La Voix sur IP (VoIP) : une opportunité pour la sécurité ?

Nicolas Dubée

Secway, France
ndubee@secway.fr
http://www.secway.fr/

Résumé Principalement représentés par SIP, les nouveaux protocoles de VoIP et leurs implémentations comportent un certain nombre de vulnérabilités, notamment liées à leur complexité. L'une d'entre elles est l'absence de standard clairement établi pour la négociation des clefs de chiffrement de la voix, débouchant dans les gros déploiements sur d'importantes lacunes en terme de confidentialité des conversations. Nous avons développé un boitier matériel pour le chiffrement des conversations VoIP selon le standard ZRTP.

Mots-clef : Sécurité et confidentialité des protocoles VoIP en SIP / SCCP / H.323 et RTP

1 Préambule

Si la sécurité des systèmes d'information est souvent vue comme une contrainte, c'est probablement parce que trop souvent, ses spécialistes oublient qu'après l'exposition des vulnérabilités, leur rôle est de fournir des solutions. Adaptées aux besoins des utilisateurs, efficaces et économiques, ces solutions permettent d'encadrer de manière sécurisée le déploiement et l'utilisation de nouvelles technologies. Cette sécurité là est force d'avancées, d'accompagnement, et non de blocage.

2 Introduction

Alors que se concrétisent les premiers déploiements massifs de voix sur IP (*VoIP*) en entreprise, les enjeux liés à la confidentialité des flux sont (temporairement !) relégués au second plan par rapport aux légitimes problématiques de disponibilité du service. Pourtant, les risques d'interception de la VoIP existent bel et bien, mais des réponses supérieures à celles disponibles dans le monde de la téléphonie « classique » peuvent y être apportées, en bénéficiant du savoir-faire sécuritaire acquis dans le monde des réseaux de données. La société Secway a décidé de répondre à ce besoin en concevant un boitier matériel de chiffrement poste-à-poste des conversations VoIP. Ce projet sera l'occasion d'un exposé sur les enjeux actuels de la VoIP, ainsi que sur la réalisation de projets de sécurité sous environnements ouverts. Nous présenterons dans ce document un retour d'expérience quant au développement d'applications de sécurité embarquées sur tandem ARM + GNU/Linux ; nous couvrirons les contraintes qui ont été découvertes ainsi que les choix retenus pour arriver au produit fini.

3 Aperçu des protocoles de VoIP

3.1 Protocoles de signalisation et protocoles de données

Une des raisons justement invoquées pour expliquer la méfiance et les retards de déploiements VoIP est la complexité des protocoles sous-jacents, complexité présente dans les protocoles eux-mêmes mais surtout dans l'abondance et l'enchevêtrement de leurs variantes et extensions.

De grandes tendances peuvent cependant se dégager dans l'offre actuelle. En effet, il semblerait que dans le cadre de déploiements VoIP en entreprise, les protocoles SIP, H.323 et Skinny (SCCP du constructeur Cisco) soient les plus répandus.

Il convient de préciser ce que l'on entend par « protocole VoIP ». Une communication en VoIP fait intervenir deux étapes : une première d'établissement et négociation de la conversation, suivie d'une étape d'échange de datagrammes contenant les données à véhiculer, telles que la voix ou la vidéo.

Cette communication de données temps réel se fait en général de pair à pair (téléphone à téléphone). Les protocoles correspondant à ces deux étapes sont en général désignés par les termes protocoles de signalisation d'une part, protocoles de données ou protocoles temps réel d'autre part. Tout échange VoIP fait intervenir ces deux types de protocoles ; tout protocole VoIP est soit de signalisation, soit de données.

