Cartes à puce internet, état de l'art et perpectives.

Pascal Urien, ***Pascal.Urien@louveciennes.tt.slb.com***
SchumbergerSema, Smartcard Research Center.

Hayder Saleh, ***Hayder.Saleh@louveciennes.tt.slb.com***
Doctorant cifre, laboratoire prism UVSQ

Adel Tizraoui, ***Adel.Tizraoui@louveciennes.tt.slb.com***
Doctorant cifre, LIP6.

Résumé : *Autrefois basées sur des processeurs de faibles puissances et offrant des capacités mémoires réduites, les puces sécurisées ont été historiquement mise en œuvre pour la sécurisation des paiements (cartes bancaires). Les récentes évolutions technologiques de ces composants, par exemple l'introduction de processeurs RISC 32 bits, permettent de définir de nouvelles architectures logicielles qui comportent une pile de communication IP et réalisent des organisations de données complexes telles que descriptions XML ou bases de données. Ce papier présente l'état de l'art de telles architectures et leurs perpectives d'applications à des services tels que eCommerce, session multimédia (VoIP,…), et gestion de la qualité de service (QoS), dans les réseaux de nouvelles génération.*

Introduction

La carte à puce à microprocesseur (SPOM – self programmable one chip microcomputer) est née au début des années 80. Schématiquement cette pastille de silicium réalise [1] cinq types d'opérations, l'entrée et la sortie de données, la lecture ou l'écriture de données dans une mémoire non volatile (ROM, E^2PROM) et le calcul d'algorithmes cryptographiques. Des capacités de stockage limitées à quelques kilo-octets, des puissances de traitement relativement modestes (de l'ordre du MIPS) ont conduit à développer des architectures logicielles basées sur un classique paradigme d'appel de procédure (Remote Procedure Call); en d'autres termes le *middleware* associé aux cartes à puce réalise la plupart des traitements logiciels. L'identification des ressources cryptographiques embarquées (et donc de la structure des données stockées dans la mémoire) est en règle générale déduite de manière implicite (par exemple une carte SIM fonctionne avec un téléphone portable) ou visuelle (à l'aide d'informations imprimées sur le rectangle de PVC qui supporte la puce).
La convergence de plusieurs facteurs peuvent contribuer à modifier les modes d'utilisation des cartes à puces dans les technologies de l'information et plus particulièrement dans les réseaux fixes et mobiles. Le nombre d'objets informatiques communiquant est en constante augmentation (téléphones mobiles, ordinateurs personnels accédant à des réseaux fixes ou mobiles, assistant personnels,…), les échanges de données entre ces entités informatiques (souvent portables) créent de nouveaux besoins (et donc des marchés) en particulier pour l'authentification des utilisateurs ou des logiciels téléchargés. Dans ces conditions il devient de plus en plus complexe de développer des middlewares cartes destinés à des terminaux hétéroclites.
On assiste aujourd'hui à une convergence des réseaux communication, qui utilisent pour la plupart d'entre eux la pile protocolaire IP. En conséquence les terminaux supportent également une pile IP, l'augmentation des capacités mémoires et des puissances de traitement des puces permet d'intégrer des couches de communication compatibles avec le monde IP, la carte devient un nœud internet sécurisé qui embarque du middleware applicatif. Les logiciels côté terminaux sont donc simplifiés, par exemple une carte à puce supporte le protocole HTTP, son interface est similaire à celle d'un serveur WEB de haute sécurité.
Ce papier présente les différentes approches pour intégrer des puces sécurisées aux technologies internet, et tente de définir quelques fonctions qui seront utiles pour les réseaux et services de nouvelles générations.

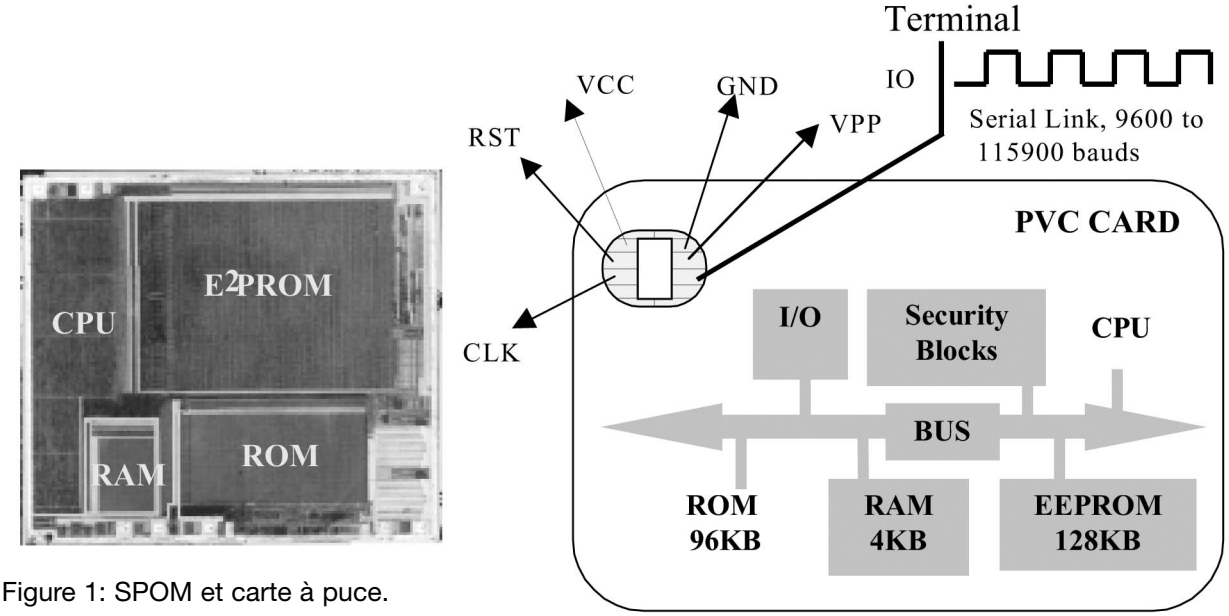

Figure 1: SPOM et carte à puce.

Principales caractéristiques.

Une puce sécurisée se présente typiquement sous la forme d'un rectangle de silicium dont la surface est inférieure à 25 mm^2. D'une part cette taille est imposée par les contraintes de flexion induite par le support en PVC, et d'autre part cette dimension limitée réalise un compromis entre sécurité physique et complexité du composant. Un SPOM (figure 1) comporte un CPU relié à l'aide d'un bus d'interconnexion à différents blocs mémoire (ROM, E^{2}PROM, RAM) ainsi qu'à un bloc IO (qui gère une liaison série).
Les capacités mémoires sont comprises entre 128 et 256 Ko pour la ROM (surface relative 1), 64 et 128 ko pour l'E^{2}PROM (surface relative 4), 4 et 8 Ko pour la RAM (surface relative 16). En raison de ces contraintes technologiques la taille de RAM est modeste; l'E^{2}PROM occupe une portion importante du CHIP. Les écritures en E^{2}PROM sont relativement lentes (de l'ordre de 1 ms par mot mémoire de 32 à 64 octets), et le nombre de ces opérations est limité (de 10^4 à 10^6). L'introduction des mémoires FeRAM devrait amoindrir ces contraintes (10^9 opérations d'écriture, capacités mémoire de l'ordre du Mo, temps d'écriture inférieur à 200ns).
Les classiques processeurs 8 bits ont des puissances de traitements comprises entre 1 et 3 MIPS, ce paramètre est supérieur à 33 MIPS pour les nouvelles architectures à bases de processeurs RISC 32 bits [4].
En termes de puissance de calcul cryptographique, les systèmes 32 bits réalisent un algorithme DES à un Mbit/s et un calcul RSA 1024 bits en 300ms.

