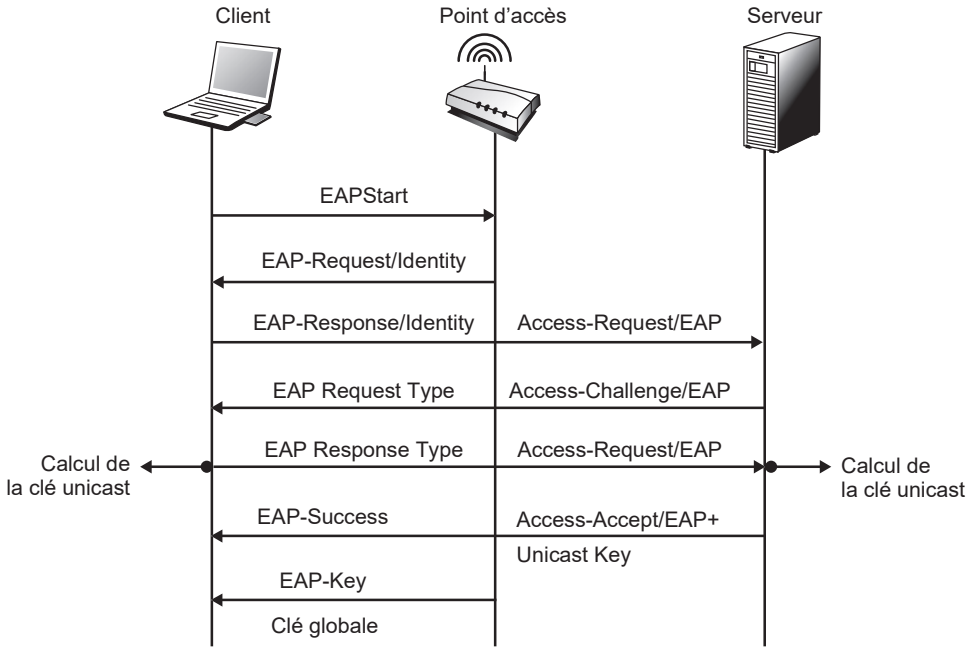


Figure 39.19

Session d'authentification

Un des points faibles du protocole EAP est sa vulnérabilité aux attaques par déni de service. Un pirate peut en effet écouter une session EAP et émettre à l'intention du client 802.1x un message d'échec (EAP-FAILURE*)*. Il ne peut toutefois obtenir la clé globale délivrée par le message EAP-RESPONSE-TYPE car cette dernière est chiffrée et signée par la clé unicast dont il ne connaît pas la valeur.

Les procédures d'authentification liées à EAP

Comme expliqué précédemment, EAP (Extensible Authentication Protocol) est devenu le tunnel standard pour l'authentification. On met en place ce tunnel pour réaliser la procédure d'authentification elle-même. Un vaste choix de mécanismes d'authentification est possible. LEAP (Lightweight Extensible Authentication Protocol) est la solution choisie par Cisco Systems pour ses premiers équipements de réseau sans fil. FAST-EAP devrait être un des standards mis en avant par Cisco à l'avenir, LEAP montrant quelques faiblesses dans des cas particuliers comme l'attaque par dictionnaire pour peu que les mots de passe ne soient pas sophistiqués. EAP/SIM et EAP/TLS sont les deux grands standards du moment. Ils correspondent aux choix effectués par les opérateurs de réseaux de mobiles et par de nombreux éditeurs de logiciels, dont Microsoft. Deux solutions supplémentaires, PEAP (Protected EAP) et EAP par carte à puce, sont poussées par Microsoft pour la première et par les équipementiers de la carte à puce pour la seconde.

LEAP (Lightweight Extensible Authentication Protocol)

L'architecture LEAP s'appuie sur la procédure d'authentification disponible sur les plates-formes Windows.

L'authentification LEAP fonctionne de la façon suivante *(voir figure 39.20) :*

1. À partir du mot de passe utilisateur, on calcule une empreinte MD4 de 16 octets. Cette dernière est complétée par cinq octets nuls. On obtient ainsi une suite de 21 octets interprétée sous la forme de trois clés DES de 7 octets, soit 56 bits. Le mécanisme d'authentification, de type CHAP, consiste à chiffrer un nombre aléatoire de 8 octets à l'aide des trois clés DES associées à un utilisateur, ce qui produit une réponse de 24 octets. LEAP est associé au type EAP 17 (0x11) pour réaliser une double authentification, entre le serveur d'authentification et le supplicant (utilisateur du réseau), d'une part, et entre l'authenticator (point d'accès) et le serveur d'authentification, d'autre part.
2. Au terme d'un scénario d'authentification réussi entre supplicant et serveur RADIUS (correspondant aux phases 1 à 5 de la figure 39.20), les deux entités déduisent une clé de session SK (unicast), qui est transportée à l'aide d'un attribut propriétaire (CISCO-AVPAIR, LEAP SESSION-KEY) du protocole RADIUS. LEAP supporte également des mécanismes de mise à jour de clés WEP, soit par la négociation d'une session RADIUS limitée (Session Timeout), soit par des demandes périodiques de réauthentification par le supplicant à l'aide des trames EAP LOGOFF et EAP START.

Le format du paquet LEAP est illustré à la figure 39.21.

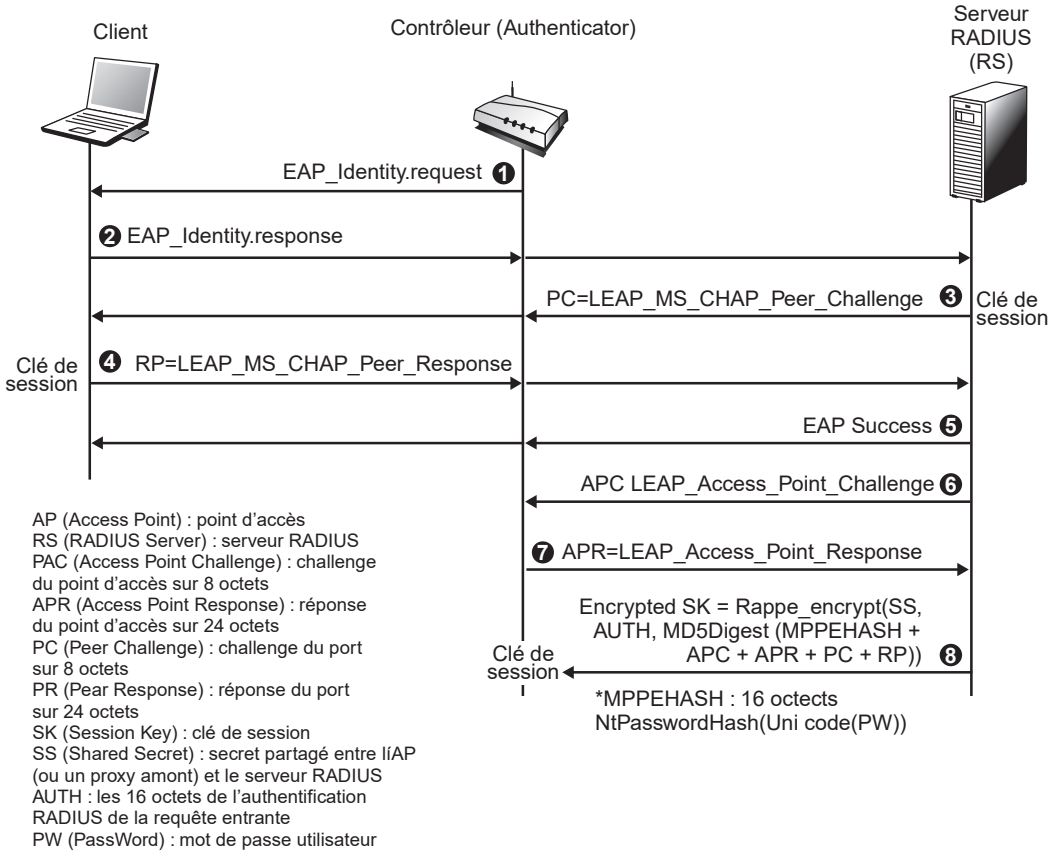
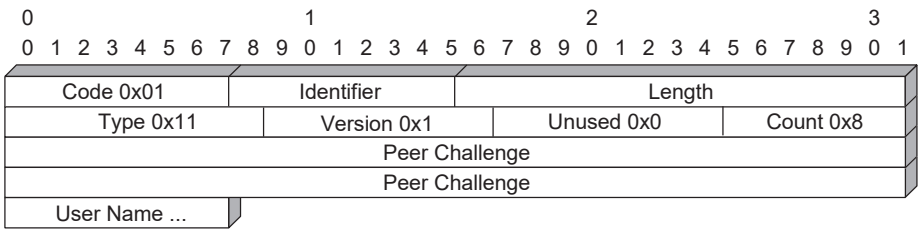


Figure 39.20

Processus d'authentification LEAP

Figure 39.21

Paquet LEAP



EAP-FAST

EAP-FAST (Extensible Authentication Protocol-Flexible Authentication via Secure Tunneling)a été développé par Cisco Systems pour résoudre une faille de sécurité de son protocole propriétaire LEAP (Lightweight EAP), que nous venons d'examiner, lorsque les mots de passe ne sont pas assez sophistiqués.

