Chapitre 2

Authentification

Refik Molva et Yves Roudier[1]

1. Introduction

Ce chapitre traite des diverses techniques par lesquelles une entité peut obtenir une assurance sur l'identité de son interlocuteur dans un environnement potentiellement hostile où des entités malveillantes peuvent tenter d'usurper l'identité des entités légitimes. Ces techniques sont fréquemment appelées identification, authentification d'entité ou vérification d'identité. L'authentification d'entité ne vise pas à protéger un message particulier ni à fournir une preuve irrévocable sur un laps de temps conséquent comme dans le cas de l'authentification de message. L'authentification d'entité, qui sera seule traitée dans ce chapitre où elle sera simplement appelée authentification, corrobore l'identité d'un correspondant en temps réel. A ce titre, le processus d'authentification accepte ou rejette immédiatement les preuves d'identité du correspondant pour cet échange particulier. On distingue parfois l'identification de l'authentification pour indiquer un schéma dans lequel une entité décline juste une identité sans apporter de preuve la corroborant, mais ces deux termes sont employés comme synonymes dans ce chapitre. Tout schéma d'authentification suppose au moins deux parties : un demandeur, qui présente une identité, et un vérificateur, qui s'assure de sa validité.

[1]EURECOM

1.1. *Objectifs*

La mise au point d'un protocole d'authentification vise les objectifs suivants :

- assurer que si une entité A s'authentifie avec succès auprès d'une entité B, alors cette dernière accepte l'identité fournie par A à la fin de l'exécution du protocole.
- empêcher une entité B de réutiliser les informations échangées avec un interlocuteur A pour se faire passer pour A auprès d'une troisième entité C.
- rendre négligeable la probabilité qu'une entité C exécutant le protocole en usurpant l'identité de A puisse être acceptée par B (déguisement).
- assurer les points précédents même si un grand nombre d'authentifications entre A et B ont été observées par C, ou dans le cas où C sait s'authentifier avec A et/ou B.

Un des premiers buts de l'authentification est de permettre le contrôle d'accès dans le cas où les privilèges sont accordés en fonction d'une identité. Le suivi de l'utilisation de ressources est une autre motivation très importante de l'authentification : de nombreuses applications s'en servent par exemple pour la facturation. L'authentification est aussi un besoin inhérent aux protocoles d'établissement de clés authentifiés.

1.2. *Propriétés*

Un schéma d'authentification doit être étudié en prenant en compte les garanties de sécurité employées, le coût de l'authentification, et les parties impliquées.

Les garanties de sécurité sont les preuves qui permettent au vérificateur de corroborer l'identité affichée par un demandeur. Un schéma d'authentification doit d'abord être évalué sur la nature de ces garanties qui peuvent reposer sur : 1) la localisation du demandeur, 2) ce que sait le demandeur, 3) ce que possède le demandeur, ou 4) ce qu'est le demandeur. Certaines garanties peuvent être prouvables. Dans le deuxième cas, où un secret est utilisé comme garantie de sécurité, sa localisation et son stockage sont aussi importants pour évaluer la sécurité du schéma d'authentification.

Le coût d'un schéma d'authentification est un autre paramètre important. Il peut être évalué par deux critères : d'une part, la puissance de calcul nécessaire, c'est à dire le nombre d'opérations requises pour exécuter le protocole ; d'autre part, le coût en communications , qu'on peut définir comme le nombre et la taille des messages nécessaires. Le temps que prend l'authentification découle

également de ce coût et est important du point de vue d'un utilisateur qui attend d'être authentifié.

Les parties impliquées dans une authentification n'ont pas le même rôle mais certains schémas d'authentification offrent une réciprocité de l'authentification : on parle d'authentification unilatérale quand l'une des deux parties seulement prouve son identité et d'authentification mutuelle quand le protocole permet à chacune des deux parties d'obtenir une preuve de l'identité de son interlocuteur. Dans certains cas, l'exécution d'un protocole d'authentification peut nécessiter l'utilisation d'une entité supplémentaire comme une tierce partie de confiance, ou encore un répertoire de certificats. Il faut aussi évaluer la nature de la confiance placée en chacune des parties, demandeur, vérificateur, tierce partie : en particulier, y a-t-il secret partagé, connaissance de la clé publique du demandeur, etc.

Une première classification des techniques d'authentification permet de dégager quatre grandes catégories basées sur la nature de la sécurité qu'elles mettent en œuvre :

- techniques d'authentification faible, exploitant des informations de taille limitée et/ou non aléatoires ;
- techniques d'authentification forte basées sur des secrets cryptographiques ;
- techniques d'authentification forte basées sur des dispositifs matériels ;
- techniques d'authentification biométrique, directement liées aux caractéristiques physiologiques ou aux traits comportementaux d'un individu.

2. Authentification faible

On dit d'un schéma d'authentification qu'il offre une authentification faible dans le cas où il repose sur des preuves de taille limitée et/ou non aléatoires. Cette section présente trois types de schémas d'authentification faibles et leurs problèmes : l'authentification à base d'adresse, l'authentification par mot de passe fixe et l'authentification par mot de passe variable.

2.1. *Authentification à base d'adresse*

2.1.1. *Principe*

Le principe général de l'authentification à base d'adresse est que chaque station d'un réseau stocke des informations qui spécifient quelles autres machines ont accès à ses ressources. L'authentification s'appuie sur l'adresse réseau dont proviennent les paquets reçus et pas sur un secret quelconque. Par exemple, un

utilisateur peut être autorisé à copier des fichiers à distance, depuis une autre machine.

Deux schémas d'authentification par adresse sont possibles :
- une station A liste les adresses réseau des machines "équivalentes". Si la station B se trouve dans cette liste, tout compte ouvert sur B sera équivalent au compte de même nom sur A. Les requêtes provenant du compte sur la machine B auront tous les droits accordés au compte sur A.
- la station A possède une liste de tuples <adresse, compte distant, compte local>. Une requête provenant du compte distant sur la machine à l'adresse donnée sera honorée avec les mêmes droits d'exécution que ceux laissés au compte local.

2.1.2. *Exemple en Unix*

La technique d'authentification par adresse a été adoptée assez tôt dans le développement des réseaux, aussi bien en Unix qu'en VMS. Sous Unix, les Berkeley rtools permettent de se connecter avec une authentification de ce type et deux schémas d'authentification par adresse sont disponibles : global : un fichier /etc/hosts.equiv est défini pour tous les comptes disponibles sur une machine. Suivant le premier schéma défini plus haut, ce fichier contient la liste des machines qui possèdent des comptes identiques (mis à part pour le compte root). personnel : chaque utilisateur définit un fichier .rhosts dans son répertoire dans lequel il indique une liste de paires <machine, compte> autorisés à accéder au compte.

2.1.3. *Attaques possibles*

L'authentification à base d'adresse n'est pas sensible aux écoutes, mais est par contre sujette à deux types d'attaque : le rebond et l'usurpation d'adresse.

Le rebond survient quand un intrus prend pied sur un système et quand ce système est autorisé à effectuer des opérations sur d'autres machines. Un utilisateur peut avoir défini des fichiers d'authentification croisée entre le système dans lequel l'intrus s'est introduit et une autre machine par commodité afin par exemple, de ne pas rentrer un mot de passe à chaque connexion. L'intrus pourra grâce à ces fichiers localiser une cible potentielle pour une nouvelle attaque : en progressant ainsi, par rebond de machine en machine, l'intrus peut donc affecter tous les systèmes auxquels un utilisateur a accès.

Le mécanisme de l'authentification à base d'adresse suppose que l'identité de la source peut être établie à partir de l'adresse source du trafic reçu. Cette condition n'est pas toujours réalisée : il est souvent facile à un intrus de prétendre envoyer un paquet depuis une adresse source autorisée. L'organisation du réseau est largement en cause dans cette attaque. On ne peut par exemple jamais s'en affranchir à l'intérieur d'un réseau local à base de bus Ethernet. Par contre, avec d'autres technologies, il serait possible de vérifier l'adresse source portée par un paquet : dans un anneau Token Ring, une station peut empêcher la propagation des messages qui portent son adresse ; un routeur pourrait vérifier que le paquet qui arrive sur un lien donné porte bien une adresse source vraisemblable. En réalité, ce genre de vérification n'est jamais effectué, ce qui rend possible les attaques.

Au vu de la technique d'usurpation d'adresse, il peut sembler que seules les attaques en aveugle soient possible. En fait, il est possible à un intrus de récupérer des information du système attaqué. S'il se trouve dans le réseau local de la machine authentifiée ou sur le chemin entre la machine authentifiée et le système visé, il lui suffit par exemple d'écouter le trafic à destination de la machine authentifiée.

2.2. *Mots de passe*

2.2.1. *Principe et Faiblesse des Mots de Passe*

Les mots de passe constituent la plus vieille technique d'authentification unidirectionnelle : pour accéder aux ressources d'un système, un utilisateur entre son identité, un mot de passe, et la ressource à laquelle il veut accéder (implicitement ou explicitement). Un mot de passe est donc un secret partagé entre l'utilisateur et le système auquel il s'authentifie : prouver qu'il connaît ce mot de passe prouve que son identité est correcte.

La faiblesse principale des mots de passe vient de ce qu'ils ne remplissent plus leur rôle dès qu'ils sont dévoilés et ne constituent donc plus un secret. Un intrus peut découvrir un mot de passe de plusieurs manières :

- en regarder un utilisateur taper son mot de passe au clavier : la seule solution est de cacher le clavier.
- en surveillant l'écran d'un utilisateur : le mot de passe peut en effet être affiché en clair lorsqu'il est entré sur certains systèmes.
- en essayant de se connecter à un compte utilisateur en essayant exhaustivement tous les mots de passe possibles.
- en récupérant un fichier de mots de passe. Celui-ci peut être en clair, auquel cas l'intrus n'a plus aucun travail, mais plus souvent, de nom-

breux systèmes les conservent dans des fichiers publics chiffrés avec un algorithme à sens unique. Malheureusement, il est bien connu que les utilisateurs ont tendance à utiliser des noms communs, des noms propres ou des nombres qui ont un sens pour eux : si l'accès aux fichiers contenant les mots de passe n'est pas restreint, il est possible de retrouver ces mots de passe en comparant leur chiffrement avec le chiffrement de tous les mots d'un dictionnaire par exemple.
- en écoutant le mot de passe lorsqu'il est transmis en clair sur un réseau entre deux machines, par exemple à l'occasion d'une connexion.

Les PINs (Personal Identification Number - ou nombre d'identification personnelle) rentrent aussi dans la catégorie des mots de passe. Pour des raisons de commodité comme pour des raisons historiques, les PINs sont entièrement constitués de chiffres et d'une taille faible par rapport aux mots de passe classiques, typiquement 4 à 6 chiffres. Les PINs sont en général employés en association avec un dispositif d'authentification matériel portable, comme par exemple une carte magnétique ou une carte à puce. L'utilisateur doit fournir son PIN pour prouver son identité en tant que possesseur du dispositif matériel à chaque fois qu'il l'utilise. Le PIN fournit ainsi un second niveau de sécurité si le dispositif matériel est volé.

Pour empêcher des recherches exhaustives, l'utilisation d'un PIN doit être accompagnée de mesures supplémentaires : par exemple, certains distributeurs de billets confisquent la carte bancaire ou la verrouillent si trois PINs incorrects ont été entrés successivement. Dans un système en ligne, une identité peut être vérifiée en comparant le PIN entré par l'utilisateur au PIN stocké dans une base de données pour l'identité fournie. Dans un système hors ligne, c'est le dispositif matériel qui doit conserver les informations permettant la vérification de l'identité. On parle alors d'authentification à deux phases : l'utilisateur s'authentifie auprès du dispositif matériel, puis ce dispositif s'authentifie auprès du système. Si l'utilisateur peut choisir son PIN, celui-ci sera chiffré avec une clé secrète connue des terminaux et conservée sous cette forme sur la carte ; au contraire, si l'utilisateur ne le choisit pas, le PIN peut simplement consister en une fonction d'une clé secrète connue des terminaux ainsi que de l'identité conservée sur la carte.