Un composant qui résiste aux attaques.

La principale qualité d'une puce sécurisée est d'être *Tamper Resistant*, c'est à dire de résister à des attaques dont l'objectif est de lire tout ou partie des données (des clés d'algorithmes cryptographiques…) stockées dans la *NVM* (Non Volatile Memory, usuellement une E^{2}PROM). Schématiquement on peut classer ces attaques en quatre catégories distinctes [3],

• les attaques par intrusion (à l'aide de microsondes…), qui ont pour objectifs l'acquisition d'informations sur le *design* de la puce (*reverse engineering*) ou sur la structure de son système d'exploitation (lecture optique du contenu de la ROM). L'enregistrement des signaux qui transitent sur le bus d'interconnexion peut permettre de capturer certaines clés (lecture de la NVM) ou de rejouer certaines séquences de commandes du CPU. Des programmes de tests, utilisés dans les phases de contrôles des *wafers*, peuvent être réactivés dans la perpective d'une lecture complète de la NVM.
les attaques logicielles, exploitent les failles de sécurité des protocoles, des algorithmes cryptographiques, et des implémentations logicielles. Par exemple un pirate analyse un protocole d'authentification et casse une clé RSA de 320 bits.

• les attaques par écoutes, qui consistent en l'enregistrement et l'analyse de tous les signaux physiques produits par le processeur au cours de son fonctionnement. Par exemple l'analyse statistique des puissances consommées ([2] DPA - Differential Power Analysis) pendant l'exécution d'une suite connue d'instructions réalisant un algorithme DES, permet de déduire par corrélation (puissance consommée / clé) la valeur de la clé.

• les attaques hardware qui tentent de générer des défauts de fonctionnements inconnus du concepteur de la puce, à l'aide de mécanismes physiques tels que défauts d'alimentation en énergie, en horloge, ou au moyen de sources de rayonnement externes. Typiquement l'attaquant espère modifier une instruction de test ou de branchement conditionnel dans certaines phases de fonctionnement de la puce, afin de lire tout ou partie de la NVM.

• Les contre-mesures tentent d'interdire les attaques par intrusion (par exemple en déposant un treillis métallique sur la puce ou en embrouillant les signaux du bus d'interconnexion), détectent les attaques hardwares à l'aide de différents capteurs (fréquence d'horloge, alimentation, température, ..) ou insèrent des délais aléatoires dans le système d'exploitation. L'implémentation des algorithmes cryptographiques tels que DES peut être adaptée pour être plus résistante aux attaques DPA.

Couches de communications ISO 7816.

Requête.	Réponse.
CLA INS P1 P2 Lc [Lc octets]	sw1 sw2
CLA INS P1 P2 Le	[Le octets] sw1 sw2
CLA INS P1 P2 Lc [Lc octets] CLA C0 00 00 Le	61 Le [Le octets] sw1 sw2

Figure 2: Ordres de commandes (APDUs) définis par la norme ISO 7816.

La norme ISO 7816 [5] décrit l'interface de communication entre la puce et son terminal associé. Ce dernier fournit l'alimentation en énergie et une horloge dont la fréquence est typiquement 3.5 Mhz. L'échange de données entre puce et terminal est assuré par une liaison série dont le débit est compris entre 9600 et 115,200 bauds. Le terminal produit une requête (APDU) qui comporte conformément au protocole de transport T=0 (figure 2) au moins 5 octets (CLA INS P1 P2 P3) et des octets optionnels (dont la longueur Lc est précisée par la valeur de l'octet P3). La carte délivre un message de réponse qui comprend des octets d'information (dont la longueur *Le* est spécifiée par l'octet P3) et un mot de status (sw1 sw2, 9000 notifiant le succès d'une opération) large de deux octets. Lorsque la longueur de la réponse n'est pas connue à priori un mot de status «61 Le» indique la longueur du message de réponse. Une fois ce paramètre connu le terminal obtient l'information au moyen de la commande *GET RESPONSE* (CLA C0 00 00 Le).
Les opérations de lecture et d'écriture, l'invocation des fonctions cryptographiques sont associées à des APDUs spécifiques. L'information contenue dans la puce sécurisée est stockée dans un système de fichiers qui inclut un répertoire racine *(MF Master File*), des sous répertoires (*DF Dedicated File*) et des fichiers (*EF Elementary File*). Chaque composant est identifié par un nombre de deux octets; la navigation à travers ce système s'effectue à l'aide d'APDUs particulières (SELECT FILE, READ BINARY, WRITE BINARY). La sécurité est assurée par des protocoles de simple ou mutuelle authentification (transportés par des APDUs), qui en cas de succès autorisent l'accès aux fichiers.
La mise en œuvre d'une carte utilise donc un paradigme d'appel de procédure, transporté par des APDUs (généralement définies pour un système d'exploitation spécifique); l'information embarquée est connue à priori et classée par un système de fichier 7816.

Intégration des cartes à puce aux technologies de l'information.

L'intégration des puces sécurisées aux technologies de l'information implique l'adaptation des logiciel applicatifs de telle sorte qu'ils génèrent les APDUs nécessaires à l'utilisation des ressources embarquées. Le nombre d'objets informatiques et de puces sécurisées augmentant sans cesse, une interface de plus haut niveau semble nécessaire pour réduire le coût logiciel d'une sécurité à base de cartes.

L'approche classique.

Schématiquement le middleware classique (figure 3) consiste à définir les éléments protocolaires (PE) requis par un service ([6] *application process*) et exécutés dans la puce; chaque élément est associé à une suite d'APDUs ([6] *application protocol)* variable selon le type de carte utilisée. L'application localisée sur le terminal utilise la puce aux moyens d'APIs (Application Programmatic Interface) plus ou moins normalisées (par exemple PC/SC [7] pour les environnements win32 et linux), qui offre une interface de niveau APDUs ou plus élevé (PE).

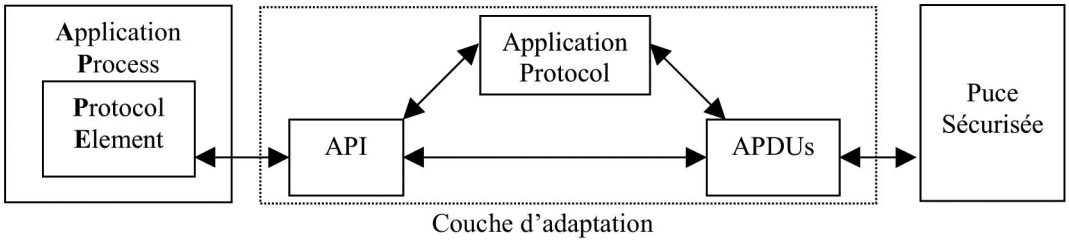

Figure 3: Middleware classique d'une carte à puce.

A titre d'exemple la figure 4 présente l'architecture des systèmes win32 permettant l'appel de fonctions cryptographiques logées dans des cartes à puces. L'application utilise les services offerts par une *CryptoAPI* qui implémente une interface avec des ressources cryptographiques (chiffrement, déchiffrement, génération de clés) fournies par des *cryptographic service provider* (CSP). Ces composants réalisent les procédures cryptographiques (chiffrement, génération de clés…) par voie logicielle (fichiers DLLs), ou à l'aide de cartes à puces (SCSP – Smartcard CSP). Dans ce dernier cas la pile PC/SC [7] permet l'échange de messages entre une puce sécurisée et la *CryptoAPI*. Il devient ainsi possible d'utiliser de manière transparente, depuis un navigateur ou une messagerie électronique, des services cryptographiques embarqués dans une puce.