Ce protocole vise notamment à contrer les attaques par dictionnaire utilisées avec succès contre LEAP. Contrairement à PEAP, que nous verrons un peu plus loin dans ce chapitre,

qui est le fruit d'une alliance entre Cisco, Microsoft et RSA Security, EAP Fast ne requiert pas la mise en place d'une infrastructure complexe de distribution de certificats pour l'établissement de tunnels sécurisés entre machines terminales.

EAP FAST est intégré dans l'ensemble des produits Aironet de Cisco ainsi que dans son serveur VPN Cisco Secure ACS. Les partenaires de Cisco auront aussi accès au standard dans le cadre de la spécification Cisco Compatibility Extensions 3.0.

De façon plus précise, EAP-FAST est une architecture de sécurité de type client-serveur, qui chiffre les transactions EAP au moyen d'un tunnel TLS. Cette solution est assez semblable à PEAP, à la différence essentielle près que le tunnel EAP-FAST est établi à l'aide de secrets forts, qui appartiennent aux utilisateurs. Ces secrets sont appelés PAC (Protected Access Credentials). Ils sont générés par le serveur Cisco Secure ACS à l'aide d'une clé maître connue uniquement du serveur Cisco Secure ACS. Les handshakes réalisés par des secrets partagés étant beaucoup plus rapides à mettre en œuvre qu'une PKI, EAP-FAST est plus simple à mettre en place que les solutions qui chiffrent les transactions EAP, comme EAP-TLS ou PEAP.

EAT-FAST s'exécute en trois phases :

- La phase 0, spécifique d'EAP-FAST, consiste à ouvrir un tunnel sécurisé entre les machines terminales en utilisant les certificats PAC. Le tunnel est établi par un échange de clés au moyen d'une procédure de type Diffie-Hellman. Si l'authentification EAP-MSCHAPv2 réussit, le serveur Cisco Secure ACS donne un certificat PAC à chaque client. Cette phase 0 est optionnelle si les certificats sont introduits par une autre méthode assurant le secret des certificats.
- En phase 1, le serveur Cisco Secure ACS et les machines terminales établissent des tunnels TLS grâce aux PAC présents dans les machines terminales. La façon dont le PAC a été introduit dans la machine terminale est indépendante de la phase 1.
- En phase 2, le serveur Cisco Secure ACS authentifie les certificats des machines terminales par l'intermédiaire d'un EAP-GTC, qui est protégé par le tunnel TLS créé à la phase 1. Le protocole EAP-FAST ne supporte pas d'autre type d'EAP. Cisco Secure ACS autorise un service réseau au travers du point d'accès si la phase 2 s'est déroulée avec succès.

Cette solution est présentée par Cisco comme étant aussi simple que LEAP et aussi sécurisée que PEAP. En fait, EAP-FAST est un compromis entre les deux. Le fait de ne pas utiliser de PKI semble plus simple mais est en réalité aussi difficile à mettre en œuvre pour obtenir une bonne sécurité. De plus, la sécurité n'est pas aussi bonne qu'avec PEAP car la phase 0 peut conduire à des attaques décisives si elle n'est pas aussi sécurisée que peut l'être une PKI.

EAP-SIM (Subscriber Identity Module)

Une solution classique d'authentification est proposée par les opérateurs de téléphones mobiles de deuxième génération, ou GSM, selon une procédure d'authentification réalisée entre le serveur de l'opérateur et la carte SIM (Subscriber Identity Module) située dans le terminal de l'utilisateur. Cette authentification utilise non pas le protocole EAP mais des protocoles provenant de l'ETSI effectuant un travail comparable.

Les sections qui suivent décrivent ce mécanisme avant de présenter EAP-SIM, une extension normalisée d'EAP pour le monde IP que les opérateurs peuvent, par exemple, utiliser dans les hotspots.

L'authentification du GSM

Le GSM est un standard de téléphonie mobile défini par l'ETSI (European Telecommunications Standards Institute). Il supporte des opérations de sécurité telles que l'authentification de l'utilisateur et le chiffrement entre le réseau nominal, où l'abonné est inscrit, et la carte SIM de l'abonné.

Les éléments du réseau GSM intervenant dans ces fonctions de sécurité sont les suivants :

- AuC (Authentication Center), ou centre d'authentification du réseau de l'opérateur.
- HLR (Home Location Register), ou base de données des abonnés de l'opérateur, qui mémorise les données de chaque abonné, telles que son identité internationale, ou IMSI (International Mobile Subscriber Identity), son numéro de téléphone, son profil d'abonnement, etc. Il stocke aussi pour chaque abonné le numéro de VLR courant.
- VLR (Visitor Location Register), ou base de données des seuls abonnés localisés dans la zone géographique gérée.

Les données d'authentification sont stockées dans la carte SIM et ne sont pas chargées dans le terminal mobile. La procédure d'authentification consiste donc en un échange de messages entre la carte SIM et le réseau.

Lors de l'inscription d'un nouvel abonné, une clé Ki (jusqu'à 128 bits) lui est attribuée. Cette clé est secrète et n'est stockée que sur sa carte SIM et sur l'AuC de l'opérateur.

La procédure d'authentification se déroule de la façon suivante :

1. Le réseau transmet au mobile un nombre aléatoire RAND, codé sur 128 bits.
2. La carte SIM du mobile calcule la signature de RAND grâce à l'algorithme d'authentification A3 et sa clé Ki. Le résultat, SRES (32 bits), est envoyé par le mobile au réseau.
3. Le réseau compare SRES avec le résultat calculé de son côté. Si les deux coïncident, l'abonné est authentifié.

Une fois l'abonné authentifié, le chiffrement est effectué selon l'algorithme A5. Il utilise la clé Kc, de 64 bits, calculée à partir de la clé secrète Ki et du nombre aléatoire RAND, selon l'algorithme A8.

Il suffit au réseau de disposer d'un triplé (RAND, SRES, Kc) pour authentifier un abonné et activer le chiffrement de ses communications. Cependant, le réseau ne calcule pas ces données en temps réel. L'AuC prépare des triplés pour chaque abonné et les transmet à l'avance au HLR, qui les stocke. Le VLR qui a besoin d'un triplé en effectue la demande le moment venu.

La procédure d'authentification entre l'équipement mobile et le VLR/HLR est illustrée à la figure 39.22.

L'algorithme A5 est implémenté dans chaque terminal et dans le réseau. Les implémentations des algorithmes A3 et A8, aussi appelés COMP128, existent sur Internet, mais aucun standard n'a encore été publié.