Autre type de mot de passe, la technique des clés dérivées à partir de mots de passe [J. 93] [LAB 93] consiste à transformer un mot de passe en une clé cryptographique. Il est bien connu que ces clés sont des nombres si grands qu'un humain ne peut s'en rappeler, alors qu'un mot de passe offre une chaîne de caractères mémorisable. Par exemple, pour former une clé secrète DES, il est possible d'appliquer une fonction de hachage à sens unique sur un mot de passe fourni par l'utilisateur et de conserver 56 bits du résultat de cette opération. Cette clé secrète peut ensuite être utilisée pour sécuriser les communications entre la machine de l'utilisateur et un système connaissant sa clé secrète. Il est

plus difficile et coûteux de représenter des clés asymétriques à partir d'un mot de passe car les clés générées doivent en plus respecter les propriétés imposées par l'algorithme à clé publique. L'utilisation de ce type de clé reste cependant sujette à certaines précautions : l'entropie du mot de passe fourni par l'utilisateur doit être suffisante. Ce type de clé n'est pas non plus destiné au chiffrement de données.

2.2.2. *Attaques*

Les systèmes d'authentification par mot de passe sont sujet à trois types d'attaques : le rejeu de mots de passe, la recherche exhaustive de mots de passe, les attaques par dictionnaire.

Si un adversaire a connaissance du mot de passe d'un utilisateur en l'observant ou par une écoute du réseau, il peut usurper l'identité de cet utilisateur en réemployant (ou en rejouant) ce même mot de passe autant de fois qu'il le souhaite auprès du système cible. C'est la raison pour laquelle les mots de passe fixes ne sont utilisables que sur des transmissions sécurisées et pas pour des transmissions en texte clair.

Le cas le plus naïf d'attaque par recherche exhaustive est celui d'un attaquant en ligne, essayant tous les mots de passe les uns après les autres : ce type d'attaque peut être évité en choisissant des mots de passe suffisamment grands, en limitant le nombre de tentative d'authentifications infructueuses et en ralentissant l'authentification. Au contraire, les attaques par recherche exhaustive hors ligne, dans lesquelles l'adversaire possède le fichier de mots de passe chiffré, sont plus sérieuses : il est théoriquement possible d'essayer exhaustivement tous les mots de passe possibles et pour chacun, de comparer son hachage à la valeur du hachage du mot de passe effectif de l'utilisateur stockée dans le fichier. La faisabilité de l'attaque dépend essentiellement du nombre de mots de passe disponibles et du temps nécessaire pour en tester un. Il est à noter que la recherche exhaustive est une activité parallélisable.

Pour améliorer la probabilité de succès par rapport à une attaque exhaustive, le troisième type d'attaque consiste à tester les mots de passe ayant une plus grande probabilité d'être utilisés. Les utilisateurs choisissent généralement des mots de passe parmi un nombre relativement réduit de mots : ainsi, un dictionnaire de 150.000 mots contiendra une bonne part des mots de passe des utilisateurs de n'importe quel système, alors même qu'un dictionnaire de taille importante fait 250.000 mots et qu'à un mot de passe de 8 caractères peuvent correspondre 268 combinaisons uniquement composées de caractères alphabétiques. Pour plus d'efficacité, une attaque par dictionnaire hors ligne peut s'appuyer non pas sur un dictionnaire en texte clair, mais sur un dic-

tionnaire préalablement haché. Il faut noter que ce type d'attaque ne permet généralement pas de trouver le mot de passe d'un utilisateur particulier ; par contre, ils permet de découvrir de nombreux mots de passe faibles dans un système. Des programmes que l'on qualifie de "crack" sont normalement employés pour effectuer des recherches de ce type et sont utilisés aussi bien par les attaquants que par les administrateurs systèmes pour tester la robustesse de mots de passe.

Techniques Les différentes techniques de mot de passe existantes peuvent se classer en fonction des moyens selon lesquels les informations permettant la vérification des mots de passe sont conservés dans le système et par la méthode de vérification. La résistance aux attaques citées plus haut dépend des choix de conception concernant le stockage des mots de passe, le choix et la mémorisation des mots de passe, et la reconnaissance des mots de passe.

Concernant le stockage, l'utilisation d'un fichier de mots de passe en clair constitue une première approche : les mots de passe sont conservés dans un fichier système protégé en lecture et en écriture (le contrôle d'accès étant par exemple assuré par le système d'exploitation) et permettent la comparaison avec le mot de passe entré par un utilisateur. Un inconvénient de cette technique non cryptographique est qu'elle ne protège pas contre les super-utilisateurs. L'utilisation de fichier de mots de passe encrypté représente une approche plus sûre : au lieu de conserver les mots de passe en texte clair, on stocke le résultat de l'application d'une fonction de hachage à sens unique sur chaque mot de passe dans un fichier qui doit toujours être protégé en écriture. Les mots de passe salés [R.] offrent un supplément de sécurité à la technique précédente : on ajoute à chaque mot de passe, lorsqu'il est entré initialement, une chaîne de t bits aléatoires, appelée "sel", avant d'appliquer une fonction de hachage à sens unique. Le mot de passe haché et le "sel" sont enregistrés dans le fichier de mots de passe. Le salage d'un mot de passe n'augmente pas la difficulté de recherche par une attaque exhaustive puisque le "sel" est conservé en clair dans le fichier de mots de passe ; par contre, elle augmente le coût d'une attaque simultanée sur plusieurs mots de passe, puisqu'il faut prévoir 2t variations de chaque mot de passe.

Certains systèmes imposent des règles aux utilisateurs dans le choix de leur mot de passe [FED 85] [DEF 85] pour leur éviter de sélectionner un mot de passe faible et de rendre le système vulnérable aux attaques par dictionnaire. Des règles typiques imposent une longueur minimale au mot de passe (8 à 12 caractères) et la présence d'un certain nombre de caractères majuscules, numériques ou non alphanumériques, ou encore la vérification de l'absence du mot de passe dans un dictionnaire ou de l'absence de tout lien avec l'identité de l'utilisateur. L'objectif de ces règles est essentiellement d'accroître l'entropie des mots de passe au-delà des possibilités d'attaque des attaques exhaustives. Pour permettre une plus grande entropie malgré les capacités de mémorisation

humaines, on peut remplacer les mots de passe par des phrases de passe. La phrase complète entrée par l'utilisateur est hachée en une valeur de taille fixe qui joue le rôle d'un mot de passe. L'idée qui sous-tend cette technique est qu'une phrase est plus simple à mémoriser qu'une suite de caractères aléatoire. Un utilisateur peut aussi appliquer cette idée aux mots de passe classiques en mémorisant non pas un mot de passe, mais une acrostiche, c'est-à-dire une phrase dont les premières lettres de chaque mot constituent le mot de passe.

Un mot de passe n'est pas forcément reconnu pour une durée illimitée. Sur de nombreux systèmes, les mots de passe expirent après une certaine durée de validité et l'utilisateur doit en fournir un nouveau. Cette durée doit être d'autant plus réduite si l'entropie du mot de passe est faible.

Il peut aussi être utile de ralentir la reconnaissance du mot de passe : le but est ici de ralentir les attaques exhaustives en augmentant le coût en calcul de la fonction de vérification des mots de passe. Le ralentissement doit être suffisant pour limiter les attaques sans pour autant gêner l'utilisateur.

2.2.3. *Exemple : gestion des mots de passe dans UNIX*

Le système d'exploitation UNIX fournit un bon exemple de système à mot de passe fixe. L'algorithme décrit ci-dessous est connu sous le nom de *crypt* en UNIX.

Le fichier de mots de passe UNIX /etc/passwd conserve le résultat du hachage du mot de passe de chaque utilisateur. Chaque mot de passe sert de clé pour chiffrer un texte clair connu, contenant 64 bits remplis de zéros : cet algorithme fournit une fonction de hachage à sens unique du mot de passe car celui-ci est connu du seul utilisateur.

L'algorithme de chiffrement utilisé est une variante de DES, dans laquelle le chiffrement est itéré 25 fois. La fonction d'expansion de DES qui appliquée sur 32 bits en fournit 48 est modifiée avec 12 bits générés aléatoirement avec l'horloge système pour produire $212 = 4096$ variations supplémentaires pour chaque mot de passe. Cette modification constitue une technique de salage particulière, destinée d'une part, à compliquer les attaques par dictionnaires, et d'autre part, à interdire l'emploi de matériels de chiffrement/déchiffrement DES disponibles dans le commerce. Deux utilisateurs avec le même mot de passe n'auront pas la même entrée dans le fichier de mots de passe.

Le mot de passe est tout d'abord tronqué à 8 caractères sur 7 bits qui forment une clé DES de 56 bits (éventuellement rempli avec des 0 si l'utilisateur a rentré moins de 8 caractères). Le texte clair de 64 bits à 0 est chiffré avec cette clé, et le résultat du chiffrement est utilisé comme nouveau texte à chiffrer, et

ce 25 fois. Le hachage du mot de passe est le mot de 11 caractères résultant de l'addition de la sortie finale sur 64 bits et de 12 bits de salage. Ce résultat est finalement stocké dans le fichier /etc/passwd.

2.3. *Mots de passe variables*

Les mots de passe variables constituent une évolution des mots de passe fixe vers un schéma question-réponse visant à résoudre les problèmes de rejeu. Dans un système à mots de passe variables, chaque mot de passe échangé en texte clair n'est valable qu'à une seule et unique utilisation : ce type de technique met l'utilisateur à l'abri des adversaires passifs effectuant seulement des écoutes et utilisant les mots de passe captés pour effectuer une attaque par rejeu. Par contre, un adversaire actif qui intercepte et bloque les communications et envoie ses propres données à la place peut déjouer ce type d'authentification.

On peut cataloguer les schémas de mots de passe variables en trois types :
- mots de passe variables basés sur des listes partagées : l'utilisateur reçoit du système une liste de mots de passe secrets à usage unique ou une table contenant des paires de questions-réponses.
- mots de passe variables mis à jour en séquence : un mot de passe (secret) est partagé au départ par le système et l'utilisateur ; à chaque authentification, l'utilisateur envoie au système un nouveau mot de passe chiffré par une clé dérivée du mot de passe précédemment partagé.
- mots de passe variables basés sur une fonction de hachage à sens unique : l'utilisateur et le système partagent le résultat secret du hachage par une fonction à sens unique, itéré un certain nombre de fois, d'un mot de passe initial connu de l'utilisateur seul.

L'algorithme de Lamport [L.] est un exemple de ce dernier type de système de mots de passe variables. Il nécessite une phase d'initialisation préalable avant d'être employé pour l'authentification proprement dite. Lors de l'initialisation, l'utilisateur choisit un mot de passe m secret. Une constante t fixe le nombre d'authentification permises avec m. L'utilisateur calcule le hachage de m par une fonction à sens unique h, itéré t fois, et communique d'une manière qui en garantisse l'authenticité $h^t(m)$ au système, qui stocke ce message. Pour sa i-ème authentification après l'initialisation (voir Figure 2.1), l'utilisateur transmet au système auprès duquel il cherche à s'identifier un message portant son identité, i, et $h^{t-i}(m)$. Le système vérifie que i correspond bien à la i-ème tentative d'authentification (c'est-à-dire à la i-ème variation du mot de passe initial) pour l'utilisateur en question et que $h^{t-i}(mreu)$ est bien égal à $h^{t-i+1}(m)$ dont la valeur est connue par le système. Si la vérification est réussie, le système incrémente le nombre de tentatives réussies et conserve la valeur $h^{t-i}(m)$ reçue lors de cette authentification. Lorsque l'utilisateur arrive au nombre maximum

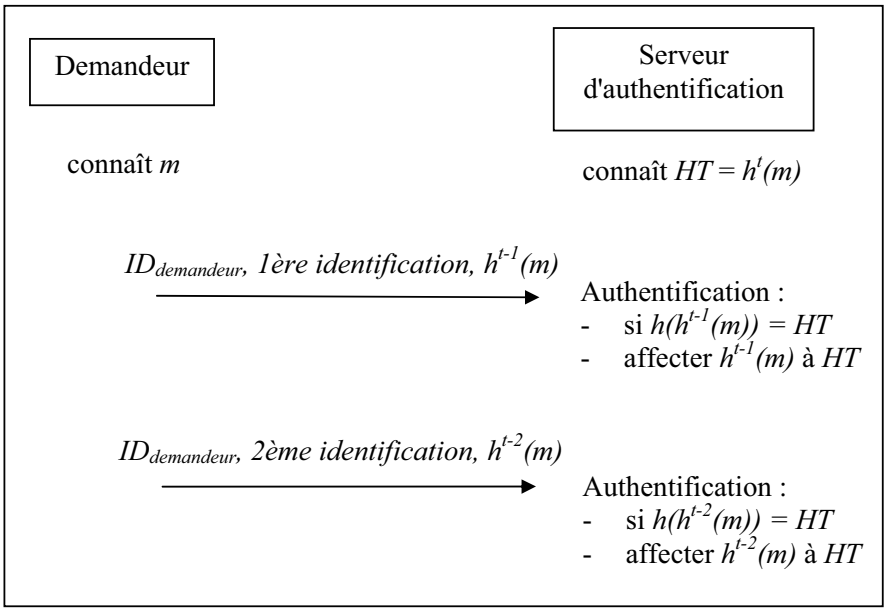

Figure 2.1. *Authentification par l'algorithme de Lamport*

t d'authentifications possibles, il doit réinitialiser le mécanisme avec un nouveau mot de passe afin d'éviter le rejeu possible.