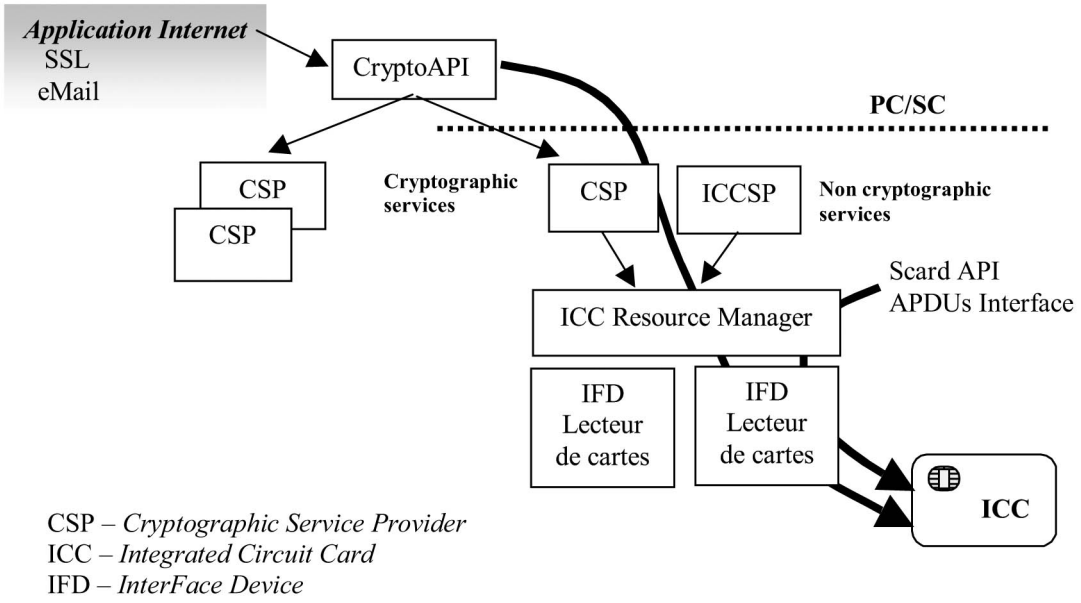

Figure 4: CryptoAPI

L'approche par dictionnaire.

L'interaction avec une puce sécurisée est décrite sous forme d'APIs (réalisant des procédures génériques, telles que authentification, débit, crédit) disponibles sur des plates-formes informatiques ouvertes (java, win32..). Un dictionnaire traduit ces actions génériques en une séquence d'APDUs intelligibles pour une carte particulière. Il est par exemple rédigé dans un langage dérivé du standard XML ([8] SML, SmartX Markup Language); il est chargé depuis un serveur spécialisé après l'identification d'une carte à puce (ce dernier point étant le plus délicat en l'absence de réel standard).

L'approche proxy.

Une application internet par exemple NFS[9], HTTP [10], ou java RMI [11], est adaptée à une puce sécurisée 7816. En règle générale elle fonctionne en mode serveur (au sens TCP), le logiciel réalisant le proxy est localisé entièrement dans le terminal ou réparti entre terminal et puce.
Un proxy NFS [9] réalise le montage du système de fichier d'une carte à puce depuis une machine Unix. A l'aide d'un middleware totalement logé sur le système hôte, il devient possible d'accéder à la puce avec les commandes de gestion de fichier d'un système UNIX.
D'origine SUN la technique UPI ([10] URL Programming Interface) consiste à implémenter dans un système hôte un serveur HTTP dédié *(card server*). Ce dernier traduit une requête HTTP en une suite d'APDUs,

personnalisée pour chaque carte.
Le javacard RMI, introduit par le Java Card Forum [11], utilise deux APDUs (SELECT et INVOKE) pour le support de serveurs RMI dans les cartes java. Le logiciel proxy est réparti entre la carte et le terminal java. Un applet RMI est identifié par un nom de 16 octets et activé au moyen de l'APDU SELECT, qui retourne au terminal le code de *l'interface (le stub)* associé à l'objet embarqué. L'invocation côté terminal, des méthodes associées à l'objet est réalisée par une APDU INVOKE. Cette technique permet l'identification et l'interaction avec des objets logés dans une puce, mais est limitée aux environnements java.

Certains auteurs [13] proposent de loger un serveur WEB dans une carte SIM, et d'émettre des requêtes HTTP vers cette dernière en transitant par un serveur internet qui réalise l'encapsulation du protocole HTTP dans des messages SMS adressés à l'équipement mobile.

L'approche pile de communication.

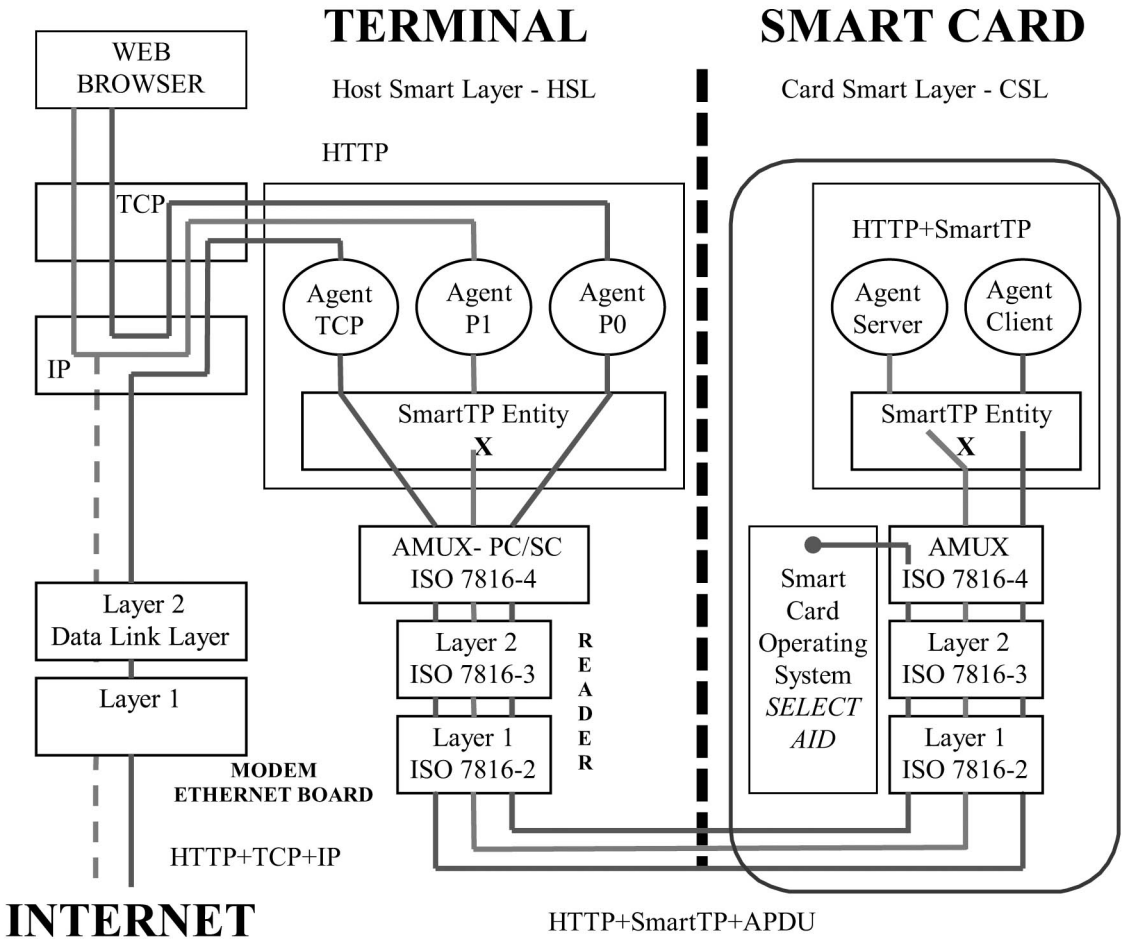

Figure 5: Pile de communication SmartTP d'une carte Internet.