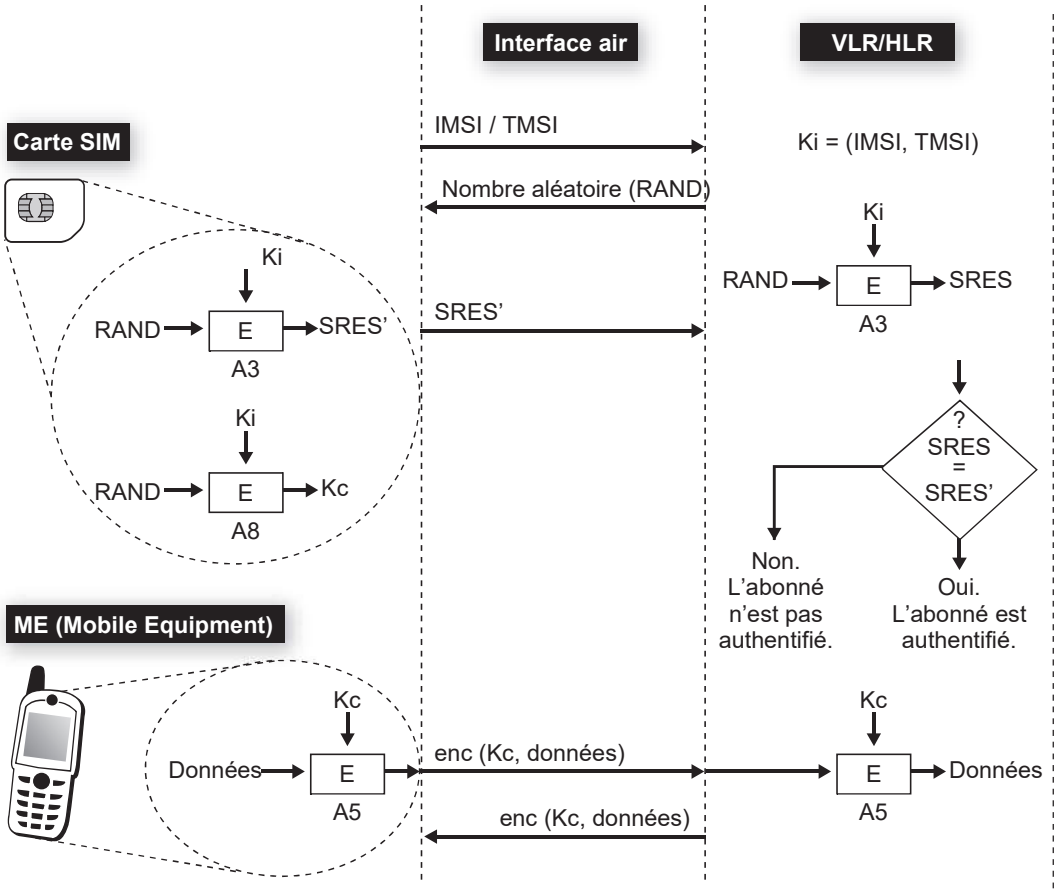


Figure 39.22

Authentification dans les réseaux GSM

L'authentification EAP-SIM

Les hotspots, ou zones publiques à forte densité de population, telles que gares, aéroports, etc., peuvent être vus par les opérateurs de mobiles comme une extension possible de leur réseau. Il existe pour ces hotspots un mécanisme d'authentification mutuelle fondé sur le module SIM, appelé EAP-SIM. Ce protocole complète les procédures d'authentification utilisées par le GSM en fournissant une authentification entre le centre d'authentification de l'opérateur mobile et chaque module SIM. Les algorithmes d'authentification sont présents à la fois dans le réseau et dans toutes les cartes à puce SIM.

La solution EAP-SIM interagit directement avec les cartes à puce existantes. Sur le terminal, le composant logiciel qui implémente le protocole EAP-SIM peut utiliser PC/SC (Personal Computer/Smart Card), un environnement défini par un groupe d'industriels mené par Microsoft, pour communiquer directement avec la carte à puce de l'abonné. Une telle configuration ne nécessite aucune modification du réseau cœur GSM pour implémenter EAP-SIM. Par contre, il est nécessaire d'implémenter les communications entre le serveur d'authentification et le HLR/AuC, côté serveur, et entre le logiciel EAP-SIM et la carte SIM, côté client.

Une solution innovante a également été mise en place par un des fabricants majeurs de téléphones portables, permettant à une carte réseau 802.11 de communiquer directement avec un module SIM intégré, sans passer par le terminal, renforçant ainsi la sécurité.

L'identité (EAP-ID) est obtenue par la concaténation du caractère 1 de la valeur, exprimée en une suite de chiffres ASCII, de l'IMSI, du caractère @ et du nom de domaine de l'opérateur (EAP-ID = 1IMSI@operator.com).

L'authentification EAP-SIM se déroule de la manière suivante *(voir figure 39.23) :*

1. Soit C le client et A le point d'accès. Dans ce processus, A utilise trois triplés d'authentification (RAND, Kc, SRES) :

$$C \rightarrow A: RC$$

   Lors de cette première étape, le client C envoie au point d'accès A un défi aléatoire Rc.

2. A répond au client par la liste des trois nombres aléatoires RAND1, RAND2 et RAND3 provenant de trois triplés. Il envoie aussi le MAC calculé sur ces 3 nombres et sur Rc (MACk) :

$$A \rightarrow C: RAND1, RAND2, RAND3, MACk[\ldots, RAND1, RAND2, RAND3, Rc]$$

3. La clé K, permettant le calcul de MACk, a été préalablement calculée par le point d'accès par dérivation d'une clé maître MK=SHA[…,Kc1,Kc2,Kc3,Rc,…], où Kc1, Kc2 et Kc3 sont les clés Kc des 3 triplés.

$$C \rightarrow A: MACk[\ldots, SRES1, SRES2, SRES3]$$

4. Quand C reçoit MACk et la liste de nombres aléatoires RAND, il vérifie le MACk. Pour ce faire, C utilise Ki (présente sur la carte à puce de l'utilisateur et partagée avec le serveur d'authentification) pour retrouver les clés Kc1, Kc2 et Kc3. Ces dernières lui permettent de générer MK, qu'il utilise pour calculer K par dérivation.

5. Avec cette même clé K, C calcule le MAC sur les trois valeurs SRES des triplés et envoie le résultat au point d'accès. À son tour, A vérifie le MAC et la liste de SRES qu'il a reçus du réseau GSM. Si les résultats obtenus sont identiques, C est authentifié.

Grâce à la technologie EAP-SIM, les opérateurs de téléphonie peuvent utiliser leur base de données client (HLR) pour assurer la facturation des services sans fil.

EAP-TLS (Transport Layer Security)

L'authentification EAP-TLS (Transport Layer Security) est devenue la technique d'authentification la mieux reconnue et est considérée comme l'une des plus solides grâce à l'authentification mutuelle qui est exercée. En fait, TLS n'est qu'une extension de la procédure SSLv3, qui est fortement utilisée pour les authentifications de niveau application entre client et serveur Web.

Cette solution EAP-TLS est celle qui a été choisie par de très nombreuses entreprises. Microsoft, par exemple, en possède une version en standard dans son système d'exploitation depuis Windows 2000.

Défini par la RFC 2716 d'octobre 1999, EAP-TLS s'appuie sur une infrastructure de type PKI. Le serveur RADIUS et le client du réseau sont munis de certificats délivrés par une autorité de certification (Certificate Authority) commune.

Le format du paquet EAP-TLS est illustré à la figure 39.24.