3. Authentification forte

Lorsqu'une entité effectue une authentification forte auprès d'une autre entité, elle prouve qu'elle connaît un secret associé avec son identité déclarée. Cette preuve s'appuie sur l'utilisation de techniques cryptographiques telles que fonctions de hachage, cryptographie symétrique ou asymétrique : on parle de protocole cryptographique question-réponse. Un protocole **question-réponse** fonctionne d'après le principe suivant : L'entité B qui joue le rôle de vérificateur choisit de manière aléatoire une donnée, appelée **question**, qui est envoyée à l'entité A qui doit prouver son identité ; l'entité A à son tour applique à la question une opération cryptographique basée sur un secret qu'elle détient ; le résultat de cette opération, appelé **réponse**, est renvoyé à B pour fournir la preuve de l'identité de A. Le but principal des protocoles question-réponse est d'empêcher une famille d'attaques connue sous le nom de **rejeu**. Le rejeu décrit la retransmission par un intrus d'une réponse qui a déjà été utilisée entre deux

entités légitimes comme réponse à une question nouvelle dans le but d'usurper l'identité du demandeur.

Afin d'assurer la protection contre le rejeu, la réponse calculée par le demandeur A doit être différente à chaque exécution du protocole d'authentification. La technique permettant d'obtenir un paramètre qui varie dans le temps constitue un des éléments essentiels d'un protocole d'authentification. Le second élément important d'un protocole d'authentification concerne la manière dont le demandeur A calcule la réponse correspondant à la question posée par le vérificateur B. Plusieurs variantes de calcul existent en fonction de la méthode cryptographique utilisée. Une dernière caractéristique déterminante pour un protocole d'authentification est le schéma de communication entre les diverses parties qui sont le demandeur A, le vérificateur B et éventuellement une tierce partie ou un serveur d'authentification dont la présence est nécessaire dans certains cas.

La suite de ce paragraphe énumère les variantes concernant les trois caractéristiques de base d'un protocole d'authentification qui sont la génération d'un paramètre variable dans le temps, la méthode cryptographique pour le calcul de la réponse et le modèle de communication. Les principales familles de protocole correspondant aux combinaisons de ces caractéristiques sont ensuite présentées. L'authentification forte des utilisateurs humains nécessite une composante nouvelle sous forme d'un dispositif personnel. Une classification des dispositifs personnels d'authentification est fournie à la fin du paragraphe.

3.1. *Génération de paramètres variables dans le temps*

L'utilisation d'un paramètre variable dans le temps pour le calcul de la réponse permet à un protocole d'authentification de résister aux attaques de type "rejeu". Grâce au paramètre variable dans le temps, chaque exécution du protocole d'authentification entre deux entités utilise des messages différents et les attaques qui consistent à répondre à une question nouvelle par un message provenant d'une exécution antérieure du protocole deviennent infructueuses.

Il existe principalement trois techniques permettant de générer des paramètres variables dans le temps : les nombres aléatoires, les numéros de séquence et l'horodatage. D'une façon générale le vérificateur B assure la variabilité du paramètre directement (en générant un nombre aléatoire et en l'envoyant à l'entité authentifiée A) ou bien indirectement (en vérifiant le numéro de séquence par rapport à un compteur local ou le cachet d'horodatage par rapport à une horloge locale).

3.1.1. *Nombres Aléatoires*

Dans le contexte des protocoles d'authentification, le terme "nombres aléatoires" désigne une suite de nombres pseudo-aléatoires qui sont imprédictibles par un adversaire. Une manière simple d'obtenir des nombres aléatoires dans ce sens consiste à s'assurer l'absence de répétitions et à éviter qu'un même nombre soit utilisé deux fois.

Dans un protocole d'authentification utilisant les nombres aléatoires, le vérificateur B génère un nombre aléatoire N et l'envoie à A comme faisant partie de la question. Si B reçoit une réponse dont le calcul est basé sur N, B peut être sûre que cette réponse vient d'être tout récemment fabriquée par l'entité légitime A (qu'elle ne résulte pas du rejeu d'une réponse ancienne par un attaquant). En effet, d'après la propriété des nombres aléatoires, N n'est jamais apparu auparavant, d'où l'unicité de la réponse qui est directement liée à N. Dans cet échange, la fraîcheur de la réponse est directement liée à celle de la question et en assurant la fraîcheur des questions qu'elle génère, l'entité B peut garantir la fraîcheur des réponses qui lui sont fournies afin d'éviter le rejeu. Il apparaît clairement que l'absence de rejeu dépend aussi de la fiabilité du lien entre le nombre aléatoire contenu dans la question et la réponse qui résulte d'un calcul utilisant ce nombre ; ce lien doit être impossible à établir pour toute entité autre que A et B.

La "fraîcheur" ou le caractère récent de la réponse dans ce contexte désigne seulement le fait que la réponse a été générée après l'émission de la question. Pour des raisons pratiques indépendantes de la sécurité et afin de se prémunir contre d'eventuels erreurs de communication, un protocole d'authentification avec les nombres aléatoires peut limiter le délai entre l'émission de la question et la réception de la réponse en utilisant un mécanisme de temporisation localisé chez l'entité B.

Les protocoles d'authentification avec les nombres aléatoires nécessitent le maintien d'un état par l'entité B afin de mémoriser la valeur du nombre aléatoire utilisé dans la question. Toutefois le maintien de l'etat n'est pas nécessaire après la vérification de la réponse.

3.1.2. *Numéros de séquence*

Un numéro de séquence est un nombre associé à un message qui permet d'identifier ce message par rapport à un ensemble ou séquence de messages. Typiquement, deux messages qui portent le même numéro de séquence sont considérés identiques. L'association des numéros de séquence aux messages se fait d'après une politique prédéfinie. La politique la plus simple qui est utilisée

dans le cadre de la plupart des protocoles de communication consiste à allouer les numéros de séquence d'une façon incrémentale en commençant par la valeur zéro qui est associée au premier message de la séquence et en incrémentant le numéro par un à la génération de chaque nouveau message.

Dans le cas des protocoles d'authentification, les numéros de séquence permettent d'identifier les messages de chaque exécution du protocole afin de détecter le rejeu. L'entité A qui désire prouver son identité à l'entité B, envoie un message contenant un nouveau numéro de séquence et l'entité B vérifie l'absence de rejeu en validant le numéro de séquence contenu dans le message. Dans ce scénario, le messsage envoyé par l'entité A correspond à la réponse calculée pour une question implicite qui est constituée par la valeur du numéro de séquence. Comme pour la technique des nombres aléatoires, le lien entre le numéro de séquence et le message qui est envoyé à B doit être fiable et impossible à établir pour toute entité autre que A et B.

Afin de générer la même suite de numéros de séquence, chaque entité doit maintenir un compteur par entité correspondante. Les compteurs de deux entités qui communiquent entre elles doivent être synchronisés dans le sens qu'ils doivent être initialisés à la même valeur et incrémentés à chaque échange de message entre ces deux entités. De plus, entre A et B, il est nécessaire de maintenir un compteur chez A et un compteur chez B pour chaque sens de la communication, chaque sens de la communication correspondant à une séquence différente de messages.

Le maintien d'un état séparé par correspondant et par sens de communication constitue le désavantage principal de la technique des numéros de séquence. Le second problème lié à l'utilisation de cette technique est due à la perte de synchronisation entre deux entités : dans le cas d'une politique simple, si le message d'authentification portant le numéro n est perdu à cause d'une erreur de transmission, le message suivant, qui doit nécessairement porter le numéro n+1 afin d'éviter d'éventuels rejeux, sera refusé par le récepteur. Une procédure de resynchronisation doit être mise en œuvre afin de rétablir l'état de l'émetteur et du récepteur après chaque perte de synchronisation. Une telle procédure implique un coût et une vulnérabilité potentielle puisque un attaquant serait tenté de l'utiliser pour forcer un état inconhérent chez le récepteur du message d'authentification. Une autre solution au problème de perte de synchronisation consiste à utiliser une politique de validation moins restrictive basée sur une suite monotone croissante où un numéro de séquence est accepté s'il est supérier ou égal à la valeur du compteur local. Tout en résolvant le problème du rejet cité auparavant, cette nouvelle politique présente d'autres inconvénients :

- la représentation d'une séquence infinie dans un champ de taille limitée est un problème complexe et peut mener à une ambiguité concernant la validation du numéro de séquence ;
- la destruction intentionelle des messages ne peut pas être détectée.

3.1.3. *Horodatage*

La technique d'horodatage permet de marquer chaque message avec l'instant auquel le message a été généré ou émis. La représentation de cet instant se fait par un champ du message appelé cachet d'horodatage dont la précision varie d'après le contexte d'utilisation.

Dans le cas d'un protocole d'authentification utilisant l'horodatage, l'entité A qui désire s'authentifier auprès de l'entité B, lit la valeur du temps t sur son horloge locale, fabrique un message d'authentification en calculant une fonction secrète de t et envoie à B le résultat de ce calcul comme message d'authentification. A la réception de ce message l'entité B extrait la valeur de t à partir du message et le message d'authenitfication est accepté si t appartient à l'intervalle d'acceptation définie d'après l'horloge local de B. Comme pour les deux précédentes techniques, le lien entre t et le message d'authentification doit être fiable et impossible à établir pour toute entité autre que A et B.

L'intervalle d'acceptation est défini autour de l'instant courant affiché par l'horloge en tenant compte du délai de transmission et de propagation des messages. La définition de cet intervalle est un problème assez délicat surtout si le délai de transfert des messages présente une variance importante. Si l'intervalle est trop petit, des messages légitimes risquent d'être rejetés à tort et dans le cas d'un intervalle trop grand le rejeu devient possible.

La sécurité des protocoles d'authentification avec horodatage dépend principalement de l'existence d'une référence de temps fiable et commune aux deux entités impliquées dans le protocole. La solution pratique à ce besoin consiste à utiliser des horloges locales comme dans le scénario précédent. Cependant le décalage entre les horloges distribués est un problème inévitable et mène aux problèmes de rejet à tort et de rejeu définis à propos de l'intervalle d'acceptation. Un remède à ce problème serait d'utiliser un protocole spécialement conçu pour la synchronisation des horloges. Comme pour la synchronisation des compteurs de numéros de séquence, le protocole de synchronisation devrait lui-même être sécurisé avec au moins le service d'authentification afin d'éviter son utilisation intempestive par un attaquant, ce qui mène à un dilemme si l'on se borne à une méthode d'authentification avec horodatage pour ce dernier.