L'objectif d'une telle approche est de loger dans une puce sécurisée des applications client/serveur spécifiées par des RFCs. La pile de communication est répartie entre le terminal et la puce. En effet cette dernière ne possédant aucun moyen physique d'accès à un réseau, le terminal doit fournir cette ressource. Les protocoles de communication définis par l'ISO 7816 ne sont pas full duplex (le paradigme mis en œuvre est de type question/réponse), en particulier parce que la puce ne comporte qu'une seule connexion d'entrée/sortie. Pour ces raisons un protocole tel que SLIP (Serial Line IP) ne peut s'appliquer aux puces. Cependant les applications internet utilisent généralement un paradigme half duplex de type requête/réponse, c'est le protocole TCP qui requiert un mode full duplex, induit en particulier par la réception des paquets d'acquittement (ACK).

Le terminal gère une file d'émission et une ou plusieurs files de réception qui mémorisent et éventuellement trient les paquets destinés à la puce.

Par exemple le CITI [13] a développé une pile expérimentale (webcard) qui comporte une couche IPv4 (avec une adresse fixe) et un sous ensemble du protocole TCP côté carte et un tunnel (file de réception) côté hôte. Cette architecture permet de loger dans la puce un serveur HTTP aux fonctionnalités réduites.

L'affectation d'une adresse IPv4 à une puce est problématique d'une part en raison du nombre important de tels systèmes (un milliard d'unités produites par an), d'autre part parce qu'une adresse fixe restreint l'usage du nœud à une faible région géographique, et enfin parce que l'implémentation d'un client DHCP dans une puce est critique en terme de complexité logicielle et de standardisation. L'association d'une adresse IPv6 aux applications embarquées dans les cartes est d'autant plus attractive que ces dernières possèdent un identifiant dont la taille est identique à celles des adresses IPv6. Le groupe SCP (Smart Card Platform) de l'ETSI, envisage la création d'un groupe de travail pour la définition d'une architecture IP dans les cartes SIM destinées aux réseaux mobiles de troisième génération.

En août 1998 Bull CP8 a déposé un premier brevet [15] relatif à un procédé transformant une carte à puce en un nœud internet; cette technologie a été présentée lors de la dernière conférence JRES'99 [16]. Le premier prototype a été intégré dans une javacard début 1999 [19]. Les acteurs traditionnels de l'industrie cartes semblent intéressés par cette technologie émergente en quête de nouveaux marchés et domaines d'application. Cet intérêt s'est manifesté notamment par deux prix, le premier lors du salon cartes'2000 [25] (Paris), et le deuxième lors des *advanced awards 2001* [26] (Londres). Notre approche consiste à partager la pile IP du terminal, le protocole TCP est traduit en un protocole plus simple SmartTP [37] adapté aux contraintes imposées par l'ISO 7816 et transporté entre terminal et carte par des APDUs.

Des entités logicielles autonomes, que nous nommons agents (figure 5), assurent côté terminal la conversion de protocoles entre la machine d'état TCP (serveur ou client) et celle de SmartTP, et réalisent côté puce les applications internet; ils communiquent à l'aide de SmartTP PDUs. Un agent associé au port TCP P0 (P0=8080) permet d'accéder au serveur WEB (agent WEB) embarqué dans la puce, un agent associé au port TCP P1 (P1=8082) réalise la conversion de messages HTTP en APDUs destinées par exemple à activer une application disponible dans la puce. Un agent carte peut ouvrir une connexion TCP de type client, à l'aide d'une session avec un agent TCP implémenté dans le terminal.

Grâce à ce pont protocolaire la puce embarque des applications serveurs (support natif du protocole HTTP) et clients (LDAP, …); elle partage l'adresse IP du terminal auquel elle est reliée mais dispose d'un ou plusieurs ports TCP que nous espérons standardiser. Nous avons défini deux types d'URLs pour l'identification et l'accès des ressources logées dans une carte internet:

• Des URLs permettant la détection et la sélection d'une application carte particulière, par exemple l'URL http://127.0.0.1:8082/?write=00A40400054A54455354 sélectionne l'applet JTEST (Application IDentifier = 5A 54 45 53 54), en cas de succès on obtient un fichier image de 1 pixel, dans le cas contraire un statut d'erreur HTTP est délivré (c'est une technique que nous nommons **CardBug** [22,23]).

• Des URLs permettant d'accéder à des ressources gérées par une application carte particulière, par exemple l'URL *http://127.0.0.1:8080/Key1=69DA379EF99580A8* permet de réaliser un chiffrement DES du mot 69DA379EF99580A8 à l'aide d'une clé Key1.

Nous remarquerons que l'adresse complète d'une application embarquée comporte trois éléments, le nom de l'application (AID), l'adresse IP du terminal et le port P0.

Architecture logicielle d'une puce internet.

Nous suggérons (figure 6) une architecture logicielle organisée autour de quatre blocs fonctionnels, l'interface de communication, un démon HTTP, un schéma d'organisation de données, et une interface avec le système de fichiers et les procédures cryptographiques qui adapte les mécanismes, usuellement propriétaires, protégeant leurs accès.

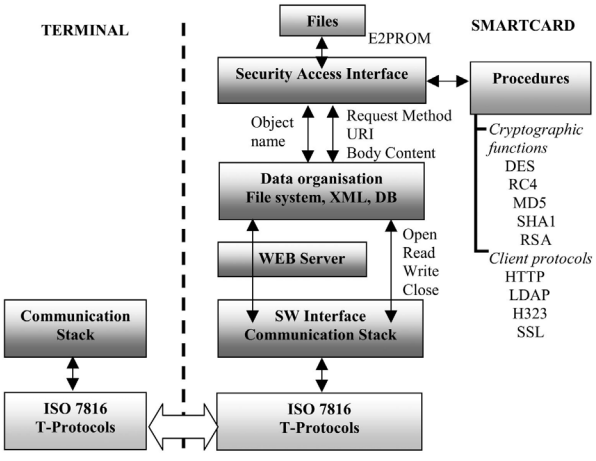

Figure 6: Organisation d'une puce internet

Interface de communication.

L'interface de communication assure le lien entre les applications et la pile de communication. Elle réalise les classiques fonctions Open, Close, Read, Write et permet d'écrire des applications indépendantes des piles de communications qui peuvent être multiples en l'absence de standards; elle permet d'assurer un fonctionnement avec des systèmes d'exploitation (cartes) ou des terminaux hétéroclites. Selon le type de pile utilisée les applications embarquées seront par exemple identifiées par l'adresse IP du terminal et un port TCP particulier (SmartTP), par une adresse IPv4 et un port, ou enfin par une adresse IPv6 et un port TCP.