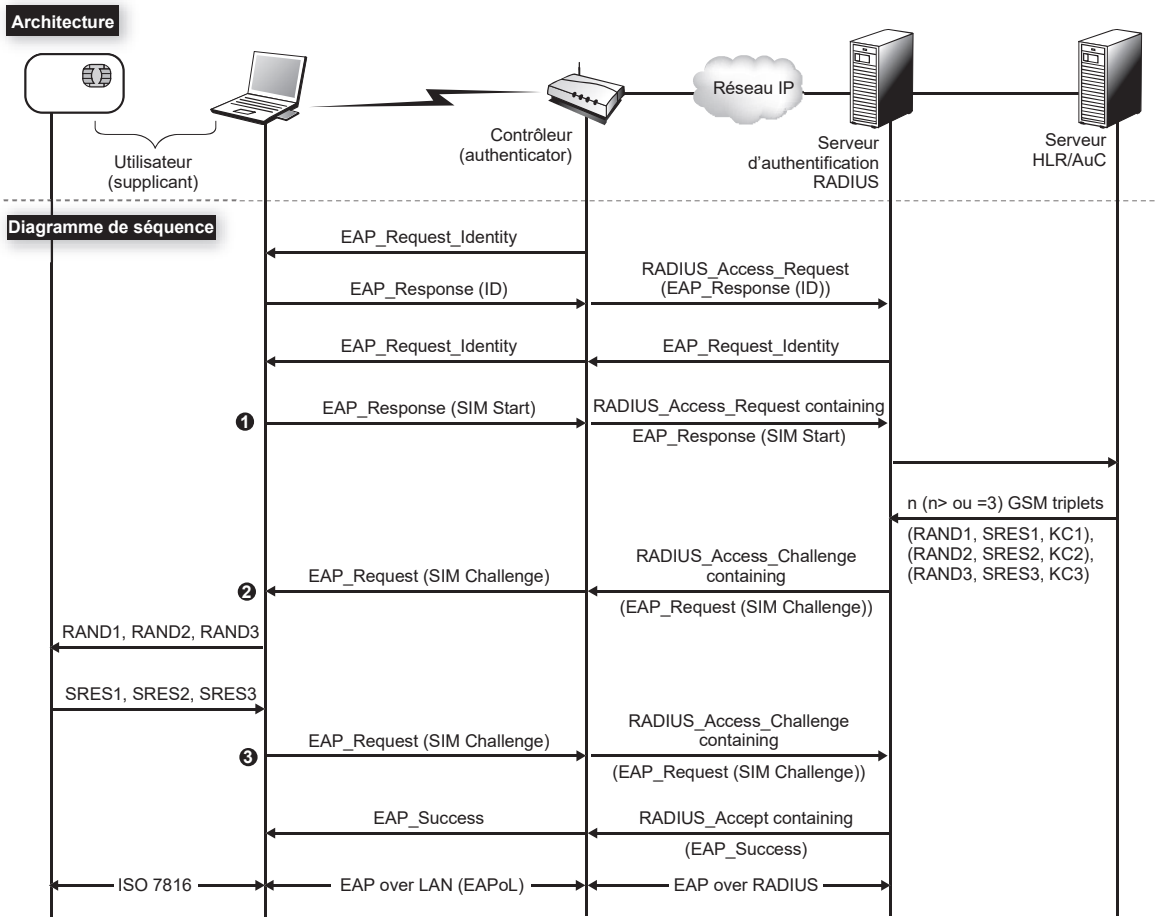


Figure 39.23

Authentification EAP-SIM

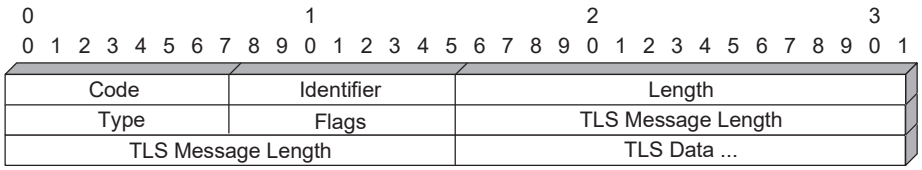


Figure 39.24

Paquet EAP/TLS

EAP-TLS utilise le handshake TLS pour permettre au client et au serveur d'échanger leur certificat numérique, fondement de l'authentification. Le serveur présente un certificat au client, que ce dernier valide. Optionnellement, le client présente son certificat au serveur. Le certificat peut être protégé côté client par un mot de passe, un code PIN ou une carte à puce.

Une conversation EAP-TLS entre un client demandant un accès au réseau et le point d'accès se déroule de la façon suivante :

1. Le point d'accès envoie un paquet EAP-REQUEST/IDENTITY.
2. Le client répond par un paquet EAP-RESPONSE/IDENTITY, contenant l'identité de l'utilisateur.
3. Le serveur envoie un paquet EAP-TLS/START.
4. La réponse du client est un paquet EAP-RESPONSE contenant un message TLS CLIENT_HELLO HANDSHAKE. Le message CLIENT_HELLO contient la version TLS du client, un nombre aléatoire et une liste d'algorithmes de chiffrement supportés par le client.
5. Le serveur envoie un paquet EAP-REQUEST dont les données contiennent un message SERVER_HELLO HANDSHAKE. Ce message spécifie la version de TLS du serveur, un autre nombre aléatoire, un identifiant de session et un message CIPHERSUITE correspondant à l'algorithme de chiffrement choisi.
6. Le client répond par un paquet EAP-RESPONSE, dont le champ de données encapsule un message TLS_CHANGE_CIPHER_SPEC et un message FINISHED_HANDSHAKE.

La figure 39.25 illustre les différents messages envoyés lors de la phase d'authentification. Ce cas représente une authentification réussie entre l'authentifiant et le client.

Le TLS Master Secret, ou MSK (Master Session Key), est le secret partagé entre le client et le serveur, résultat de la phase de handshake.

Les données suivantes sont dérivées à partir de MSK :

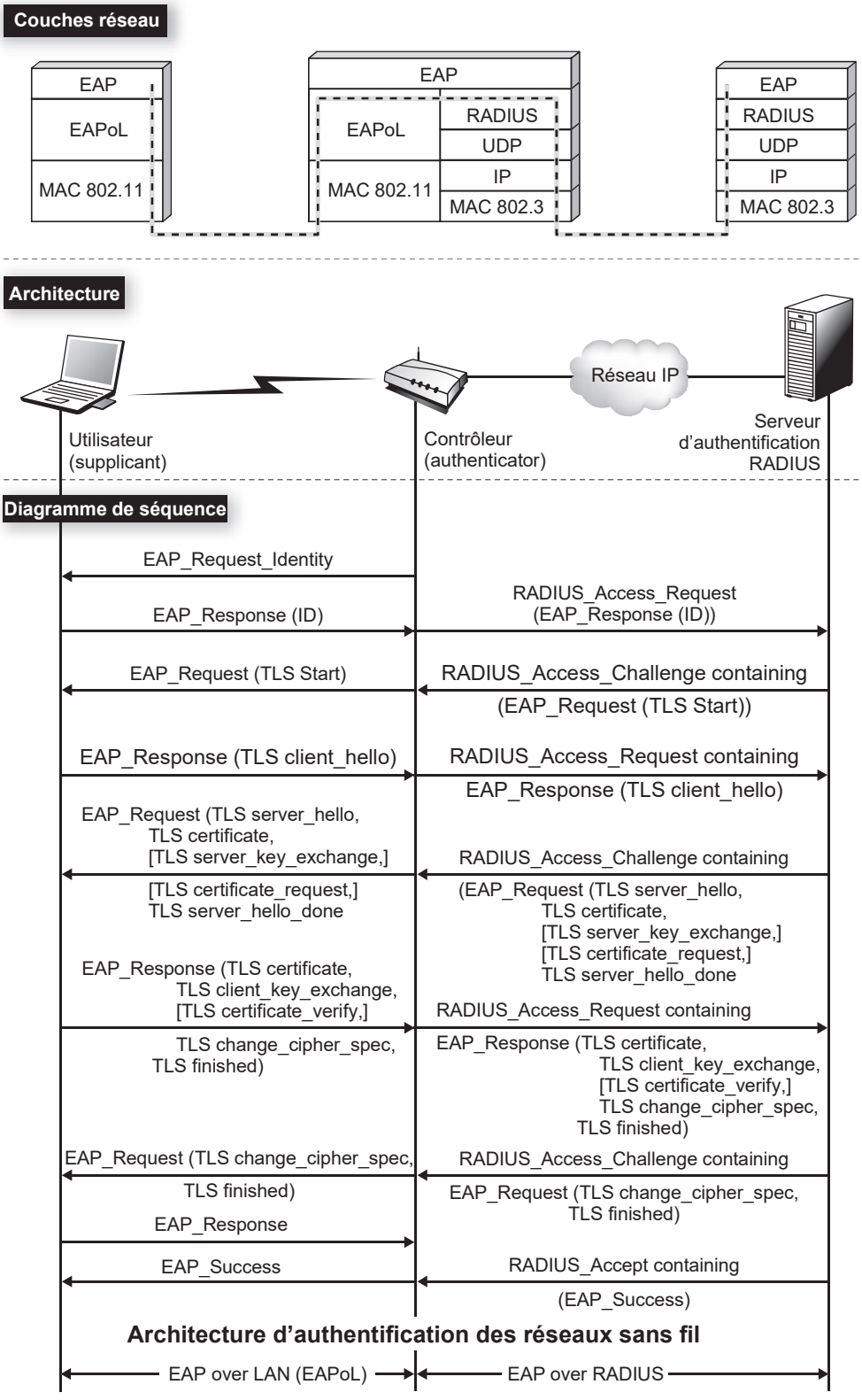
- clé de chiffrement client (MSK(0,31)) ;
- clé de chiffrement serveur (MSK(32,63)) ;
- clé d'authentification client pour le calcul du MAC côté client (MSK(64,95)) ;
- clé d'authentification serveur pour le calcul du MAC côté serveur (MSK(96,127)) ;
- deux vecteurs d'initialisation (IV).

La hiérarchie des clés dérivées est illustrée à la figure 39.26.

La clé de chiffrement client, aussi appelée PMK (Pairwise Master Key), est transmise au point d'accès *via* l'attribut RADIUS MS-MPPE-RECV-KEY. La clé WEP sera chiffrée avec cette clé puis signée avant d'être remise au client.

Le serveur d'authentification peut vérifier si le certificat d'un client est révoqué. Inversement, le client peut vérifier la validité du certificat du serveur. Cette vérification ne peut toutefois s'effectuer qu'une fois la phase de connexion achevée. En effet, un client en train d'initier une conversation de niveau liaison n'a pas de connectivité.

Le transport de messages TLS pose essentiellement un problème de segmentation. La taille d'un enregistrement TLS est d'au plus 16 384 octets, mais le protocole RADIUS limite sa charge utile à 4 096 octets. De surcroît, la taille des trames 802.11 est limitée à 2 312 octets. EAP-TLS doit donc supporter un mécanisme de segmentation des enregistrements. Contrairement à l'usage courant de TLS, mettant en œuvre une authentification simple du serveur, EAP-TLS utilise une authentification mutuelle entre le serveur RADIUS et le client 802.1x *(voir figure 39.27)*.