Il existe une solution aux problèmes de rejet à tort et de rejeu qui sont dus à l'inadéquation de l'intervalle, à des délais de transfert trop importants ou à des décalages entre les horloges qui consiste à maintenir un état en plus de l'horloge locale. Plusieurs alternatives peuvent être imaginés : un état global peut représenter l'ensemble des messages reçus pendant l'intervalle d'acceptation courant ou bien un état individuel par entité correspondante peut mémoriser le dernier cachet d'horodatage reçu de chaque correspondant. Dans tous les cas, ces améliorations alourdissent considérablement la technique d'horodatage par

l'introduction d'un état.

3.1.4. *Comparaison des trois techniques*

La technique d'horodatage est la plus simple à mettre en œuvre parmi les trois techniques de génération de paramètres variables dans le temps puisque les protocoles d'authentification utilisant l'horodatage s'exécutent en un seul message (contrairement à la techique des nombres aléatoires) et que l'horodatage ne nécessite pas le maintien d'un état (contrairement à la technique des numéros de séquence). Cependant le second avantage de cette technique disparaît dans le cas où le délai de transfert des messages ou le décalage des horloges deviennent importants. Par conséquent, la technique d'horodatage reste une solution de choix dans les environnements avec de faibles délais de communication comme les réseaux locaux et pour des applications de type client-serveur pour lesquelles le maintien d'état et les échanges multiples de messages sont prohibitifs.

La technique des numéros de séquence exige le maintien d'une information d'état par chaque entité et la taille de cette information est proportionnelle aux nombres d'entités correspondantes. Face aux pertes de messages cette technique pose aussi un besoin de synchronisation sans solution parfaite comme l'horodatage. Cependant dans le cas où le nombre d'entités communicantes est limité et la communication est fiable les numéros de séquence sont très appropriés car ils permettent de réaliser un protocole d'authentification simple qui s'exécute en un seul message tout en assurant l'absence de rejeu et de rejet à tort.

La technique des nombres aléatoires présente un avantage important sur les deux précédentes : elle ne repose par sur la synchronisation ni sur le maintien d'état. Tous les problèmes que nous avons rencontrés dans les deux premiers cas et qui découlent de ces deux besoins ne se posent pas dans le cas de cette technique. La mise en œuvre d'un protocole d'authentification basé sur les nombres aléatoires est donc très simple et les vulnérabilités des deux premières techniques qui sont dues à leur dépendance des mécanismes de synchronisation et de maintien d'état n'existent pas dans ce cas. Cette technique présente cependant un désavantage principal qui explique son utilisation relativement limitée : deux messages au moins sont nécessaires pour sa mise en œuvre dans un protocole d'authentification.

3.2. *Méthodes cryptographiques pour le calcul de la réponse*

Dans le paragraphe précédent nous avons mentionné à plusieurs reprises la nécessité du lien entre le paramètre variable dans le temps et le message d'authentification envoyé par le demandeur A vers le vérificateur B. La sécurité de ce lien constitue l'élément crucial d'un protocole d'authentification. Si l'on suppose que la possibilité du rejeu est éliminée grâce aux techniques présentées dans le paragraphe précédent, il reste encore la possibilité pour un attaquant de fabriquer un message d'authentification valide en utilisant la question explicite (nombre aléatoire) ou implicite (numéro de séquence ou cachet d'horodatage). Le principe des procotoles d'authentification forte est de garantir que seule le demandeur A est capable de fabriquer une réponse valide en réponse à la question (implicite ou explicite). Afin de restreindre cette capacité à la seule entité A, les protocoles existants utilisent des techniques cryptographiques pour permettre à l'entité à de lier d'une façon forte la question (nombre aléatoire, numéro de séquence ou cachet d'horodatage) à un secret connu seul par A. La sécurité de la technique cryptographique utilisée pour calculer ce lien détermine la garantie contre la possibilité de fabrication frauduleuse d'une réponse d'authentification.

Il existe deux familles de techniques cryptographiques pour assurer ce lien : les techniques symétriques où A et B partagent le secret qui est nécessaire aussi bien pour la génération que pour la vérification du message d'authentification et les techniques asymétriques où la connaissance du secret est nécessaire sulement pour la génération du message d'authentfication par A, sa vérification pouvant être faite sans la connaissance du secret. Il existe aussi une classe particulière de techniques asymétriques basées sur le concept de preuve interactive appelée techniques à apport nul de connaissance qui assurent une parfaite confidentialité du secret connu par A.

3.2.1. *Techniques symétriques*

Les techniques symétriques permettant de calculer le message d'authentification se caractérisent par le fait que le secret connu par le demandeur A est partagé par le vérificateur B. La connaissance du secret est nécessaire aussi bien pour la génération que pour la vérification du message d'authentification. Deux méthodes cryptographiques se touvent à la base de ces techniques : le chiffrement symétrique ou les fonctions de hachage.

Dans le cas du chiffrement symétrique, le calcul du message d'authentification R envoyée par A vers B en réponse à une question explicite (nombre aléatoire) envoyée par B ou une question implicite (numéro de séquence ou cachet d'horodatage) déterminée par A peut être représenté par l'expression

suivante :

$$R = E_k\left(f\left(q, P\right)\right)$$

où

- E représente le chiffrement par un algorithme symétrique comme le Data Encryption Standard (DES) [STI 96],
- K est le secret partagé par A et B et correspond à la clé de chiffrement,
- f est une fonction non-cryptographique qui dépend du protocole et qui permet d'établir un lien entre la question q et d'autres paramètres éventuels représentés par P ; dans la plupart des cas f représente la concaténation simple des paramètres d'entrée ;
- q représente la question, c-à-d., un cachet d'horodatage, un numéro de séquence ou un nombre aléatoire, d'après le type de paramètre variable utilisé dans le protocole.

La vérification de R par B peut être effectuée de deux manières différentes :

– si f est une fonction inversible (q et P peuvent être déterminés à partir de $f(q, P)$) : B effectue le déchiffrement de R en utilisant l'algorithme symétrique et la clé K , extrait la valeur de q à partir du résultat du déchiffrement en calculant l'inverse de $f(q, P)$, vérifie la valeur de q par rapport à l'état du protocole (nombre aléatoire ou compteur) ou par rapport à l'intervalle d'acceptation (horodatage) ;

– B calcule $f(q, P)$ et effectue le chiffrement du résultat en utilisant l'algorithme symétrique et la clé K et compare le résultat au message reçu ; dans le cas des nombres aléatoires et des numéros de séquence B peut localement déterminer q et $f(q, P)$ à partir de l'état du protocole, dans le cas où q correspond à un cachet d'horodatage, la valeur en texte clair de q peut être envoyée dans le message d'authentification en compagnie de R ou bien l'entité B peut générer l'ensemble des résultats (R) correspondant à tous les instants (q) inclus dans l'intervalle d'acceptation et vérifier si le message reçu fait partie de cet ensemble.

Grâce aux propriétés de l'algorithme de chiffrement, il est impossible de calculer R pour un paramètre d'entrée q qui n'a jamais été utilisé auparavant sans connaître la clé de chiffrement K. Seul A et B qui connaissent le secret partagé peuvent donc calculer R, d'où B peut déduire avec certitude que l'origine du message R est l'entité A.

L'autre méthode symétrique pour calculer le lien entre la question et le secret consiste à utiliser les fonctions de hachage. Les fonctions de hachage sont des fonctions à sens unique qui génèrent un résultat de taille fixe pour un paramètre d'entrée de taille variable. Seules les fonctions de hachage résistantes aux collisions sont adaptées aux besoins des protocoles d'authentification (voir les propriétés de sécurité des fonctions de hachage dans [STI 96] [MEN 97]). Les fonctions Message Digest (MD4 [STI 96], MD5 [RIV 92]) ou les normes

SHA [MEN 97] et SHS [STA] sont des exemples de mise en œuvre logicielle pour de telles fonctions.

En utilisant une fonction de hachage, le calcul du message d'authentification R envoyée par A vers B en réponse à une question explicite (nombre aléatoire) envoyée par B ou une question implicite (numéro de séquence ou cachet d'horodatage) déterminée par A peut être représenté par l'expression suivante :

$$R = h_k\,(q, P)$$

où
- h représente une fonction de hachage résistante aux collisions, h_K correspond à la fonction de hachage avec clé où la clé est un paramètre de base de la fonction (comme pour un algorithme de chiffrement) ou bien elle est concaténée au message d'entrée par une forme qui dépend du protocole (Example : $h_K(m)$ représente $h(K1, m, K2)$ et $K = (K1, K2)$)
- K est le secret partagé entre A et B,
- q et P représentent respectivement la question et des paramètres spécifiques au protocole comme dans le cas du chiffrement.

Afin de vérifier le message d'authentification, B calcule l'expression R en utilisant la fonction de hachage, le secret partagé, la valeur de q et de P et compare le résultat au message reçu. En cas d'égalité le message d'authentification est accepté. Comme pour le cas du chiffrement, en fonction de la méthode utilisée pour générer q, la valeur de q peut être retrouvée à partir de l'état local (nombres aléatoires ou numéros de séquence) ou être envoyée en texte clair par A en compagnie du message R. Le fait que le calcul de R nécessite le secret partagé assure qu'un attaquant qui ne connaît pas ce secret ne peut pas générer une réponse R qui est valide pour une question q qui n'a pas été posée auparavant. Comme le même secret est utilisé pour plusieurs exécutions du protocole entre A et B, l'attaquant pourrait être tenté d'utiliser une valeur de R correspondant à une question ancienne (rejeu). La propriété de résistence aux collisions de la fonction de hachage assure que la probabilité d'obtenir le même résultat de hachage pour des valeurs d'entrée (questions) différentes est très petite. De plus, la présence du secret dans le calcul de R rend la recherche des collisions impossible sans la connaissance du secret.

D'autres expressions que les deux exemples précédentes peuvent être utilisées pour établir le lien entre q et le secret partagé dans la fabrication du message R (voir [MEN 97] et [TSU] pour des exemples d'expression plus complexes).

3.2.2. *Techniques asymétriques*

Dans le cas des techniques asymétriques, le calcul du message d'authentification par l'entité A se fait en utilisant un secret connu par la seule A tandis que la vérification du message se fait en utilisant une méthode publique qui ne nécessite pas la connaissance du secret. Deux types de fonctions cryptographiques s'appliquent dans cette catégorie : les fonctions de chiffrement asymétriques dites "à clé publique" comme le RSA [LAB 93] ou les fonctions de signature numérique comme le DSS [STI 96].

Les fonctions de chiffrement à clé publique offrent une opération de chiffrement E qui utilise une clé secrète appelée clé privée qui est connue par une seule entité et une fonction de déchiffrement D qui utilise une clé publique. Le résultat d'un chiffrement avec la clé privée ne peut être déchiffré que par la fonction de déchiffrement et en utilisant la clé publique. Chaque entité dans le système possède ainsi une clé privée connue seulement par l'entité elle-même et une clé publique connue par toutes les autres entités. Les fonctions de signature numérique offrent également deux opérations asymétriques : l'opération de génération de signature qui fournit une chaîne binaire de taille fixe appelée "signature" pour un message de taille variable et l'opération de vérification de signature qui teste le lien entre une signature et un message. Un paramètre d'entrée obligatoire pour la génération de la signature est une clé secrète connue par la seule entité qui est l'origine du message ou clé privée. La vérification de la signature nécessite la fourniture d'une clé publique associée à l'entité origine. Comme pour le chiffrement asymétrique chaque entité est dotée d'une clé privée et d'une clé publique. La différence fonctionnelle principale entre le chiffrement asymétrique et la signature numérique est que le chiffrement asymétrique est une bijection dont l'inverse est le déchiffrement tandis que la signature numérique n'est pas une bijection, l'inverse de l'opération de signature n'existe pas par définition. Un message chiffré par un algorithme asymétrique peut être retrouvé par le déchiffrement du cryptogramme tandis que la signature d'un message ne permet pas de reconstituer le message lui-même.

Le calcul des messages d'authentification par les fonctions asymétriques présente plusieurs similarité avec le calcul utilisant les fonctions symétriques. Dans le cas du chiffrement asymétrique, le message d'authentification calculé par A peut être représenté par l'expression suivante :

$$R = E_{KS_A}(f(q, P))$$

où
- E représente le chiffrement par un algorithme asymétrique,
- KS_A est la clé privée de l'entité A,

– les autres paramètres ont la même signification que pour le chiffrement symétrique.