Daemon HTTP.

Ce composant logiciel traite les messages HTTP, dont il extrait le type de méthode (GET ou POST), le nom de la ressource carte invoquée (URI - Uniform Resource Identifier), quelques entêtes MIME utiles (comme par exemple un cookie) et des données associées optionnelles (BODY).

Organisation des données.

Ce bloc est l'élément central d'une puce internet. Les objets embarqués (fichiers et procédures cryptographiques) et les messages échangés avec ces derniers sont incrustés dans un sous ensemble de l'URI ou transportés par le corps de message (BODY). Les principaux problèmes à résoudre sont l'inventaire des objets présents dans la puce, l'exécution des procédures d'authentification, et enfin le format des messages nécessaires à l'invocation des méthodes. Bien que cette liste ne soit pas exhaustive nous envisageons trois types d'organisation de données, système de fichier, parseur XML, ou base de données.

• Un système de fichiers classique, organisé à la façon des systèmes Unix est une solution qui permet à l'aide de pages hypertexte de browser les ressources embarquées, à partir d'une page de garde dont l'URL est connue. Les procédures sont dans ce cas invoquées au moyen d'un classique mécanisme CGI. La technique du cardbug que nous avons développé [22,23] permet de connaître le contenu d'une puce internet, la présence d'un fichier image (typiquement de 1 pixel), dont le chargement est facilement détecté par un javascript dans une page HTML, indique la disponibilité de ressources particulières.

• L'introduction d'un parseur XML [24] permet d'interagir avec les objets embarqués à l'aide de messages au format XML, réalisant l'appel des procédures embarquées avec leurs paramètres. Un script XML, transporté par le protocole HTTP, est interprété par la puce sécurisée réalisant l'interface avec les méthodes disponibles. Les principaux bénéfices de cette technologie sont la description des ressources embarquées à l'aide de DTD bien connues, et la compatibilité avec les méthodes d'appels d'objets basées sur XML, telles que SOAP [32].

• Les capacités actuelles des cartes à puce permettent d'embarquer des bases de données. Dans le cadre du projet Virtual Campus [14], le laboratoire PRISM étudie ce type de composant (baptisé picoDBMS); la taille de cette application est de l'ordre de 64 koctets (de code byte java), sans les données. La motivation principale d'une telle approche est d'une part d'associer à une classe d'utilisateur des droits plus ou moins limités permettant de construire des vues restreintes des tables relationnelles embarquées, et d'autre part de traiter dans la puce des opérations algébriques (déduites de requêtes SQL) relativement complexes (jointure, projection, restriction…).

Interface d'accès sécurisée aux fichiers et procédures.

Pour des raisons de sécurité ou de conception, certaines données ou procédures sont gérées par les couches basses du système d'exploitation (par exemple un algorithme d'authentification GSM), et sont protégés par des mécanismes divers à base de mot de passe (PIN code etc …), de simple ou de mutuelle authentification. Il est donc nécessaire de disposer d'une interface, compatible avec la description des objets mise en œuvre, qui assure les opérations d'authentification. Par exemple [24] cette interface génère en formulaire HTML qui inclut un lien sur une procédure interne d'authentification et un formulaire qui contient un champ *password*.

Nouveaux usages pour carte à puce internet.

Nous avons préalablement montré que les caractéristiques actuelles des puces sécurisées rendent possible leur connexion au réseau internet ainsi que la définition de schémas de données adaptés à ce dernier. Nous travaillons à l'intégration des cartes dans les nouvelles technologies de l'information, et en particulier nous explorons leur insertion dans le domaine du commerce électronique, de la voix sur IP (et plus généralement dans les protocoles multimédia), des architectures distribuées d'échange de donnée et de la mise en œuvre de la qualité de service (QoS) dans les réseaux sans fils et internet de nouvelle génération.

Commerce électronique.

Le modèle classique du commerce électronique consiste en un échange de données entre un serveur WEB et un navigateur, authentifié par un mécanisme de cookie et sécurisé par le protocole SSL [22]. Nous allons brièvement présenter une analyse critique de ses principales caractéristiques (une description plus détaillée est disponible dans [21]).
Le client (le navigateur) débute une session par un message *ClientHello* qui comporte un *Session_ID* (identificateur de session) dont la valeur est égale à zéro pour une nouvelle session et non nulle pour la reprise d'une session préalablement ouverte. Il propose un choix d'options disponibles, nommé CipherSuites et qui comporte :

• une liste de mécanismes d'échange de clés, en règle générale à base de clé publique RSA. La législation Américaine limite les clés de chiffrement RSA *export* à 512 bits.

• un ensemble d'algorithmes de chiffrement (symétriques) tels que RC2, RC4, DES. Les clés sont limitées généralement à 40 bits, bien que certains logiciels supportent des clés (RC4) de 128 bits.

• une liste de fonctions de digests (usuellement MD5).

Le message comporte également un nombre aléatoire *ClientHello.random* utilisé pour le calcul des clés de session.
Le serveur répond par un ensemble de messages, qui typiquement comporte un nombre aléatoire (*ServerHello.random*), le choix d'un *CipherSuite*, une liste des certificats des clés publiques du serveur, et de manière optionnelle (et rarement utilisée) une demande d'authentification du client, à l'aide d'une clé RSA par exemple.

Le client analyse la validité du certificat du serveur, génère un *pre_master_secret* (un mot de 48 octets) qui est chiffré avec la clé publique du serveur ($pre\_master\_secret^{ServerPublicKey}$). De manière optionnelle le client fournit un certificat, à l'aide duquel il réalise la signature du *pre_master_secret*. Les clés de chiffrement et divers paramètres (IV, MAC Secret...) sont déduits du *master_secret* (obtenu à partir du pre_master_secret) et des nombres aléatoires,

Key(s) = F(master_secret, ClientHello.random, ServerHello.random).

Le serveur calcule les clés et divers paramètres, la session SSL assure dès lors la confidentialité et l'intégrité des données échangées.

Lorsque le Session_ID est non nul, les deux extrémités calculent de nouvelles clés de session à partir des paramètres ClientHello.random, ServerHello.random, L'intégrité des messages est garantie par la conservation des valeurs courantes des fonctions de Hash.

• Le premier point faible du protocole SSL est la liste des possibles algorithmes de chiffrement proposés par le navigateur. Par exemple l'algorithme de chiffrement RC4 (dont la largeur de clé peut atteindre 128 bits) utilise couramment une clé de 40 bits. A raison de 1 million de clés par seconde il faut 12.7 jours pour générer toutes les clés. En juillet 1995, Damien Doligez, doctorant à l'INRIA a cassé un clé RC4 de 40 bits en une semaine, à l'aide d'une centaine de machines générant chacune de l'ordre de 10000 clés/seconde. Un pentium 3, 500 Mhz est capable de produire 200,000 clés par secondes, une demi douzaine de PC suffisent aujourd'hui pour le test d'un million de clés par seconde. Remarquons également que certain navigateurs (et serveurs !) supportent l'absence de chiffrement (*NULL Cipher*) tolérée par le protocole SSL.

• La deuxième action critique est le choix par le serveur de l'algorithme de chiffrement. Par exemple si une alternative se présente entre une clé de 128 bits et 40 bits, le serveur peut opter pour la deuxième solution.
La troisième opération délicate est la vérification du certificat du serveur, qui est un aspect clé de la sécurité. Il est en effet nécessaire de faire confiance au code du navigateur qui analyse le certificat.