Couches réseau
EAP
EAPoL
MAC 802.11
EAP
EAPoL
RADIUS
UDP
IP
MAC 802.11
MAC 802.3
EAP
RADIUS
UDP
IP
MAC 802.3
Architecture
Réseau IP
Utilisateur
(supplicant)
Contrôleur
(authenticator)
Serveur
d'authentification
RADIUS
Diagramme de séquence
EAP_Request_Identity
EAP_Response (ID)
RADIUS_Access_Request
(EAP_Response (ID))
EAP_Request (TLS Start)
RADIUS_Access_Challenge containing
(EAP_Request (TLS Start))
EAP_Response (TLS client_hello)
RADIUS_Access_Request containing
EAP_Response (TLS client_hello)
EAP_Request (TLS server_hello,
TLS certificate,
[TLS server_key_exchange,]
[TLS certificate_request,]
TLS server_hello_done
RADIUS_Access_Challenge containing
(EAP_Request (TLS server_hello,
TLS certificate,
[TLS server_key_exchange,]
[TLS certificate_request,]
TLS server_hello_done
EAP_Response (TLS certificate,
TLS client_key_exchange,
[TLS certificate_verify,]
TLS change_cipher_spec,
TLS finished)
RADIUS_Access_Request containing
EAP_Response (TLS certificate,
TLS client_key_exchange,
[TLS certificate_verify,]
TLS change_cipher_spec,
TLS finished)
EAP_Request (TLS change_cipher_spec,
TLS finished)
RADIUS_Access_Challenge containing
EAP_Request (TLS change_cipher_spec,
TLS finished)
EAP_Response
EAP_Success
RADIUS_Accept containing
(EAP_Success)
Architecture d'authentification des réseaux sans fil
EAP over LAN (EAPoL)
EAP over RADIUS

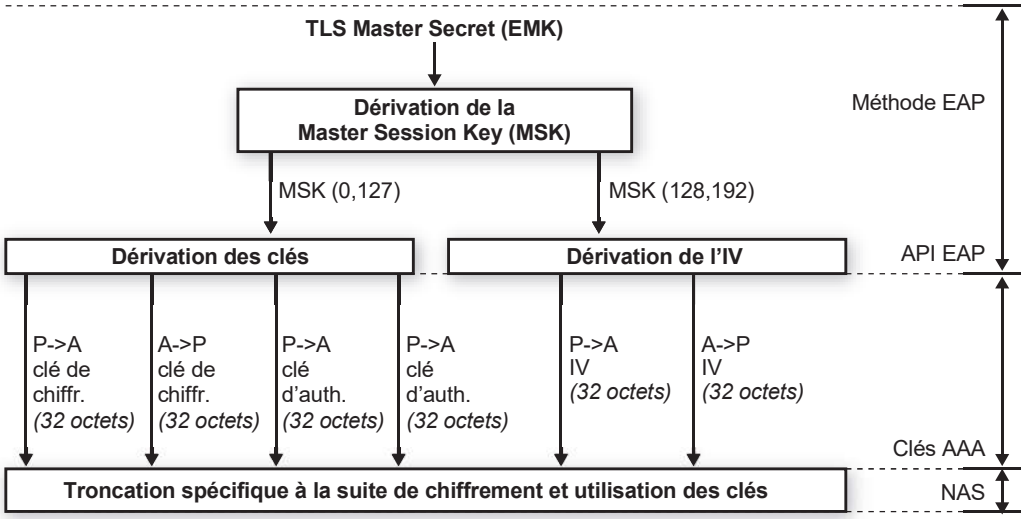


Figure 39.26

Schéma de dérivation des clés dans EAP-TLS

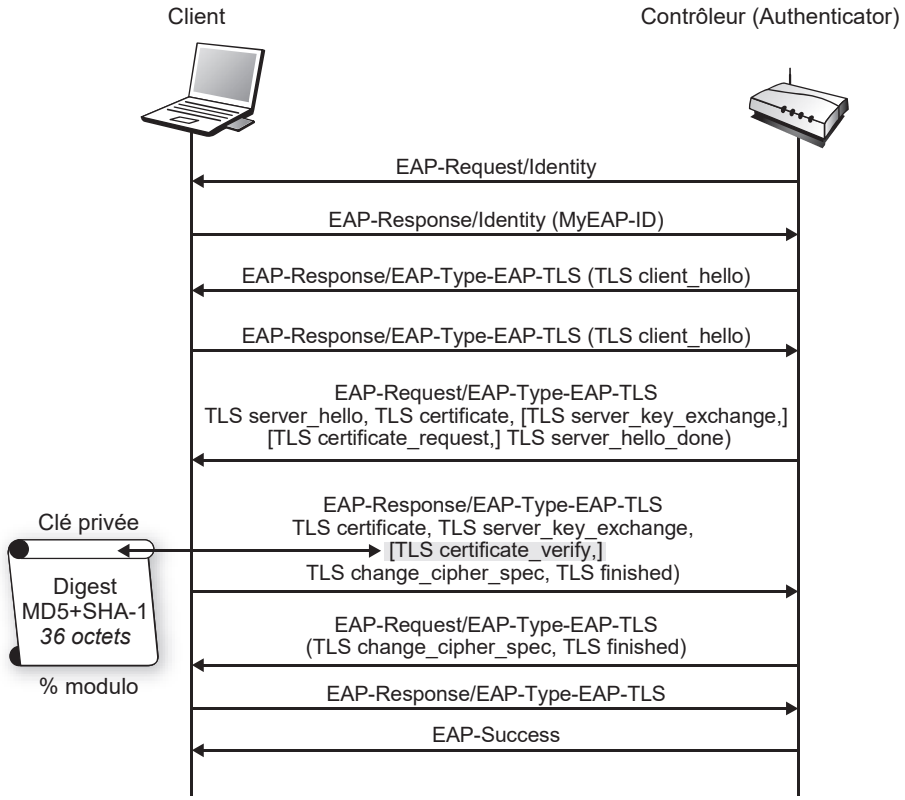


Figure 39.27

Authentification mutuelle EAP-TLS

L'usage d'une clé privé par le client 802.1x soulève le problème critique de la sécurité requise par son stockage ainsi que de la mise en œuvre d'un tel composant. Dans les plates-formes informatiques usuelles, cette sécurité est assurée par des mots de passe permettant de déchiffrer et d'utiliser la clé privée. La carte à puce constitue une solution de rechange plus sécurisée à cette méthode.

L'utilisation de l'authentification à base de certificats numériques oblige à posséder une infrastructure PKI convenable. Si une telle infrastructure n'est pas déployée, les certificats client entraînent un surplus important de gestion. Toutefois, EAP/TLS est nativement supporté sur les plates-formes Windows, où le certificat client peut être stocké dans une carte à puce.

PEAP (Protected Extensible Authentication Protocol)

Les installations sans fil actuellement déployées utilisent des protocoles d'authentification hétérogènes. De ce fait, la mobilité du client est difficile à gérer. Pour une entreprise, EAP offre l'avantage de réutiliser dans son environnement sans fil des mécanismes déjà adoptés dans l'environnement filaire.

La sélection d'une méthode d'authentification est une décision stratégique pour le déploiement sécurisé d'un réseau. La méthode d'authentification conduit au choix du serveur d'authentification, qui, à son tour, conduit au choix du logiciel client. Dans le cas où une infrastructure PKI n'est pas déjà déployée, il existe d'autres méthodes d'authentification, présentant un niveau de sécurité équivalent à celui obtenu avec les certificats numériques et permettant de s'affranchir des barrières liées à la mise en place d'une infrastructure PKI. Ces méthodes permettent aussi de protéger les procédures d'authentification du client fondées sur des mots de passe.

Par exemple, EAP-TTLS (Tunneled Transport Layer Security) et PEAP conservent les fortes fondations cryptographiques de TLS et d'EAP mais utilisent d'autres mécanismes pour authentifier le client.

Ces protocoles établissent d'abord un tunnel sécurisé TLS, après quoi le client authentifie le serveur *(voir figure 39.28)*.

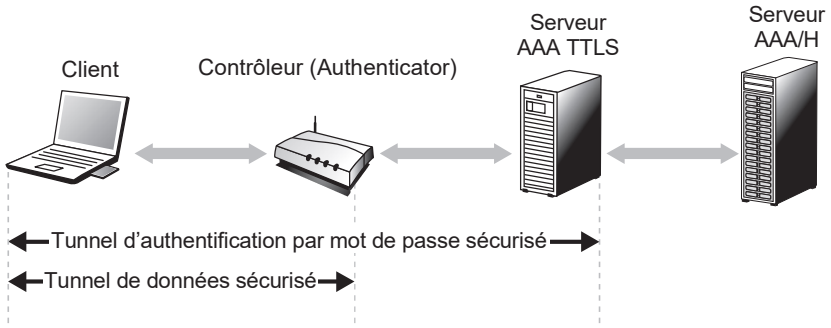


Figure 39.28

Tunnels PEAP et EAP-TTLS

Dans une seconde étape, des paquets d'authentification sont échangés. TTLS échange des AVP (Attribute-Value Pairs) avec un serveur, qui les valide pour tout type d'authentification. Le format des paires de valeurs d'attributs est illustré à la figure 39.29.