La vérification de ce message d'authentification se fait par B en utilisant la clé publique de l'entité A.

Dans le cas de la signature numérique, l'entité A calcule et envoie à B le message

$$R = S_{KS_A}(q, P)$$

où
– S représente l'opération de génération de signature,
– KS_A est la clé privée de l'entité A,
– les autres paramètres ont la même signification que pour le chiffrement symétrique.

La vérification de ce message d'authentification se fait par B en appliquant l'opération de vérification de signature et en utilisant la clé publique de l'entité A. La vérification de la signature nécessite de connaître le message (q, P). Comme pour les fonctions de hachage, d'après le type de question utilisé par le protocole, si le message ne peut pas être déterminé par B en utilisant l'état local du protocole, la valeur en texte clair du message doit être tranmise en compagnie de R.

Dans le cas des fonctions asymétriques, le partage et la distribution d'un secret entre A et B n'est plus nécessaire, par contre un nouveau besoin apparaît : la sécurité du protocole d'authentification repose sur l'intégrité du lien entre chaque clé publique et l'identité de l'entité à laquelle cette clé appartient. Plusieurs solutions peuvent être utilisées pour assurer l'intégrité de ce lien, de la gestion manuelle des tables locales jusqu'aux protocoles de certification de clés publiques [ITU 93].

3.2.3. *Techniques à apport nul de connaissance*

L'authentification forte par les protocoles question-réponse apporte l'amélioration suivante par rapport à l'authentification faible : l'entité A prouve à l'entité B qu'elle possède le secret sans la dévoiler. Un protocole d'authentification question-réponse peut néanmoins laisser filtrer de l'information à propos du secret comme par exemple dans le cas d'un grand nombre d'exécutions en utilisant le même secret où certaines caractéristiques du secret peuvent être déterminées par l'attaquant si la technique cryptographique sous-jacente ne garantit pas la protection contre des attaques de type à texte clair choisi (voir [STI 96] pour la définition de ces attaques). Les techniques à apport nul de connaissance garantissent que la quantité d'information qui sera dévoilée pendant le

protocole d'authentification est inconditionnellement nulle. Ces techniques nécessitent plusieurs interactions sous forme de question-réponse et malgré la sécurité absolue qu'elles offrent en terme de fuite d'information elles restent trop complexes pour la plupart des besoins applicatifs où le nombre de messages d'authentification constitue un facteur de coût primordial.

3.3. *Modèles de communication*

Le dernier élément qui permet de distinguer les protocoles d'authentification est le modèle de communication qui concerne les entités impliquées dans le protocole d'authentification et la manière dont les messages sont échangés entre ces entités.

On distingue d'abord les variantes concernant seulement deux entités par rapport aux variantes où l'implication d'une tierce partie est nécessaire en plus des deux entités de base.

Dans le cas de la relation entre deux entités, une entité appelée **demandeur** prouve son identité à l'autre entité appelée **vérificateur**. Cette relation peut être **unidirectionnelle** quand seul le demandeur prouve son identité ou **mutuelle** quand les deux parties échangent des preuves entre elles pour s'authentifier réciproquement. L'authentification mutuelle est nécessaire quand l'initiateur d'une communication doit s'assurer de l'identité du répondeur en plus du répondeur qui vérifie l'identité de l'initiateur. Les applications transactionnelles ou certaines applications client-serveur où chaque client doit s'assurer de l'identité du serveur avant d'envoyer des requêtes sont des exemples de ce besoin.

L'implication d'une tierce partie s'avère nécessaire afin d'assurer des fonctions supplémentaires comme la distribution de clés ou la certification.

Dans le cas des protocoles utilisant les techniques symétriques, le partage d'un secret entre le demandeur et le vérificateur peut se faire par une installation manuelle de ce secret auprès de chaque partie concernée. Cependant la gestion manuelle devient très complexe quand le nombre de parties communicantes augmente et le recours à un système automatique devient nécessaire. Une solution à ce problème consiste à utiliser une tierce partie fiable appelée **serveur d'authentification** qui assure l'identité de chaque partie vis-à-vis des autres en éliminant le besoin de partage de secret de longue durée entre les parties communicantes. Les protocoles avec serveur d'authentification simplifient la gestion de la population d'utilisateur. Ainsi chaque utilisateur doit partager seulement un secret avec le serveur et l'installation d'un secret pour chaque paire d'entités communicantes n'est plus nécessaire. Grâce à l'utilisation d'un serveur d'authentification, la complexité de l'installation manuelle de

secrets est réduite de $O(n^2)$ à $O(^n)$ pour une population de n utilisateurs (la différence est déjà significative avec un petit nombre d'utilisateurs comme 100). L'impact de l'ajout ou de la suppression d'un utilisateur et de la mise à jour des secrets est aussi fortement réduit grâce à l'utilisation d'un serveur d'authentification. Bien que le problème du secret partagé disparaîsse dans le cas des techniques asymétriques, un nouveau besoin apparaît qui est celui de la garantie du lien entre la valeur de la clé publique et l'identité de l'entité qui possède cette clé pour chaque entité potentiellement impliquée dans un protocole d'authentification. Les solutions manuelles correspondant à ce problème atteignent très vite leur limite quand la population d'utilisateurs augmente comme pour le partage du secret dans le cas précédent. Une solution pratique consiste à utiliser une tierce partie appelée **autorité de certification** afin de garantir le lien entre la clé publique et l'identité de chaque entité par un procédé qui peut être vérifié par une seule clé publique connue et acceptée par tous. Le lien entre chaque clé publique et l'identité de son propriétaire est représenté par une information appelée **certificat**. La garantie du lien est assurée par le fait que chaque certificat est cryptographiquement signé par l'autorité de certification et cette signature peut être vérifiée par toutes les entités en utilisant la clé publique de l'autorité. Le vérificateur doit pouvoir accéder au certificat de la clé publique du demandeur pendant la vérification du message d'authentification. Cependant contrairement aux protocoles à base de serveur d'authentification, les protocoles de certification ne nécessitent pas l'accès en temps réel à l'autorité de certification. Seule la connaissance de la clé publique de l'autorité de certification est indispensable au moment de la vérification d'un certificat.

3.4. *Protocoles avec secret partagé*

Une grande variété de protocoles ont été proposés dans la littérature technique et plusieurs de ces protocoles présentent des failles de sécurité. Nous présentons ci-dessous quelques protocoles qui schématisent les exemples les plus significatifs. La notation suivante est utilisée pour la description de ces protocoles :

- X : représentation binaire de l'identité de l'entité X
- t_X : cachet d'horodatage émis par l'entité X
- s_X : numéro de séquence émis par l'entité X
- n_X : nombre aléatoire émmis par l'entité X
- $E_K(m_1, m_2, \ldots m_n)$: chiffrement par un algorithme symétrique de la chaîne binaire constituée par la concaténation des messages $m_1, m_2, \ldots m_n$
- $h_K(m_1, m_2, \ldots m_n)$: résultat du hachage avec la clé K de la chaîne binaire constituée par la concaténation des messages $m_1, m_2, \ldots m_n$
- KXY : clé secrète partagée par les entités X et Y

3.4.1. *Protocoles d'horodatage*

Protocole à base de chiffrement $A \rightarrow B : A, E_{Kab}(t_A, B)$

B retrouve la valeur de t_A en déchiffrant le message d'authentification avec la clé K_ab qui identifie l'entité A auprès de B. A est authentifié si t_A appartient à l'intervalle d'acceptation courant chez B. L'inclusion de l'identité de B dans le message d'authentification est une technique très simple qui permet d'éliminer les attaques qui consisteraient à effectuer un rejeu de ce message dans une session parallèle établie vers A par un intrus qui voudrait passer pour B.

Protocole à base de fonction de hachage $A \rightarrow B : A, h_{Kab}(t_A, B), t_A$

L'envoi de t_A est nécessaire afin de permettre à B de reconstituer tous les paramètres d'entrée de la fonction de hachage, la fonction de hachage n'étant pas inversible. En utilisant les paramètres ainsi constitués B calcule la fonction de hachage en utilisant la clé Kab et vérifie si le résultat est identique au message reçu. L'authentification de l'entité A se termine avec succès si tA appartient à l'intervalle d'acceptation courant chez B.

3.4.2. *Protocole utilisant les numéros de séquence*

Ce protocole est obtenu en utilisant un numéro de séquence comme paramètre variable dans le temps à la place du cachet d'horodatage. La génération de ce paramètre par A et sa vérification par B sont effectuées en utilisant des compteurs locaux qui sont synchronisés entre eux. $A \rightarrow B : A, E_{Kab}(s_A, B) ouh_{Kab}(s_A, B)$

La vérification du message se fait comme pour l'horodatage avec la différence suivante : la valeur de s_A est retrouvé par B en utilisant le compteur local.

3.4.3. *Protocoles utilisant les nombres aléatoires*

Protocole d'authentification unidirectionnelle Ce protocole introduit la notion de question implicite envoyée par le vérificateur. Par conséquent deux messages sont nécessaires à sa mise en œuvre.

1. $B \rightarrow A : n_B$
2. $A \rightarrow B : A, E_{Kab}(n_B, B) ouh_{Kab}(n_B, B)$

La vérification de la réponse se fait en utilisant le nombre aléatoire contenu dans l'état temporaire mémorisé par B.

Protocole d'authentification mutuelle L'authentification mutuelle constitute le véritable domaine d'application des nombres aléatoires. Le désavantage d'une interaction explicite avec deux messages qui est propre à cette technique disparaît dans le cas de l'authentification mutuelle où une interaction minimum avec deux messages est nécessaire par définition. Ce protocole présente une simplicité notable par l'absence d'état et de dépendance des mécanismes supplémentaires comme les horloges.

1. $A \rightarrow B : A, n_A$
2. $B \rightarrow A : B, n_B, E_{Kab}(n_A, A)\ ou\ h_{Kab}(n_A, A)$
3. $A \rightarrow B : E_{Kab}(n_A, n_B, B)\ ou\ h_{Kab}(n_A, n_B, B)$

Il faudrait noter dans le troisième message de ce protocole la présence des deux nombres aléatoires n_A et n_B qui est nécessaire pour éviter des attaques basées sur une exécution parallèle du même protocole afin d'obtenir les réponses en utilisant les entités légitimes comme un oracle. Si le troisième message de ce protocole avait le même format que le second message (si n_A n'y était pas pris en compte pour le calcul du résultat), l'attaquant X pourrait mettre en œuvre le scénario suivant pour passer pour A auprès de B :

1. $(session\ 1)\ X \rightarrow B : A, n_X$
2. $(session\ 1)\ B \rightarrow X : B, n_B, E_{Kab}(n_X, A) ou h_{Kab}(n_X, A)$
3. $(session\ 2)\ X \rightarrow A : B, n_B$
4. $(session\ 2)\ A \rightarrow X : A, n_A, E_{Kab}(n_B, B) ou h_{Kab}(n_B, B)$
5. $(session\ 1)\ X \rightarrow B : E_{Kab}(n_B, B) ou h_{Kab}(n_B, B)$

La deuxième session dans ce scénario, établie par l'attaquant en passant pour B auprès de l'entité A, permet à l'attaquant d'obtenir la réponse à la question (n_B) posée par B dans la première session en utilisant A comme oracle. Cette attaque réussit parce que le second message de la dexième session peut parfaitement être utilisé en tant que troisième message de la première session. Malgré leur simplicité apparente la conception des protocoles d'authentification résistant à des attaques similaires est une tâche assez complexe. [BIR 93] présente une analyse étendue des attaques correspondantes et une méthode de conception permettant de les éviter.

3.5. *Protocoles à base de clés publiques*

Deux méthodes se trouvent à la base des protocoles d'authentification utilisant les algorithmes à clés publiques :

- le demandeur chiffre (ou signe) la question avec sa clé privée et la réponse résultante est déchiffrée par le vérificateur en utilisant la clé publique du demandeur ;

– le demandeur déchiffre avec sa clé privée une question qui a été chiffrée par le vérificateur en utilisant la clé publique du demandeur.