• Enfin la taille de la clé publique proposée par le serveur pour le chiffrement est un facteur sensible. Pour mémoire, en Août 1999 le laboratoire hollandais CWI a réalisé la factorisation d'une clé RSA 512 bits à l'aide d'une puissance de calcul de l'ordre de 8400 MIPS-an, fournie par 300 ordinateurs et stations de travail en 7 mois.

Un proxy SSL logé dans une puce évite ces faiblesses, sous l'hypothèse d'accorder sa confiance à l'émetteur de la carte. Par exemple une URL (carte) *http://127.0.0.1:8080/?ssl=www.bank.fr/urienp* réalise l'ouverture d'une session SSL avec le site *www.bank.fr* depuis une puce sécurisée. Elle est dès lors gérée par la carte, qui en particulier réalise la vérification du certificat, et peut authentifier le client à l'aide d'un certificat enregistré dans la puce.

Nous soulignerons à ce propos que dans le protocole SSL l'authentification du client est optionnelle, et rarement mise en œuvre (le stockage d'une clé privé étant problématique en l'absence de carte à puce). Une session entre l'internaute et un site, est une succession de requêtes HTTP disjointes (chiffrées par des sessions SSL non liées les unes aux autres), identifiées par un *cookie*. Le premier cookie est délivré en réponse à l'envoi d'un formulaire qui comporte typiquement un login et un mot de passe; SSL embarqué dans une puce assure la protection optimale de tels éléments.

L'implémentation d'une couche SSL implique la disponibilité dans la puce de fonctions SHA-1 et MD5, RC4, d'un parseur ASN.1 de certificat X.509, de clés publiques d'autorité de certification, et d'algorithme de calcul RSA.

La principale contrainte de réalisation du protocole SSL est de disposer d'une capacité mémoire suffisante pour embarquer le code nécessaire; nous estimons que cette taille est inférieure à 64 ko et nous pensons disposer d'un premier prototype courant 2002.

VoIP - Enregistrement Localisation Signalisation Confidentialité.

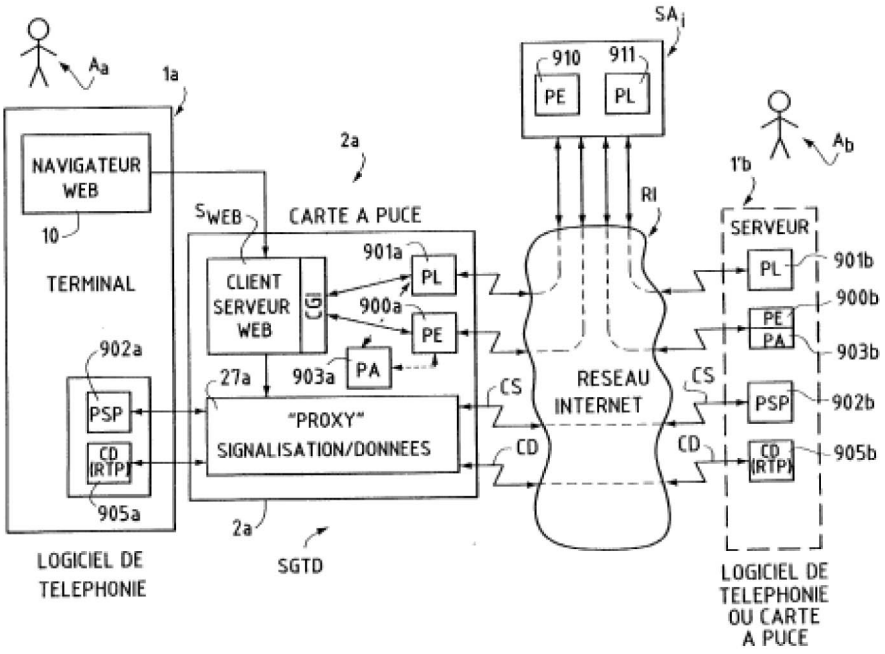

Figure 7: Exemple d'adaptation d'une carte Internet à un logiciel VoIP d'après [17,18]

L'échange de données multimédia, en particulier la conduite d'appels téléphoniques à travers les réseaux IP fixes et mobiles, est un marché attractif pour une économie de réseaux en quête de rentabilité. Outre les problèmes de qualités de services requises et de tarification (le réseau peut il fournir la bande passante nécessaire et si oui à quel prix), l'ouverture d'une session multimédia implique la mise en place d'une architecture comportant les éléments suivants:

• Des serveurs de localisation, qui assurent la correspondance entre un identifiant *universel* d'un internaute (adresse courrier – dupont@freemail.com, adresse SIP dupont@hotsip.com), et l'adresse IP du terminal informatique auquel il accède. En conséquence un utilisateur doit s'enregistrer sur un tel annuaire en utilisant un protocole spécifique (PE – protocol d'enregistrement), qui de manière optionnelle met en œuvre une procédure d'authentification. Symétriquement la recherche d'un correspondant implique la consultation d'un ou de plusieurs annuaires à l'aide d'un protocole de recherche (PR) adéquate. Dans ce contexte l'internaute effectue un nouvel enregistrement à chaque fois qu'il change de terminal (ordinateur personnel, assistant, téléphone mobile, terminal sans fils …)

• Un protocole de signalisation (PS) qui initialise et supervise la session (sonnerie, prise d'appel, abandon). L'authentification du correspondant peut être une condition préalable à l'ouverture de la communication.

• Un canal de données (CD) et des protocoles associés (RTP) qui gère le flux multimédia dont il assure de manière optionnelle l'authentification, la confidentialité, et l'intégrité.

Il existe aujourd'hui deux architectures possibles pour la mise en œuvre de la téléphonie sur IP, H323 [38] et SIP [33].

Dans les implémentations actuelles de H323 (netmeeting, …) le service d'enregistrement/localisation est assuré au moyen d'annuaires LDAP[30]. La sécurité des opérations LDAP est de manière optionnelle assurée par un mécanisme de type SASL (*Simple Authentication and Security Layer* [29]) qui permet d'ajouter des fonctionnalités d'authentification (Kerberos, S/KEY,…), de chiffrement ou d'intégrité de données (par exemple en protégeant la session LDAP à l'aide d'une couche SSL ou TLS). Le standard H.235 [39] introduit des opérations d'authentification et d'intégrité relatives aux messages RAS, H.225 (signalisation) et aux canaux logiques H.245. Le canal de données peut être également chiffré à l'aide de clé de session éphémères (DES, RC2, triple DES …).

L'architecture SIP utilise de manière native des serveurs de localisation, supportant divers schémas de chiffrement de messages et d'authentification, importés de protocoles tels que HTTP (basic schema, digest schema,…) ou PGP (auto authentification à l'aide d'une clé privé RSA). La sécurité du canal de données peut être assurée par la protection des paquets RTP grâce à des procédures importés de IPSEC ou H.235.

Dans l'état de l'art actuel il n'existe pas d'interface programmatique permettant d'intégrer les cartes à puces aux logiciels de téléphonie. Les problèmes à résoudre sont la nature de l'interface avec la puce, l'identification des algorithmes cryptographiques hébergés, ainsi que des clés d'accès aux services. Nous suggérons d'utiliser un port dédié TCP, et un serveur WEB embarqué. La future puce de téléphonie IP (que nous baptisons SIM-IP) mémorise le profil de l'abonné (PA), peut exécuter des protocoles d'enregistrement et de localisation (tels que LDAP ou SIP) et les procédures de sécurité nécessaires, réaliser tout ou partie du protocole de signalisation, stocker des clés additives (utiles par exemple au chiffrement du canal de données).