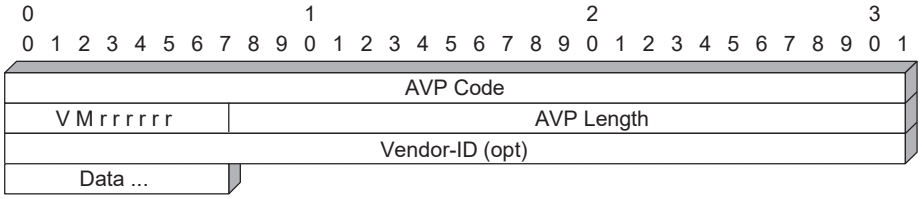


Figure 39.29

Format des paires de valeurs d'attributs

PEAP utilise le canal TLS pour protéger un second échange EAP. MS-CHAP-V2 peut être utilisé pour les clients n'ayant pas de PKI. Pour les clients ayant une PKI, EAP-TLS peut être utilisé. L'avantage de PEAP par rapport à l'EAP-TLS classique est que l'identité du client est protégée lors de l'échange.

La figure 39.30 illustre le principe de fonctionnement de PEAP.

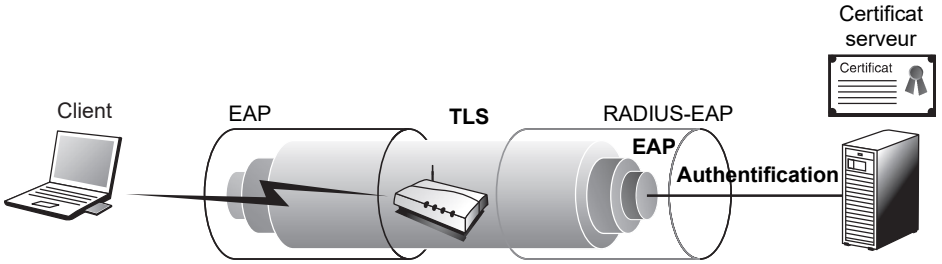


Figure 39.30

Principe de fonctionnement de PEAP

RADIUS (Remote Authentication Dial-In User Server)

Quel que soit le choix du mécanisme d'authentification entre le client et le serveur d'authentification, les paquets EAP sont généralement acheminés grâce au protocole RADIUS. RADIUS est depuis longtemps le protocole AAA (Authentication, Authorization, Accounting) le plus largement adopté. Utilisé par les FAI pour authentifier les utilisateurs, il est principalement conçu pour transporter des données d'authentification, d'autorisation et de facturation entre des NAS (Network Access Server) distribués, qui désirent authentifier leurs utilisateurs et un serveur d'authentification partagé.

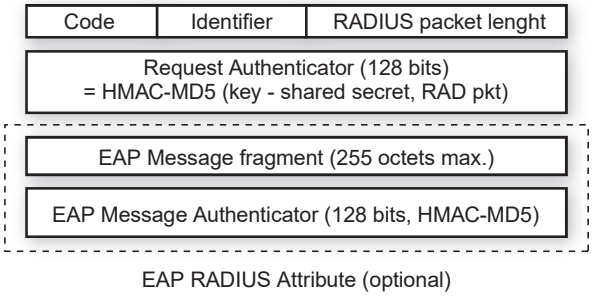
RADIUS utilise une architecture client-serveur qui repose sur le protocole UDP. Les NAS, qui jouent le rôle de client, sont responsables du transfert des informations envoyées par l'utilisateur vers les serveurs RADIUS. Ces derniers prennent en charge la réception des demandes d'authentification, l'authentification des utilisateurs et les réponses contenant toutes les informations de configuration nécessaires aux NAS. Les serveurs RADIUS peuvent également agir comme proxy pour d'autres serveurs RADIUS.

Si un équipement mobile a besoin d'accéder au réseau en utilisant RADIUS pour l'authentification, il doit présenter au NAS des crédits d'authentification (identifiant utilisateur, mot de passe, etc.). Ce dernier les transmet au serveur RADIUS en lui envoyant un ACCESS-REQUEST. Le NAS et les proxy RADIUS ne peuvent interpréter ces crédits d'authentification car ces derniers sont chiffrés entre l'utilisateur et le serveur RADIUS destinataire. À réception de cette requête, le serveur RADIUS vérifie l'identifiant du NAS puis les crédits d'authentification de l'utilisateur dans une base de données LDAP (Lightweight Directory Access Protocol) ou autre.

Les données d'autorisation échangées entre le client (le NAS) et le serveur RADIUS sont toujours accompagnées d'un secret partagé. Ce secret est utilisé pour vérifier l'authenticité et l'intégrité de chaque paquet entre le NAS et le serveur.

La figure 39.31 illustre le format type d'un paquet RADIUS. L'authentifiant du message sur 128 bits n'est autre qu'un résumé HMAC-MD5 du paquet échangé, calculé à l'aide du secret partagé.

Figure 39.31
Format type d'un paquet RADIUS

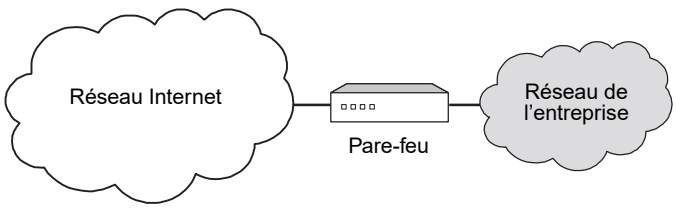


RADIUS peut supporter plusieurs mécanismes d'authentification. Il peut utiliser, par exemple, des procédures de défi/réponse (Chap) et des messages ACCEPT-CHALLENGE. L'authentification par mot de passe, ou PAP (Password Authentication Protocol), est aussi prise en charge. Les serveurs RADIUS répondent aux demandes d'authentification par des messages ACCESS-ACCEPT ou ACCESS-REJECT. Les paquets ACCESS-ACCEPT fournissent les informations de configuration nécessaires pour autoriser les clients RADIUS à commencer une connexion sécurisée avec des utilisateurs.

Les pare-feu

Un pare-feu est un équipement de réseau, la plupart du temps de type routeur, placé à l'entrée d'une entreprise afin d'empêcher l'entrée ou la sortie de paquets non autorisés par l'entreprise. La situation géographique d'un pare-feu est illustrée à la figure 39.32.

Figure 39.32
Situation d'un pare-feu dans l'entreprise



Toute la question est de savoir comment reconnaître les paquets à accepter et à refuser. Il est possible de travailler de deux façons :

- interdire tous les paquets sauf ceux d'une liste prédéterminée ;
- accepter tous les paquets sauf ceux d'une liste prédéterminée.

En règle générale, un pare-feu utilise la première solution en interdisant tous les paquets, sauf ceux qu'il est possible d'authentifier par rapport à une liste de paquets que l'on souhaite laisser entrer. Cela comporte toutefois un inconvénient : lorsqu'un client de l'entreprise se connecte sur un serveur à l'extérieur, la sortie par le pare-feu est acceptée puisque authentifiée. La réponse est généralement refusée, puisque le port sur lequel elle se présente n'a aucune raison d'accepter ce message s'il est bloqué par mesure de sécurité. Pour que la réponse soit acceptée, il faudrait que le serveur puisse s'authentifier et que le pare-feu lui permette d'accéder au port concerné.

L'autre option est évidemment beaucoup plus dangereuse puisque tous les ports sont ouverts sauf ceux qui ont été bloqués. Une attaque ne se trouve pas bloquée tant qu'elle n'utilise pas les accès interdits.

Avant d'aller plus loin, considérons les moyens d'accepter ou de refuser des flots de paquets. Les filtres permettent de reconnaître un certain nombre de caractéristiques des paquets, comme l'adresse IP d'émission, l'adresse IP de réception, parfois les adresses de niveau trame, le numéro de port et plus généralement tous les éléments disponibles dans l'en-tête du paquet IP. Pour ce qui concerne la reconnaissance de l'application, les filtres sont essentiellement réalisés sur les numéros de port utilisés par les applications. Nous verrons toutefois un peu plus loin que cette solution n'est pas imparable. Un numéro de port est en fait une partie d'un numéro de socket, ce dernier étant, comme expliqué au chapitre précédent, la concaténation d'une adresse IP et d'un numéro de port. Les numéros de port correspondent à des applications. Les principaux ports sont recensés au tableau 39.2.