La deuxième méthode qui requiert l'échange d'une question explicite est bien appropriée pour la technique des nombres aléatoires. Les trois techniques de génération de paramètre variable dans le temps peuvent être mises en œuvre par la première méthode. Bien qu'il existe beaucoup moins de variantes utilisant les techniques asymétriques que celles utilisant les techniques symétriques, les protocoles d'authentification à base de clés publiques sont de plus en plus répandus dans la mise en œuvre des nouveaux protocoles de commuication en raison de l'absence de besoin de distribution de secret partagé et grâce au développement des infrastructures de certification qui permettent une utilisant sécurisée des clés publiques. Nous présentons quelques exemples significatifs en utilisant la notation suivante pour la description de ces protocoles :

$CP_X(m_1, m_2, \cdots m_n)$: chiffrement de la chaîne binaire constituée par la concaténation des messages m1, m2, . . .mn par un algorithme asymétrique en utilisant la clé publique de l'entité X

$S_X(m_1, m_2, \cdots m_n)$: signature de $m1, m2, \cdots mn$ par l'entité Xăou chiffrement en utilisant la clé privée de X et un algorithme de chiffrement asymétrique (dans ce cas le résultat n'est quand même pas inversible puisque une fonction de hachage est appliquée avant ce chiffrement)

$Cert_X$: le certificat de la clé publique de l'entité X, permet de vérifier que la clé publique contenue dans le certificat appartient effectivement à X.

3.5.1. *Protocole d'horodatage*

$$A \rightarrow B : A, t_A, S_A(t_A, B), Cert_A$$

B retrouve et vérifie la clé publique de l'entité A à partir de $Cert_A$ et vérifie ensuite la signature en utilisant cette clé publique. A est authenitifié si la signature est valide et si t_A appartient à l'intervalle d'acceptation courant chez B.

L'envoi de t_A est nécessaire afin de permettre à B de reconstituer tous les paramètres d'entrée de la fonction de hachage, la fonction de signature n'étant pas inversible.

3.5.2. *Protocole utilisant les numéros de séquence*

Ce protocole est obtenu en utilisant un numéro de séquence comme paramètre variable dans le temps. La génération de ce paramètre par A et sa

vérification par B sont effectuées en utilisant des compteurs locaux qui sont synchronisés entre eux. $A \rightarrow B : A, S_A(s_A, B), Cert_A$

Contrairement au cachet d'horodatage (t_A) qui devrait être transmis en clair, la valeur de s_A est retrouvé par B en utilisant le compteur local.

3.5.3. *Protocoles utilisant les nombres aléatoires*

Authentification unidirectionnelle Le premier protocole d'authentification unidirectionnelle avec les nombres aléatoires utilise la méthode de la signature :

1. $B \rightarrow A : n_B$
2. $A \rightarrow B : A, S_A(n_B, B)), Cert_A$

Ce protocole crée une vulnérabilité potentielle : en choisissant des valeurs particulières pour le nombre aléatoire n_B, B peut obtenir la signature par A d'un message particulier qui peut être utile en dehors du contexte de l'authentification. Ce type de vulnérabilité connu sous le nom d'attaque à texte clair choisi peut constituer un faille importante pour l'algorithme de chiffrement sous-jacent. La solution à ce problème consiste à introduire un nombre aléatoire supplémentaire choisi par A dans le calcul de la réponse. Si ce nombre est aléatoirement choisi par A, B ne pourra pas contrôler le message qui sera signé par A. Le second protocole est basé sur la méthode du déchiffrement :

1. $B \rightarrow A : B, P_A(n_B, B)$
2. $A \rightarrow B : n_B$

Un problème similaire à celui de la signature existe dans ce contexte : en utilisant ce protocole B peut arriver à faire déchiffrer à A un message secret que A avait chiffré auparavant avec sa clé privée en dehors du contexte de l'authentification. Ce type de vulnérabilité est connu sous le nom d'attaques à cryptogramme choisi. La solution à ce problème consiste à forcer la vérification par A du fait que B a bien effectué le chiffrement dans le premier message. A cet effet B peut inclure dans le message avant chiffrement un champ supplémentaire contenant une valeur aléatoire n' et ajouter un champ supplémentaire au premier message qui contient $h(n')$. Ainsi A ne répondrait à B que si le hachage par h du nombre n' qu'elle obtient après déchiffrement est bien égal au champ supplémentaire du premier message.

Authentification mutuelle L'authentification mutuelle peut aussi être effectuée par les deux variantes. La première utilise la méthode de la signature :

1. $A \rightarrow B : A, n_A$
2. $B \rightarrow A : B, n_B, S_B(n_A, A), Cert_B$
3. $A \rightarrow B : S_A(n_B, B), Cert_A$

Comme pour l'authentification unidirectionnelle, l'inclusion d'un second nombre aléatoire dans chaque message signé peut être utile afin d'eviter des attaques à texte clair choisi. La seconde variante utilise la méthode du déchiffrement :

1. $A \rightarrow B : A, P_B(n_A, A)$
2. $B \rightarrow A : B, P_A(n_A, n_B, B)$
3. $A \rightarrow B : n_B$

3.6. *Protocoles utilisant un serveur d'authentification*

Le concept de serveur d'authentification décrit dans le paragraphe 3.2.3. résoud principalement le problème de la distribution du secret partagé dans le cas des protocoles à base d'algorithme symétrique. Le premier protocole d'authentification utilisant un serveur a été introduit par Needham et Schroeder [NEE] en 1978. Dans ce protocole, au lieu de partager un secret différent avec chacun de ses correspondants potentiels, chaque entité partage seulement un seul secret avec le serveur d'authentification appelé S. Quand une entité (A) a besoin de s'authentifier auprès d'une autre (B), A contacte S comme suit :

1. $A \rightarrow S : A, B, n_A$
2. $S \rightarrow A : E_{Ka}(n_A, B, Kab, ticket = E_{Kb}(A, Kab))$
3. $A \rightarrow B : ticket = E_{Kb}(A, Kab)$
4. $B \rightarrow A : E_{Kab}(n_B)$
5. $A \rightarrow B : E_{Kab}(n_B - 1)$

Un nouveau secret Kab qui est valable pour la durée de cette instance du protocole est généré par le serveur et envoyé à A dans le message 2. qui est chiffré avec le secret individuel Ka partagé par A et S. Ce message contient également un autre champ chiffré appelé **ticket** qui est une enveloppe chiffrée par la clé Kb de B et contenant la nouvelle clé Kab. A la réception du message 2., A découvre la valeur de Kab par le déchiffrement du message en utilisant la clé Ka. Par ce déchiffrement A obtient aussi la valeur du ticket qu'elle transmet directement à B. D'une manière similaire, B retrouve la nouvelle clé partagée Kab en déchiffrant le ticket avec sa clé individuelle Kb. Les messages 4. et 5. constituent un protocole pour l'authentification unidirectionnelle de A par B utilisant le secret partagé Kab.

Au départ seule deux relations binaires sont possibles en utilisant les clés individuelles Ka et Kb : l'authentification entre A et S et celle entre S et B. Le protocole de Needham-Schroeder permet d'établir une troisième relation entre A et B par l'extension de ces deux relations initiales. Cette extension repose

cependant sur une propriété nouvelle qui est la confiance placée en S. B doit en effet faire confiance à S à propos de l'authentification de A par S. Ainsi le ticket chiffré fourni par S est un certificat qui garantit à B que toute entité qui prouve sa connaissance de la nouvelle clé Kab peut être considérée comme étant A. Comme B ne peut pas vérifier la manière dont la clé Kab est communiquée à A, B doit faire confiance à S dans l'authentification de A pendant la distribution de cette clé.

Bien qu'étant le précurseur dans son domaine, le protocole de Needham-Schroeder présente une faiblesse par rapport aux hypothèses ci-dessus. Malgré la nécessité pour A de connaître la clé secrète Ka pour pouvoir obtenir la nouvelle clé Kab, aucun élément dans ce protocole ne permet de garantir que la clé Kab est nouvelle. Ainsi si la valeur de la clé Kab correspondant à une exécution antérieure du protocole avait été dévoilée à un attaquant (en théorie l'affichage d'une clé ancienne ne constitute pas une vulnérabilité puisque chaque nouvelle exécution du protocole est censée utiliser une nouvelle clé), l'attaquant peut parfaitement réussir à passer pour A auprès de B en effectuant un rejeu du message 3. et en exécutant le reste du protocole en utilisant la valeur de Kab dévoilée par mégarde.

Cette faiblesse a été remarquée par les inventeurs de ce protocole [NEE 87] et une version corrigée du protocole incluant quelques améliorations a été mise en œuvre par le système Kerberos [J. 93]. L'amélioration principal du protocole de Kerberos consiste à inclure un cachet d'horodatage dans le ticket afin de limiter la durée de vie de la clé Kab. Plusieurs variantes de protocoles d'authentification avec un serveur ont été proposées [OTW 87] [BIR 95].

3.7. *Dispositifs personnels*

L'authentification forte repose sur la connaissance par le demandeur d'un secret qui résiste à diverses attaques concernant la méthode cryptographique utilisée par le protocole d'authentification. Dans le cas des technques symétriques, un secret de 90 bits aléatoirement choisi est considéré comme sûr à l'état actuel de la technologie. D'une façon similaire, pour un algorithme asymétrique, une clé privée sure doit au minimum contenir 768 bits. L'exigence concernant la taille du secret pose un problème dans le cas où l'entité qui exécute le protocole d'authentification forte est un système public qui agit au nom d'un utilisateur humain (terminal public, distributeur de billets, etc.) :

- d'une part, le secret ne peut pas être mémorisé par l'utilisateur humain puisque contrairement aux informations redondantes qui sont faciles à mémoriser pour l'homme, un secret sûr est une donnée aléatoire ou pseudo-aléatoire qui ne présente pas de redondance,

– d'autre part, un secret ne peut pas être stocké dans la mémoire d'un système qui est public pour des raisons évidentes.

La solution à ce problème consiste à doter chaque utilisateur d'un dispositif personnel qui lui permet de mettre en œuvre un protocole d'authentification forte à base de secrets sûrs sans toutefois nécessiter leur mémorisation par l'utilisateur. Nous considérons deux types de dispositifs personnels qui peuvent répondre à ce besoin :

– les dipositifs passifs ont la seule fonction de mémoire et
– les dipositifs actifs qui possèdent les fonctions de mémoire et de calcul qui leur permettent d'agir comme une entité d'authentification active.

3.7.1. *Dispositifs passifs*

Les dispositifs passifs contiennent une portion de mémoire sur laquelle sont inscrites les données nécessaires à la vérification de l'identité d'un utilisateur. En utilisant ces données plusieurs systèmes d'authentification publics peuvent procéder à l'authentification forte de l'utilisateur comme dans le schéma fonctionnel présenté dans la figure 2.1.

Dans ce schéma, le système d'authentification vérifie le secret de l'utilisateur (PIN) par rapport à l'identité (ID) fournie par le dispositif en utilisant une clé secrète K qui est commune à tous les systèmes d'authentification. Le PIN est un secret faible et facile mémoriser par un utilisateur humain tandis que K est un secret sûr dont la découverte par recherche exhaustive est impossible. La valeur d'ID et le résultat d'une fonction à sens unique (FSU) calculée sur ID, PIN et K sont stockés sur le dispositif et peuvent être lus par un simple lecteur de dispositif. L'expression h(PIN, ID, K) utilisant une fonction de hachage comme MD5 peut être un exemple pour une FSU. Le caractère "à sens unique" de FSU assure que les paramètres d'entrée ne peuvent pas être retrouvées à partir du résultat de la fonction. Grâce à la fonction à sens unique, la possession du dispositif sans la connaissance de PIN ne permet pas à un intrus de s'authentifier en passant pour l'utilisateur légitime. Le vol du dispositif ne permet de découvrir la valeur du PIN par une recherche exhaustive de toutes les valeurs, même si le PIN constitute un secret faible, puisque la connaissance de l'autre secret K, qui est un secret sûr, est aussi nécessaire pour tester les valeurs candidates pour le PIN. Le test exhaustif qui consiste simplement à rentrer successivement des valeurs de PIN en utilisant un système d'authentification n'est pas permis, le nombre d'essais étant limité à un petit nombre sur chaque système.