A titre d'illustration la figure 7 présente une architecture de carte SIM-IP, adaptée aux logiciels de téléphonie actuels sans API carte spécifique. Le composant intègre pour l'essentiel le profil d'abonné et des protocoles d'enregistrement/localisation. On peut également envisager d'utiliser la puce sécurisée de manière analogue au GateKeeper présent dans l'architecture H323.

Il est aujourd'hui possible d'intégrer dans des puces sécurisées des protocoles d'enregistrement/localisation tels que SIP ou LDAP. Cependant les très faibles débits supportés par ces composants ne permettent pas de réaliser le traitement direct du flux multimédia.

Architecture distribuées – identification des données et mécanismes d'interaction.

Dans l'approche classique les données sont classées dans un système de fichiers qui comporte un répertoire racine (MF - Master File) des sous répertoires (DF - Dedicated Files) et des fichiers (EF - Elementary File). Ces différents éléments sont identifiés par un nombre de deux octets; la navigation sur le système de fichier est réalisée à l'aide d'APDUs telles que SELECT FILE, READ BINARY, WRITE BINARY; l'appel de procédures cryptographiques est également associé à des APDUs particulières. La nature de l'information stockée dans la puce est réalisée visuellement par les informations imprimées sur le support en PVC des cartes mobiles, cette identification n'est pas utile pour les cartes fixes associées de manière permanente à un équipement (comme par exemple les cartes SIM).

Dans l'approche carte internet, la puce est un nœud virtuel du réseau, elle se comporte comme un serveur ordinaire capable de recevoir et de traiter des messages destinés aux objets embarqués. De plus en plus la syntaxe XML s'affirme comme un standard incontournable tant pour le transport de messages (SOAP, XML-RPC, …) que pour la description des interfaces des objets traitant ces messages. Nous avons récemment présenté une architecture de puce internet [24], organisée autour d'un parseur XML, capable d'interpréter des scripts XML basés sur des éléments vides (*empty tags*).

Nous suggérons d'adapter tout ou partie du protocole SOAP [32], basé sur un transport HTTP de requêtes XML, pour intégrer les cartes à puces aux architectures distribuées, en particulier en définissant d'une part un fichier DTD relatif au protocole de transport (la grammaire de l'enveloppe *SOAP:ENVELOPPE* des messages), et d'autre part un DTD (référencé par le corps – *SOAP:BODY* du message) décrivant l'interface (les procédures et variables) associée à un objet.

QoS, qualité de service.

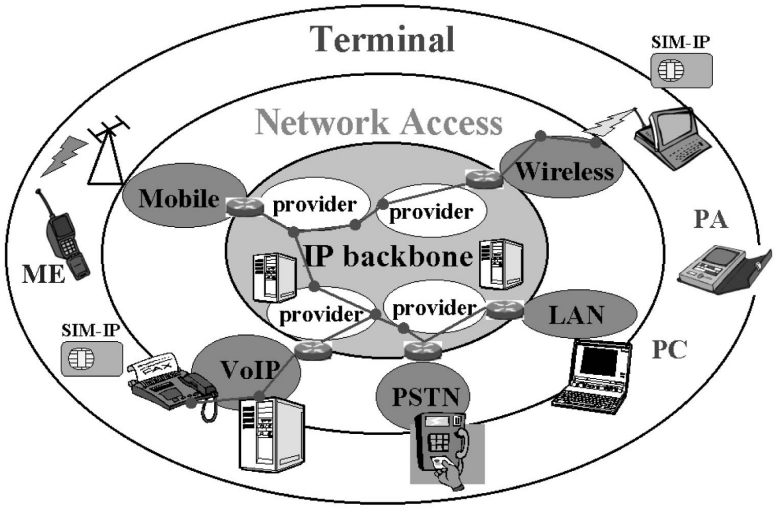

Figure 8: QoS et cartes SIM-IP

Le réseau internet de nouvelle génération (figure 8) comportera un cœur (backbone) constitué de plusieurs régions (ou domaines) contrôlées par différents prestataires de services (provider). Il semble probable que la gestion globale de la qualité de service, c'est à dire un ensemble de paramètres tels que débit, temps de traversé et gigue (jitter) sera assurée par un mécanisme de type *differentiated service* [31], dont le principe consiste à inclure dans chaque paquet IP un champ (traité par les routeurs des domaines traversés) spécifiant la classe de QoS associée.

Le cœur du réseau IP de nouvelle génération est accessible à partir de réseaux d'accès hétéroclites tels que réseaux d'entreprise, réseaux de téléphonie mobile (GSM, GPRS, UMTS), réseaux sans fils divers (802.11, …).

L'utilisateur dispose d'un ou plusieurs terminaux, chacun d'entre eux pouvant supporter de multiples interfaces de communication (par exemple GSM, GPRS, UMTS, 802.3, 802.11; bluetooth, IrDa …).
La gestion de la qualité de service de bout en bout est dès lors un processus complexe, qui implique le traitement des paquets IP échangés depuis deux terminaux à travers des réseaux d'accès et un ensemble de domaines du cœur.

De manière analogue aux cartes SIM (GSM 11.11) qui stockent l'identité d'un abonné (IMSI), une clé (Ki) et une procédure d'authentification (A3+A8), un utilisateur est associé à un profil (qui définit ses droits en fonction des services auxquels il a souscrit) comportant une clé et une procédure d'authentification.

L'ouverture d'une session nécessitant une certaine qualité de service, implique les opérations suivantes,

• une phase d'authentification avec le réseau d'accès

• la recherche du correspondant, à l'aide par exemple de protocoles tels que SIP, destinés à localiser ce dernier sur le réseau et identifier le type et les capacités de terminal dont il dispose. Cette opération peut inclure une procédure d'authentification.

• La négociation depuis le réseau d'accès d'une qualité de service de bout en bout qui pourra impliquer une phase d'authentification avec les serveurs de politiques (policy servers). Par exemple une puce sécurisée supporte tout ou partie d'un protocole d'allocation de ressources tel que COPS[35].
Une fois la qualité de service négociée le flux de données échangé par les extrémités de la session doit être identifié sur le chemin qui mène des terminaux aux routeurs de frontières du réseau d'accès (*edge routeur*), par exemple en utilisant à l'aide des mécanismes de signature de paquets tels que ceux spécifiés dans IPSEC (AH, …).

Les messages relatifs à la négociation de la QoS, sont transportés des protocoles RSVP [34] ou COPS [35] et authentifiés à l'aide d'éléments appelés *INTEGRITY OBJECT*. Ces derniers réalisent l'empreinte (*MD5 digest*) du message concaténé à une clé secrète. En première approche la gestion de la qualité de service depuis une puce internet consiste à intégrer dans celle-ci tout au partie du protocole COPS, et en particulier des clés d'authentification. Le mécanisme de distribution de telles clés symétriques n'est pas aujourd'hui défini, on peut par exemple envisager un schéma de pré-paiement (cartes rechargeables), ou la mise en place d'une architecture de type PKI assurant l'authentification de l'utilisateur et le paiement du service.

Une architecture de gestion de la QoS, basée sur des puces sécurisées sera étudiée dans le cadre du projet MMQoS (Maîtrise de la mobilité et de la qualité de service dans la 4°Génération de mobile) labellisé en mai 2001 par le RNRT (Réseau National de la Recherche en Télécommunications).