Un pare-feu contient donc une table, qui indique les numéros de port acceptés.

Le tableau 39.3 donne la composition d'un pare-feu classique, dans lequel seulement six ports sont ouverts, dont l'un ne l'est que pour une adresse de réseau de classe C spécifique.

Les pare-feu peuvent être de deux types, proxy et applicatif. Dans le premier cas, le pare-feu a pour objectif de couper la communication entre un client et un serveur ou entre un client et un autre client. Ce type de pare-feu ne permet pas à un attaquant d'accéder directement à la machine attaquée, ce qui donne une forte protection supplémentaire. Dans le second cas, le pare-feu détecte les flots applicatifs et les interrompt ou non suivant les éléments filtrés. Dans tous les cas, il faut utiliser des filtres plus ou moins puissants.

Quelques ports réservés TCP		
N° de port	**Service**	**Rôle**
1	tcpmux	Multiplexeur de service TCP
3	compressnet	Utilitaire de compression
7	echo	Fonction écho
9	discard	Fonction d'élimination
11	users	Utilisateurs
13	daytime	Jour et heure
15	netstat	État du réseau
20	ftp-data	Données du protocole FTP
21	ftp	Protocole FTP
23	telnet	Protocole Telnet
25	smtp	Protocole SMTP
37	heure	Serveur heure
42	name	Serveur nom d'hôte
43	whols	Nom NIC
53	domain	Serveur DNS
77	rje	Protocole RJE
79	finger	Finger
80	http	Service WWW
87	link	Liaison TTY
103	X400	Messagerie X.400
109	pop	Protocole POP
144	news	Service News
158	tcprepo	Répertoire TCP
Quelques ports réservés UDP		
7	echo	Service écho
9	rejet	Service de rejet
53	dsn	Serveur de nom de domaine
67	dhcp	Serveur de configuration DHCP
68	dhcp	Client de configuration DHCP

TABLEAU 39.2 • Principaux ports TCP et UDP

Port accepté	Adresse IP
21	*
23	*
25	Adresse réseau C à adresse réseau B
43	*
69	*
79	*

TABLEAU 39.3 • Composition d'un pare-feu classique

Les filtres

Comme expliqué précédemment, les filtres sont essentiellement appliqués sur les numéros de port. La gestion de ces numéros de port n'est toutefois pas simple. En effet, de plus en plus de ports sont dynamiques. Avec ces ports, l'émetteur envoie une demande sur le port standard, mais le récepteur choisit un nouveau port disponible pour effectuer la communication. Par exemple, l'application RPC (Remote Procedure Call) affecte dynamiquement les numéros de port. La plupart des applications P2P (Peer-to-Peer) ou de signalisation de la téléphonie sont également dynamiques.

L'affectation dynamique de port peut être contrôlée par un pare-feu qui se comporte astucieusement. La communication peut ainsi être suivie à la trace, et il est possible de découvrir la nouvelle valeur du port lors du retour de la demande de transmission d'un message TCP. À l'arrivée de la réponse indiquant le nouveau port, il faut détecter le numéro du port qui remplace le port standard. Un cas beaucoup plus complexe est possible, dans lequel l'émetteur et le récepteur se mettent directement d'accord sur un numéro de port. Dans ce cas, le pare-feu ne peut détecter la communication, sauf si tous les ports sont bloqués. C'est la raison essentielle pour laquelle les pare-feu n'acceptent que des communications déterminées à l'avance.

Cette solution de filtrage et de reconnaissance des ports dynamiques n'est toutefois pas suffisante, car il est toujours possible pour un pirate de transporter ses propres données à l'intérieur d'une application standard sur un port ouvert. Par exemple, un tunnel peut être réalisé sur le port 80, qui gère le protocole HTTP. À l'intérieur de l'application HTTP, un flot de paquets d'une autre application peut passer. Le pare-feu voit entrer une application HTTP, qui, en réalité, délivre des paquets d'une autre application.

Une entreprise ne peut pas bloquer tous les ports, sans quoi ses applications ne pourraient plus se dérouler. On peut bien sûr essayer d'ajouter d'autres facteurs de détection, comme l'appartenance à des groupes d'adresses IP connues, c'est-à-dire à des ensembles d'adresses IP qui ont été définies à l'avance. De nouveau, l'emprunt d'une adresse connue est assez facile à mettre en œuvre. De plus, les attaques les plus dangereuses s'effectuent par des ports qu'il est impossible de bloquer, comme le port DNS. Une des attaques les plus dangereuses s'effectue par un tunnel sur le port DNS. Encore faut-il que la machine réseau de l'entreprise qui gère le DNS ait des faiblesses pour que le tunnel puisse se terminer et que l'application pirate s'exprime dans l'entreprise. Nous verrons à la section suivante comment il est possible de renforcer la sécurité des pare-feu.

Pour sécuriser l'accès à un réseau d'entreprise, une solution beaucoup plus puissante consiste à filtrer non plus aux niveaux 3 ou 4 (adresse IP ou adresse de port) mais au niveau applicatif. Cela s'appelle un filtre applicatif. L'idée est de reconnaître directement sur le flot de paquets l'identité de l'application plutôt que de se fier à des numéros de port. Cette solution permet d'identifier une application insérée dans une autre et de reconnaître les applications sur des ports non conformes. La difficulté avec ce type de filtre réside dans la mise à jour des filtres chaque fois qu'une nouvelle application apparaît. Le pare-feu muni d'un tel filtre applicatif peut toutefois interdire toute application non reconnue, ce qui permet de rester à un niveau de sécurité élevé.

La sécurité autour du pare-feu

Comme nous l'avons vu, le pare-feu vise à filtrer les flots de paquets sans empêcher le passage des flots utiles à l'entreprise, flots que peut essayer d'utiliser un pirate. La structure de l'entreprise peut être conçue de différentes façons. Deux solutions générales sont mises en œuvre. La première est illustrée à la figure 39.33, et la seconde à la figure 39.34.

Dans le premier cas, la communication, après avoir traversé le pare-feu, se dirige au travers du réseau d'entreprise vers le poste de travail de l'utilisateur. Dans ce cas, il faut que les postes de travail de l'utilisateur soient des machines sécurisées afin d'empêcher les flots pirates qui auraient réussi à passer le pare-feu d'entrer dans des failles du système de la station. Comme cette solution est très difficile à sécuriser, puisqu'elle dépend de l'ensemble des utilisateurs d'une entreprise, la plupart des architectes réseau préfèrent mettre en entrée de réseau une machine sécurisée, que l'on appelle machine bastion *(voir figure 39.34)*.

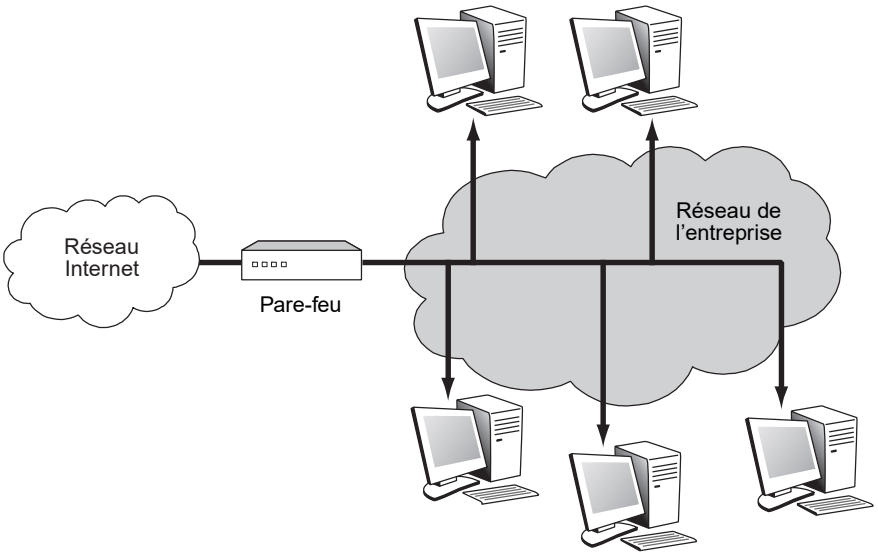


Figure 39.33

Place d'un pare-feu dans l'infrastructure réseau

La machine bastion apporte quelques difficultés supplémentaires de gestion. En effet, elle prend en charge l'ouverture et la fermeture des communications d'un utilisateur avec l'extérieur. Par exemple, un client avec son navigateur ne peut plus accéder à un serveur externe puisque la machine bastion l'arrête automatiquement. Le bastion doit être équipé d'un serveur proxy, et chaque navigateur être configuré pour utiliser le proxy. La communication se fait donc en deux temps. L'utilisateur communique avec son proxy, et celui-ci ouvre une communication avec le serveur distant. Lorsqu'une page parvient au proxy, ce dernier peut la distribuer au client. Le bastion peut d'ailleurs servir de cache pour les pages standards utilisées par une entreprise.