Le processus de vérification décrit précédemment pourrait être mis en œuvre par un système d'authentification sans utiliser de dispositif personnel. Le dispositif personnel joue principalement le rôle d'une mémoire déportée pour le

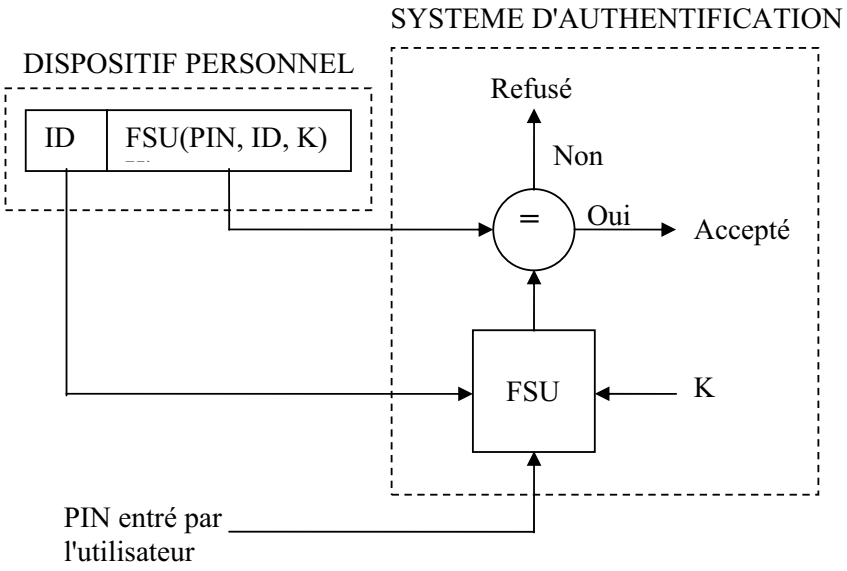

Figure 2.2. *Schéma d'authentification à base d'un dispositif personnel passif*

système de vérification et permet une gestion distribuée de la population d'utilisateurs.

Grâce à l'utilisation d'une clé sure (K) et à la limitation des essais sur le secret faible (PIN), ce type de dispositif permet d'obtenir une authentification forte. Le désavantage principal de ce schéma est le partage du secret K par tous les systèmes d'authentification qui constitue un point de faiblesse commun pour l'ensemble. L'exemple le plus connu de tels dispositifs est le système de carte bancaires à bande magnétique.

3.7.2. *Dispositifs actifs*

Les dispositifs actifs possèdent en plus de la capacité de mémorisation, une capacité de calcul qui leur permet d'agir comme une entité indépendante dans le processus d'authentification. Les secrets mémorisés par un dispositif actif ne sont pas accessibles en lecture pour une entité externe ; seul le dispositif lui-même peut accéder aux secrets contenus dans sa mémoire. Ainsi le dispositif actif communique avec les entités externes en échangeant des messages d'authentification définis d'après un des protocoles présentés dans ce chapitre.

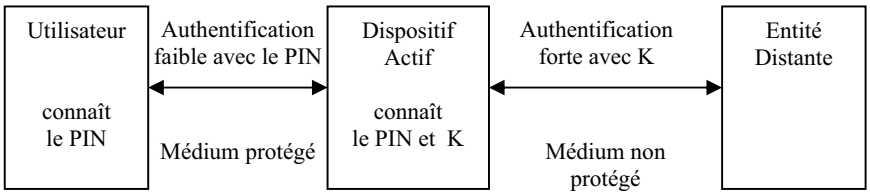

Figure 2.3. *Les deux phases du processus d'authentification par un dispositif actif*

Conformément au principe d'authentification forte, un dispositif actif ne divulgue jamais le secret mais démontre sa connaissance du secret en répondant à des questions. L'authentification forte d'un utilisateur par une entité distante et à travers un dispositif actif s'effectue en deux phases (figure 2.2) :

- dans une première phase le dispositif actif procède à la vérification de l'utilisateur sur la base du secret faible (PIN) et à travers un médium de communication protégé comme dans le cas d'un dispositif passif ;
- une fois l'authentification de l'utilisateur par le dispositif terminée avec succès, le dispositif agit au nom de l'utilisateur en mettant en œuvre un protocole d'authentification forte à base d'un secret sûr (K) mémorisé dans le dispositif pour s'authentifier auprès d'autres entités distantes à travers un médium de communication qui est éventuellement exposé à des attaques.

La première phase de ce processus présente un avantage par rapport à l'authentification avec un dispositif passif : il n'est pas nécessaire de placer un secret commun dans plusieurs systèmes d'authentification. En effet, grâce à sa capacité de calcul, le dispositif actif peut assurer la vérification de l'utilisateur et une garantie contre la recherche exhaustive sans que la lecture par un système externe des secrets contenus dans le dispositifs soit nécessaire.

Le principe du dispositif actif peut être mis en œuvre de plusieurs manières différentes, une analyse de l'existant permet de distinguer trois catégories :

- les ordinateurs personnels : Un ordinateur portable dont l'utilisation strictement personnelle peut être assurée par un système d'exploitation constitue une forme très puissante de dispositif actif.
- les calculettes : C'est la mise en œuvre le plus simple du principe de dispositif actif. Les deux phases du processus d'authentification sont assurées par un système autonome sous forme d'une calculatrice de poche qui comporte une organe d'entrée (clavier ou simple bouton) et un organe de sortie qui est un affichage alphanumérique. La caractéristique principale d'une calculette est qu'elle ne possède pas d'interface de communication avec un système externe, l'utilisateur doit assurer le transfert des informations entre la calculette et l'entité distante pour la deuxième

phase du processus d'authentification. Ceci est un avantage du point de vue matériel puisqu'un système avec plusieurs utilisateurs peut être doté d'un mécanisme d'authentification forte sans qu'aucune modification matérielle soit nécessaire sur le parc de terminaux existants. Le désavantage des calculettes est le manque d'ergonomie pour les utilisateurs pour qui l'avantage de l'authenitfication forte s'accompagne de la nécessité de passer par des manipulations complexes à chaque authentification.

– les cartes à puce à microprocesseur : Cette catégorie réunit les avantages d'un dispositif personnel actif avec la facilité d'utilisation. Contrairement aux calculettes, les cartes à puce possèdent une interface à travers laquelle elles peuvent communiquer directement avec une entité externe comme un terminal public ou un ordinateur personnel. Grâce à cette interface, l'utilisateur ne doit plus jouer le rôle d'intermédiaire entre le dispositif et l'entité externe. Cependant le déploiement de tels dispositifs nécessite que les entités externes soient dotées d'une interface matérielle permettant la communication avec les dispositifs. Cette interface se présente sous forme d'un lecteur de carte à puce connecté à un ordinateur. Les cartes à puce à microprocesseurs peuvent exécuter les deux phases du processus d'authentification et pour la seconde phase elles offrent une grande flexibilité de mise en œuvre concernant le choix du protocole d'authentification forte puisque l'interface directe avec l'entité externe permet d'échanger plusieurs messages de plusieurs centaines d'octets.

4. Authentification Biométrique

Même s'ils sont forts, les moyens d'authentification passés en revue plus haut n'offrent pas une sécurité absolue quand ils sont effectivement mis en œuvre : un utilisateur peut perdre ou se faire voler sa carte à puce ; il semble que 25 % des gens conservent leur PIN avec leur carte bancaire ; de nombreux utilisateurs écrivent leur mot de passe pour s'en souvenir.

Ces problèmes mettent en lumière le manque d'ergonomie des techniques d'authentification traditionnelles. C'est pourquoi des méthodes de vérification de l'identité basées sur les caractéristiques physiologiques ou les traits comportementaux d'un individu ont été avancées : elles ne reposent pas sur l'endroit où se trouve l'utilisateur (une adresse), ni sur ce que l'utilisateur sait (un secret), ni sur ce qu'il possède (un dispositif externe), mais sur "ce qu'il est", en permanence et en tout lieu. L'ensemble de ces caractéristiques mesurables sont qualifiées de caractéristiques biométriques [K.].

Les dispositifs d'authentification biométrique sont d'un coût supérieur aux autres méthodes d'authentification, ce qui en limite l'usage courant, mais ce

coût tend à baisser en particulier en raison de l'augmentation de la puissance de calcul aujourd'hui disponible. Leur acceptation par les utilisateurs est un point essentiel : en particulier, un dispositif désagréable d'utilisation ne peut normalement être retenu mis à part certains cas très particuliers.

4.1. *Processus de vérification biométrique*

Dans toutes les techniques d'identification biométrique, le processus de vérification de l'identité se déroule systématiquement en deux temps : le système est initialisé par une présentation du sujet, qui peut être suivie de vérifications réelles.

Lors de la présentation du sujet, ses caractéristiques biométriques sont acquises, traitées et enregistrées sous forme d'une "signature" biométrique. Cette signature peut être conservée localement dans le dispositif de lecture lui-même, stockée dans une base de données centralisée, ou bien encore enregistrée dans un dispositif matériel personnel comme une carte à puce si la signature n'est pas de taille trop importante.

Pour la vérification, le sujet décline tout d'abord son identité en tapant un identifiant ou en insérant une carte d'identité numérique ; le système de vérification récupère les caractéristiques biométriques enregistrées à la première étape sous l'identité présentée ; le sujet utilise enfin le système d'identification biométrique pour s'identifier. Le système de vérification construit un profil du sujet et le compare à celui enregistré à la première étape du processus.

La comparaison des deux profils ne produit pas une adéquation exacte, ce qui serait d'ailleurs suspect en soi : contrairement aux systèmes d'authentification classiques, il n'y a pas authentification sûre à 100 %. Le système de vérification se baser sur une notion de similitude, celle-ci pouvant être établie à une valeur entre 0 % et 99 %. Cette variation des résultats d'un individu est plus liée à la qualité de l'acquisition de l'information biométrique qu'à la modification de la caractéristique biométrique de l'individu, généralement stable dans le temps.

L'acceptation ou le rejet de l'identité nécessite le choix d'un seuil de décision, c'est-à-dire en conséquence une certaine marge d'erreur. Le réglage de ce seuil de décision est critique pour le bon fonctionnement du système : trop faible, le taux d'acceptation d'imposteurs est inacceptable ; trop élevé, des sujets autorisés seront rejetés par le système de vérification. Ainsi, un système d'authentification biométrique peut être caractérisé par deux taux [W. 89] : le taux de faux rejets (TFR), aussi appelées fausses alarmes ou erreurs de type I, qui indique le pourcentage d'utilisateurs rejetés par erreur ; le taux de fausses

acceptations (TFA), aussi appelées erreurs de type II, qui indique le pourcentage d'acceptations d'imposteurs. Ces taux vont dépendre de la caractéristique biométrique vérifiée, de la qualité d'acquisition et de traitement du système, mais également du niveau de sécurité souhaité. La performance d'un tel système est généralement évaluée par le TFR et le TFA, ainsi que par le temps pris par l'authentification elle-même.

4.2. *Caractéristiques biométriques*

L'utilisation de nombreuses caractéristiques biométriques a été proposée. Elles peuvent être classées selon deux types différents :

- caractéristiques comportementales : signatures manuelles, vitesse de frappe
- caractéristiques morphologiques : forme de la main, empreintes digitales, empreintes rétiniennes, reconnaissance de l'iris, reconnaissance du visage, empreintes vocales, thermographie du visage

Les caractéristiques morphologiques ont en général l'avantage d'être stables dans la vie d'un individu contrairement aux caractéristiques comportementales, sujettes en particulier au stress de l'utilisateur. Elles sont idéalement uniques pour un individu et non reproductible par des artefacts.