Conclusion.

Dans cet article nous avons tenté de présenter les différentes perpectives d'intégration de puces internet dans les réseaux mobiles et internet de nouvelles génération. Les capacités de ces composants en termes de taille mémoire et de puissance de traitement, permettent aujourd'hui leur utilisation comme un véritable co-processeur de sécurité, par exemple identifié par l'adresse IP et un port TCP spécifique de la machine à laquelle il est connecté. La sécurité est un paramètre clé pour les services transportant des informations sensibles ou impliquant des mécanismes de paiement. Bien que la plupart des protocoles définissent nativement des éléments de sécurité, ils n'ont pas envisagé jusqu'à présent l'usage de puces sécurisées. Nous espérons que la nécessaire définition d'un modèle viable pour l'économie de réseau sera un moteur pour la définition des architectures des puces internet en terme notamment de pile de communication et d'organisation des données.

Références

[1] L.C.Guillou, M.Ugon, and J.J Quisquater, "The smard card: A standardized security device dedicated to public cryptology", In Contemporary cryptology - The science of information integrity (1992), G. J. Simmons, Ed., IEEE Press, pp. 561-613.

[2] P.Kocher, J.Jaffe, and B.Jun "Differential Power Analysis" Proceedings of CRYPTO'99, Springer-Verlag, August 1999, pp 388-397.

[3] O.Kommerling, M.G.Kuhn. Design Principles for Tamper-Resistant smartcard Processors. Proceedings of the USENIX Workshop on Smartcard Technology Smartcard'99 Chicago Illinois USA May 10-11 1999, pp 9-20

[4] J.P. Tual. "MASSC : A Generic Architecture for Multi application Smart Cards". IEEE Micro journal N°0272-1739/99, 1999

[5] International Organization for Standardization (ISO). "Identification cards - Integrated circuit(s) card with contact". ISO/IEC 7816.

[6] Open Card Framework – OCF. http://www.opencard.org.
[7] “Interoperability Specification for ICCs and Personal Computer Systems”, PC/SC, © 1996 CP8 Transac, HP, Microsoft, Schlumberger, Siemens Nixdorf.
[8] X.Lorphelin. “InternetandSmartCardApplicationDeployment”.http://www.smartcardcentral.com/technical/articles/jsource/article9907.pdf
[9] N. Itoi, P. Honeyman, and J. Rees. “SCFS: A UNIX Filesystem for Smartcards”. USENIX Workshop on Smartcard Technology, Chicago (May 1999).
[10] Jon Barber “The Smart Card URL Programming Interface”, Proceedings of Gemplus Developer Conference (GDC’99), Paris, France, 21-22 June 1999.The Smart Card URL Programming Interfcace”, Proceedings of Gemplus Developer Conference (GDC’99), Paris, France, 21-22 June 1999.
[11] Sun Microsystem. “Java Card™ 2.2 Remote Method Invocation Design. Draft 1 Revision 1.1”. May 07th 2001
[12] S.Guthery, R.Kehr, J.Posegga “How to turn a GSM SIM into a Web Server”. Smart Card Research and Advanced Application Conference CARDIS’2000, Bristol UK pp 209-222.
[13] J. Rees and P. Honeyman, “Webcard: a Java Card web server,”. Smart Card Research and Advanced Application Conference CARDIS’2000
[14] Christophe Bobineau, Luc Bouganim, Philippe Pucheral, Patrick Valduriez, “PicoDBMS: Scaling down Database Techniques for the Smartcard” VLDB’2000 26th International Conference on Very Large Database
[15] “Procédé de communication entre une station d’utilisateur et un réseau, notamment du type INTERNET, et architecture de mise en œuvre” - Inventeur Pascal Urien – Brevet N°98/10401déposé le 13 août 98 -
[16] Pascal Urien - Hayder Saleh “Une nouvelle approche de la carte à puce réseau” - JRES 99 - 3°Journées Réseaux Ministère de l’Education Nationale de la Recherche et de la Technologie - Montpellier - Décembre 1999.
[17] “Procédé d’enregistrement d’un usager sur un serveur d’annuaire d’un réseau de type Internet et / ou de localisation d’un usa ger sur ce réseau, et carte à puce pour la mise en œuvre du procédé” . Inventeur Pascal Urien - Brevet N°00/01664, déposé le 10 février 2000
[18] “Procédé de gestion de transmissions de données multimédias via un réseau de type Internet, notamment de données télépho niques ou visiophoniques, et carte à puce pour la mise en œuvre du procédé”. Inventeur Pascal Urien – Brevet N°00/01663, déposé en France le 10 février 2000
[19] Pascal Urien “ Internet Card, a smart card as a true Internet node”, Computer Communication, volume 23, issue 17, Octobre 2000.
[20] Pascal Urien, Hayder Saleh, Adel Tizraoui “Internet Card, a smart card for internet”, Protocols for Multimedia Systems (PROMS) Cracow Poland, October 22-25 2000.
[21] Pascal Urien, Hayder Saleh, Adel Tizraoui “SSL dans une carte à puce” Journées Doctorales Informatique et Réseaux, JDIR’2000, Ministère de la Recherches, Paris, 6-8 novembre 2000.
[22] Pascal Urien - Hayder Saleh - Adel Tizraoui. “Authentification dynamique par carte à puce internet, une possible alternative à l’usage polémique des cookies et des WebBugs”. Infosec’Com 2001 – 29,30,31 mai 2001 la défense
[23] Pascal Urien, Hayder Saleh, Adel Tizraoui , “XML Smartcards”, IEEE International Conference on Networking, ICN’01, July 11-13, 2001 - CREF, Colmar, France.
[24] Pascal Urien “Programming internet smart card with XML scripts”, e-Smart 2001Cannes – 19-21 Septembre 2001.
[25] http://www.cartesexpo.com/frame_gb/sesame.htm
[26] http://www.smart-ventures.com
[27] RFC 1633 Integrated Services in the Internet Architecture: an Overview June 1994
[28] RFC 2205 Resource Reservation Protocol RSVP September 1997
[29] RFC 2222 Simple authentication and Security Layer (SASL) October 1997.
[30] RFC 2251 LDAP v3 December 1997.
[31] RFC 2475 An Architecture for Differentiated Services December 1998
[32]]Simple Object Access Protocol SOAP, http://msdn.microsoft.com/xml/c-frame.htm?/xml/general/soapspec.asp
[33]]RFC 2543, “SIP: Session Initiation Protocol,” - March 1999.
[34] RFC 2747, RSVP Cryptographic Authentication. January 2000
[35] RFC 2748 The COPS(Common Open Policy Service) Protocol January 2000
[36] Extensible Markup Language (XML) 1.0 (Second Edition), W3C Recommendation 6 October 2000.
[37] Internet draft “*Smart Transfer Protocol*”, draft-urien-smarttp-00.txt, June 2001, available at www.ietf.org.
[38] International Telecommunication Union (ITU) Recommendation H323, “Packet-based multimedia communication systems”, October 1997.
[39] International Telecommunication Union (ITU) Recommendation H.235 “Security and encryption for H-Series (H.323 and other H.245-based) multimedia terminals”
[40] International Organization for Standardization (ISO) “Identification cards – Integrated circuit(s) cards with contacts ISO/IEC 7816–Part 7: Inter industry commands for structured Card Query Language (SCQL).” 1997