Le défaut de cette dernière architecture provient de sa relative lourdeur, puisqu'il est demandé à une machine spécifique d'effectuer le travail réseau pour toutes les machines de l'entreprise. De plus, la sécurité de toute l'entreprise peut être menacée si l'ordinateur

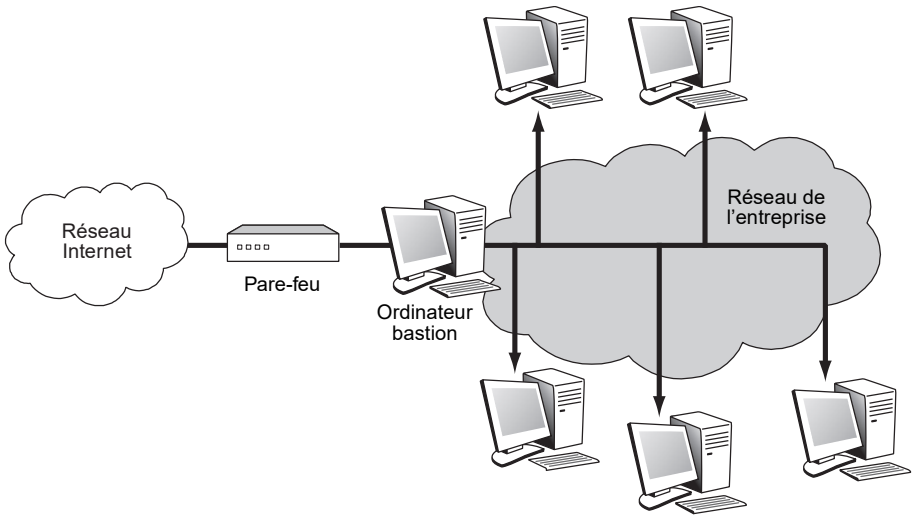


Figure 39.34

Pare-feu associé à une machine bastion

bastion n'est pas parfaitement sécurisé, car un pirate externe peut avoir accès à l'ensemble des ressources de l'entreprise. De fait, l'architecture de sécurité peut s'avérer plus complexe lorsqu'un ordinateur bastion est mis en place.

La figure 39.35 illustre quelques-unes des architectures de sécurité qui peuvent être mises en place.

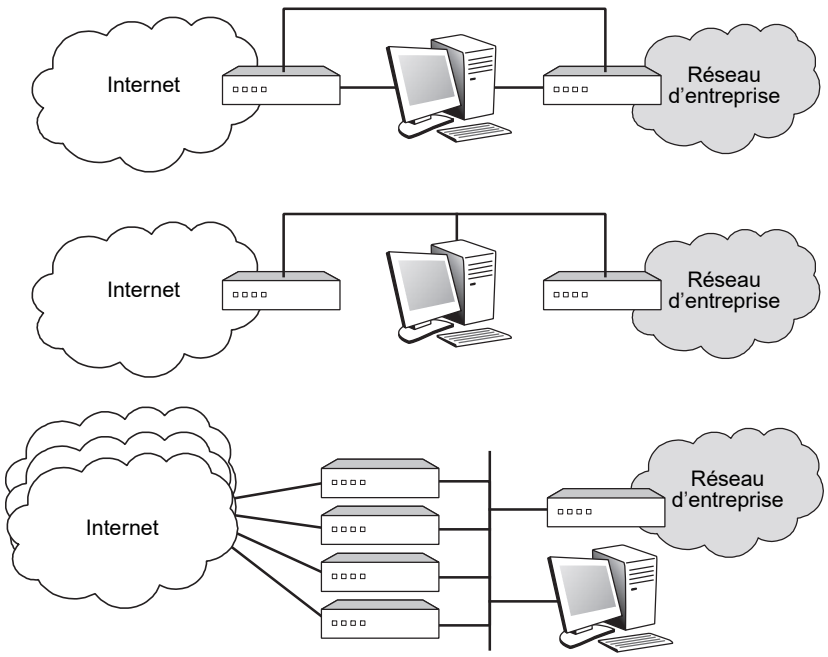


Figure 39.35

Architectures de sécurité avec machine bastion

La partie supérieure de la figure représente une organisation assez classique, dans laquelle l'ordinateur bastion est protégé des deux côtés par des pare-feu, pour filtrer aussi bien ce qui arrive de l'entreprise que ce qui arrive de l'extérieur. Le schéma montre deux pare-feu. Il est possible d'utiliser un seul pare-feu connecté à l'ordinateur bastion. Il est aussi possible de mettre en place manuellement une connexion directe entre les deux pare-feu pour effectuer des tests et des mises au point.

La deuxième partie de la figure est assez semblable à la précédente. Elle montre toutefois une organisation un peu différente, utilisant un réseau local pour relier les deux pare-feu et l'ordinateur bastion. La troisième partie de la figure montre une architecture encore plus complexe, dans laquelle une entreprise peut accéder à plusieurs opérateurs simultanément. Dans ce cas, un pirate peut entrer dans le réseau d'un opérateur en provenance d'un autre opérateur en passant par la passerelle d'une entreprise. Là, le piratage ne vise pas l'entreprise mais une autre entreprise, située sur le réseau de l'opérateur piraté. Pour sécuriser ce passage, l'ordinateur bastion doit de nouveau jouer le rôle de proxy, empêchant le passage direct.

Conclusion

La sécurité dans Internet est un problème complexe pour la simple raison qu'elle n'a pas été introduite en même temps que les protocoles de base. Pour arriver à vendre des produits rapidement, les équipementiers ont laissé la sécurité de côté en pensant pouvoir facilement l'ajouter par la suite. En réalité, l'effort à faire pour ajouter les éléments de sécurité dans un environnement qui n'a pas été conçu pour cela pose de nombreux problèmes, dont les utilisateurs prennent conscience peu à peu.

Des efforts énormes sont déployés en ce sens depuis une dizaine d'années. Toutefois, même si l'on dispose maintenant de toute une batterie d'outils pour assurer la sécurité d'un réseau IP, ils ne sont généralement pas faciles à utiliser.

Partie XII

Les applications

Lorsqu'on construit un réseau, c'est pour y écouler du trafic. Ce trafic provient des applications qui se déroulent sur le pourtour du réseau. Ces applications sont en nombre infini, et notre propos n'est pas d'entrer dans leur détail, ni même de les présenter. Cette partie se focalise sur le comportement que doit adopter le réseau pour que telle ou telle application s'y exécute à la satisfaction du client.

Dans le monde des réseaux, les applications se classent en deux grandes branches, les applications rigides et les applications élastiques. Le mot rigide indique que l'on ne peut pas modifier les caractéristiques définissant le flot de l'application sans risquer de voir cette dernière ne plus s'exécuter. L'exemple le plus classique d'application rigide est la parole téléphonique, puisque le flot des octets composant la parole doit arriver au destinataire à des instants précis et avant un temps limite très court. Si cette rigidité n'est pas respectée par le réseau, l'application ne peut se dérouler convenablement. Les contraintes qui en découlent sur le réseau sont fortes, et tous les réseaux ne peuvent y satisfaire.

Au contraire d'une application rigide, une application élastique accepte une déformation de son flot. Des paquets peuvent arriver en retard sans que l'application soit compromise. Les données informatiques font partie de cette catégorie. Lorsqu'on effectue, par exemple, un transfert de fichier, il n'est pas nécessaire que les paquets arrivent à la microseconde près, et quelques centaines de millisecondes, voire une seconde, suffisent. Les réseaux n'ont dès lors pas besoin de contrôles aussi puissants que lors du transport d'applications rigides.

Nous nous intéressons aux applications élastiques au chapitre 40 et aux applications rigides au chapitre 41. Ce dernier est consacré à la parole téléphonique, une des applications les plus rigides que l'on puisse trouver. Le chapitre 42 est dévolu aux applications multimédias, qui intègrent à la fois la voix, la vidéo et les données et posent de ce fait des problèmes supplémentaires. Le chapitre 43 examine les applications développées pour les réseaux de mobiles. Ces applications sont, pour certaines, élastiques, comme les transferts de données, et, pour d'autres, rigides, comme les applications dérivées de la parole téléphonique.