4.2.1. *Signatures manuelles*

C'est une forme classique d'authentification humaine. Les signatures peuvent être utilisées par le système de vérification comme par un expert humain. Il est aussi possible d'enregistrer non seulement la signature, mais aussi la vitesse réelle des mouvements. Ce genre de système nécessite l'utilisation de tablettes électroniques ou d'accéléromètres dans le stylo. La signature est depuis longtemps établie comme une preuve de l'identité : c'est l'avantage principal de ce type d'authentification en termes d'acceptation par les utilisateurs. De plus, il est très difficile pour un faussaire de reproduire les mouvements et la pression exercée par l'auteur authentique d'une signature.

4.2.2. *Vitesse de frappe*

La façon dont un individu tape au clavier est distinctive et des études ont été menées pour baser une authentification sur ce critère. Un problème avec ce type d'authentification est qu'une blessure peut modifier la vitesse de frappe. De plus, le processus ne peut s'effectuer de manière centralisée car le rythme

d'arrivée des frappes au clavier est déformé en traversant un réseau. Cette méthode n'est pas non plus adaptée pour le grand public en général.

4.2.3. *Forme de la main*

Il s'agit ici de mesurer les dimensions de la main : longueur et largeur des doigts, proportions, taille de la main, etc. Contrairement à d'autres caractéristiques biométriques, la forme de la main n'est pas aussi unique à un individu. L'intérêt de cette caractéristique biométrique repose plutôt sur le fait que toute personne peut l'utiliser facilement, contrairement par exemple à l'empreinte rétinale, très intrusive, ou à l'empreinte digitale, sujette à la qualité d'acquisition des empreintes. Utiles pour garder l'accès à des locaux, les lecteurs sont par contre trop encombrants pour un usage sur un bureau, dans une voiture ou sur un téléphone.

4.2.4. *Empreintes digitales*

Parce qu'un individu laisse une empreinte latente sur tout objet touché, les empreintes digitales sont utilisées avec succès comme moyen d'identification par les polices du monde entier la fin du XIXème siècle. L'utilité des empreintes digitales provient essentiellement du fait que leur variabilité d'un individu à l'autre est si grande qu'il est généralement admis qu'il n'existe pas deux personnes partageant la même empreinte. Alors même que les doigts changent de taille avec la croissance, la géométrie des empreintes digitales ne varie pas, sauf à employer des moyens chirurgicaux ou en cas de blessure.

Une empreinte comporte deux types d'arrangement des lignes papillaires : chaque doigt comporte d'une part une ou plusieurs structures à grande échelle comme des arches, des boucles, des tourbillons ; d'autre part, chaque empreinte porte entre 50 et 200 minuties qui sont des points caractéristiques comme les arrêts de lignes, bifurcations, lacs, îlots, points. L'identification se base surtout sur la localisation et l'orientation de ces minuties. Dans la pratique judiciaire actuelle, il faut de 8 à 17 points sans discordance pour qu'on estime établie l'identité d'un individu. Des analyses statistiques ont établi que la probabilité de retrouver deux configurations similaires de minuties sur les empreintes digitales de deux individus est de l'ordre de 10-20.

La technologie la plus utilisée pour l'acquisition d'empreinte était jusqu'à présent optique. Les nouvelles générations de lecteurs se servent de grilles semi-conductrices : chaque point de la grille constitue une moitié de condensateur, un point de l'empreinte jouant le rôle de l'autre. L'usage de ces dernières générations de lecteurs, plus petits et moins chers, pourrait se généraliser. Du point

de vue de l'utilisateur, l'acquisition d'une empreinte est non intrusive, même si une image de criminalité reste attachée à l'utilisation des empreintes digitales.

4.2.5. *Empreintes rétiniennes*

Suggérée dès les années 1930, cette technique consiste en l'examen des petits vaisseaux sanguins qui tapissent la rétine, au fond de l'œil. La disposition de ces vaisseaux est extrêmement caractéristique d'un individu. Le sujet regarde fixement un point à travers un oculaire. La machine peut alors localiser la fovea, une zone non irriguée du centre de l'œil. En tournant autour de cette zone, elle localise alors les nœuds et les branches des vaisseaux rétiniens. Plus de 400 variables différentes peuvent être isolées par cette méthode. Quoique la qualité des caractéristiques mesurées soit excellente, cette technique qui est considérée comme très intrusive nécessite la coopération attentive du sujet, ce qui en limite l'utilisation à des utilisations critiques, militaires notamment. De plus, le coût de cette technique est assez élevé.

4.2.6. *Reconnaissance de l'iris*

On doit cette méthode à des ophtalmologues, autour de 1980. L'iris, c'est-à-dire la zone comprise entre la pupille et le blanc de l'œil, comporte un motif radial dense, unique, indépendant de toute caractéristique génétique et invariant dans le temps, duquel on peut dégager plus de 260 variables indépendantes. La probabilité est très faible de confondre deux individus : sur un système qui code le dessin de l'iris sur 512 octets, la probabilité de trouver deux iris identiques à 75 % est estimée à 10-16. L'acquisition de l'image de l'iris ne nécessite pas d'éclairage comme pour la rétine et n'est donc pas intrusive. L'utilisateur fixe simplement l'objectif d'une caméra qui récupère en 1/4 s le dessin de son iris. Par contre, l'éclairage de l'iris pose un problème de reflets : on utilise souvent un éclairage artificiel calibré, tout en atténuant le plus possible l'éclairage ambiant, éclairage d'autant mieux toléré qu'il peut-être infrarouge et donc invisible pour l'œil. Le système peut potentiellement être trompé à partir d'une photo ou d'une lentille de contact reproduisant l'iris d'un utilisateur autorisé, mais la résolution demandée est très importante. De plus il est possible de repérer par filtrage que l'iris présenté est constitué d'une suite régulière de points et non d'un motif continu et varié. Enfin, d'autres techniques permettent de s'assurer que l'iris présenté n'est pas un artefact. On peut faire varier l'éclairage, faisant aussi varier le diamètre de la pupille et ce, avec une latence et une vitesse de variation mesurables. On peut aussi éclairer l'oeil dans des longueurs d'ondes ultraviolettes ou infrarouges lointaines dans lesquelles il est opaque.

4.2.7. *Reconnaissance du visage*

Un système de reconnaissance du visage peut fonctionner avec une simple caméra comme en sont équipés de nombreux ordinateurs personnels aujourd'hui : c'est donc certainement l'un des systèmes biométriques les moins coûteux aujourd'hui. Certains systèmes reconnaissent un utilisateur à son profil. D'autres s'appuient sur des caractéristiques anthropométriques comme par exemple la forme des oreilles, l'écartement des yeux, l'écartement des narines ou encore les dimensions de la bouche pour identifier un individu. La méthode doit tenir compte de certain changement possibles de physionomie (lunettes, barbe, etc.) et de l'environnement, en particulier les conditions d'éclairage ou l'orientation du sujet.

4.2.8. *Empreintes vocales*

La reconnaissance des individus par leur voix est un talent humain assez courant et aisé, même dans un environnement bruyant. Ce type d'authentification est utile dans la vie courante pour identifier son interlocuteur au téléphone par exemple. De manière automatique, chaque son peut être caractérisé par une fréquence, une intensité et une tonalité. Il est possible de faire une analyse de la voix d'une personne et de s'en servir comme empreinte pour une reconnaissance ultérieure.

Cette technique économique est cependant peu employée. En effet, elle est sujette à des attaques : imitation de la voix d'une personne autorisée, ou pire, enregistrement de cette personne. De plus, fatigue, stress ou maladie peuvent altérer la voix d'un utilisateur qui risque alors de ne pas être reconnue. Elle pourrait trouver sa place dans les équipements de taille réduite.

4.2.9. *Thermographie du visage*

Dans cette technique, une caméra thermique réalise un cliché infrarouge du visage de l'utilisateur afin de faire une carte des zones de températures différentes et leur répartition, voire du réseau veineux du visage, invisible à l'œil nu. Cette carte est unique pour chaque individu. Très onéreux, ce système reste expérimental.

4.3. *Choix d'un système biométrique*

Il n'existe pas un seul et unique système d'authentification biométrique valable pour toutes les applications possibles. Plusieurs paramètres entrent en jeu dans le choix d'un système biométrique :

- techniques : résistance aux attaques physiques, facilité de fabrication d'un artefact, temps pour effectuer la reconnaissance, calculs nécessaires pour le processus d'identification, facilité à être circonvenu, sûreté et facilité de maintenance.
- ergonomiques : commodité pour l'utilisateur, interfaçage du dispositif avec une application donnée, temps et efforts impliqués pour la mise à jour des données biométriques.
- économiques : coût du dispositif de reconnaissance et de son utilisation, coût pour protéger le dispositif, coût de distribution et support logistique.

La Figure 2.2 (d'après [ANA 00]) analyse quelques systèmes biométriques en fonction de quelques uns de ces paramètres : le temps et la difficulté pour l'utilisateur (effort), le degré intrusif du dispositif de mesure biométrique, le coût du dispositif de mesure, et l'efficacité biométrique (exactitude). On peut trouver d'autres études sur les systèmes biométriques dans [W. 89].

Les caractéristiques biométriques qui paraissent actuellement les plus prometteuses pour une utilisation grand public sont la reconnaissance d'iris et les empreintes digitales pour leur exactitude, la reconnaissance du visage pour son coût, et dans une moindre mesure, la reconnaissance vocale pour son ergonomie dans un appareil portable.

Pour finir, pourquoi utiliser un système d'authentification biométrique plutôt qu'un autre type d'authentification ? Un système biométrique est essentiellement destiné à authentifier un utilisateur avec aussi peu de gêne que possible : celui-ci porte naturellement avec lui les preuves qui permettent de vérifier son identité avec un maximum d'ergonomie. Plusieurs systèmes biométriques n'ont pas une exactitude suffisante pour être employés seuls. Il peuvent être associés à des techniques d'authentification plus traditionnelles mais perdent en confort d'utilisation et en ergonomie même si la sécurité en est renforcée. Autre approche plus intéressante, plusieurs systèmes biométriques faciles d'emploi mais pas assez exacts peuvent être associés pour fournir une identification plus sûre ou diminuer le taux de fausses alarmes : on parle alors de systèmes multi-biométriques ou de biométrique multi-modale.

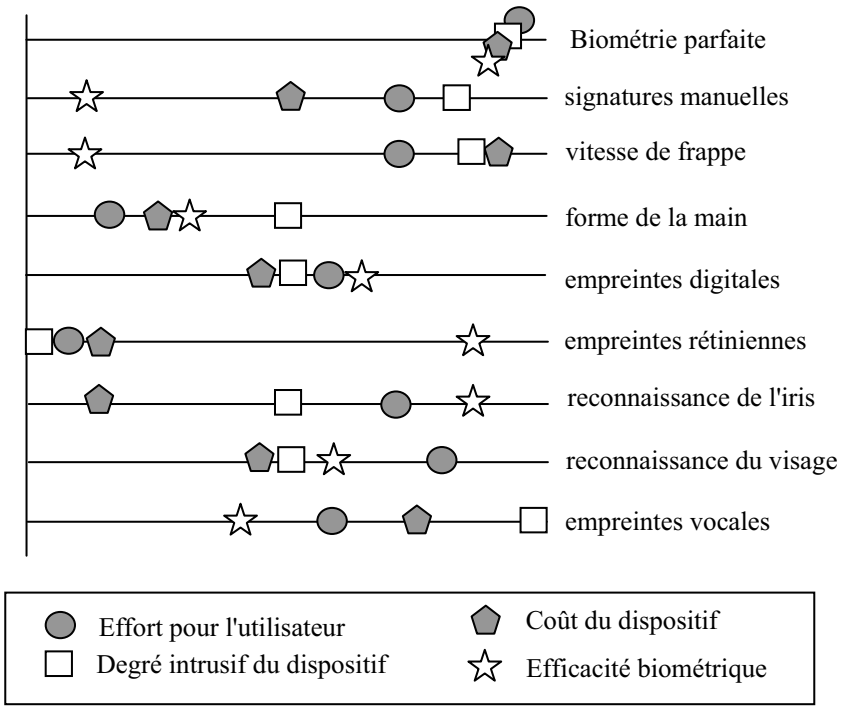

Figure 2.4. *Comparaison de caractéristiques biométriques*