3.2 Protocoles de données : la suprématie du RTP

Concernant les protocoles de données, un monopole de fait existe au profit de RTP, le *Realtime Transport Protocol*. Conçu par le groupe Audio/Vidéo de l'IETF et publié dans les RFC 1889 puis 3550, RTP définit un protocole de transport de données temps réel (voix, vidéo) au dessus d'UDP[1]. Les données en question sont transmises dans des paquets UDP, précédées d'un en-tête applicatif minimal, contenant des informations de synchronisation du flux. Les informations de contrôle qualité sont véhiculées dans des paquets RTCP (*Realtime Transport Control Protocol*). Si son nom peut effectivement être trompeur, RTCP n'a aucun rapport avec TCP et n'est aucunement la version « TCP » du RTP. D'autre part, ni RTP ni RTCP n'offrent de garanties quant aux délais de transmission ; c'est aux protocoles de niveaux inférieurs et aux équipements traversés de s'en charger. Notons par ailleurs qu'outre son application en unicast dans la voix sur IP, RTP a de nombreuses autres utilisations sur les réseaux IP, certaines multicast, comme dans la vidéo des offres Internet triple-play.

3.3 Protocoles de signalisation

RTP et RTCP s'occupent uniquement de la diffusion des données temps réel. Ils ne véhiculent aucune information relative à l'établissement de la conversation. Ils ne reposent d'ailleurs même pas sur des ports standards, la seule contrainte étant selon les RFC qu'un port dynamique de numéro pair est choisi au démarrage de chaque conversation pour RTP, tandis que le flux RTCP relatif à l'échange est transmis sur le port impair immédiatement supérieur.

L'établissement de la conversation, la négociation de ses paramètres (dont les codecs utilisés, mais aussi les numéros de ports RTP/RTCP) sont l'apanage des protocoles de signalisation. Suivant l'architecture VoIP retenue, tel ou tel protocole de signalisation est utilisé. Les protocoles de

[1] RTP peut être utilisé au dessus d'autres protocoles de transport qu'UDP, mais dans le cas de réseaux IP, l'UDP est le plus courant.

signalisation les plus courants sont : SIP (standard s'imposant progressivement face aux autres), H.323 (importante utilisation chez les opérateurs) ou SCCP (alias *Skinny*, protocole propriétaire Cisco employé dans les architectures à base d'équipements de ce constructeur).

SCCP et H.323 sont des protocoles anciens, composés de datagrammes binaires, relativement simples à analyser d'un point de vue informatique. Encore très utilisé, notamment dans le monde des opérateurs de télécom, H.323 est en fait un ensemble de protocoles issu de l'ITU-T, et appliquant l'ensemble des fonctions requises pour des communications audio-vidéo au monde des réseaux par commutation de paquets. On rappelle à ce propos que H.323 fait partie de la série des protocoles ITU-T H.32x, qui décrivent les communications sur les réseaux de type PSTN[2], ISDN...

SIP hérite quant à lui d'années d'expansion du web, et est à ce titre un protocole texte volontairement très semblable à HTTP, avec lequel il partage même ses codes d'erreur (`200 OK`, `404 Not found` etc...). Dans sa version actuelle (SIP v2, RFC 2543 sorti en 1999), les requêtes sont échangées sur le port 5060, TCP ou UDP. La sélection du protocole de transport se fait comme pour DNS : UDP est par défaut utilisé, le protocole TCP est mis en oeuvre dès lors que la requête dépasse la taille du MTU moins 200 octets[3]. Un client SIP (aussi appelé *User-Agent Client* ou UAC dans la littérature SIP, par exemple un téléphone VoIP) envoie à une entité serveur (*User Agent Server*, UAS, par exemple un autre téléphone VoIP) des requêtes SIP qui contiennent, sous forme de texte UTF-8, un verbe indiquant l'action à effectuer (`REGISTER`, `INVITE`, `ACK`, `BYE`, `CANCEL`, `OPTIONS`) suivi de multiples en-têtes. Un ou plusieurs messages contenant un code d'état (numéro et description textuelle, comme en HTTP) sont retournés par l'UAS. Cet échange SIP cherche à établir une session, dans notre cas une conversation VoIP, entre l'UAC et l'UAS. La négociation des caractéristiques de la conversation elles-mêmes est effectuée dans le payload des paquets SIP échangés. Ces caractéristiques sont décrites selon le protocole SDP (*Session Description Protocol*) : SIP sert en fait ici de véhicule pour des échanges d'informations en SDP. Si deux téléphones SIP pourraient théoriquement communiquer en direct, sans élément intermédiaire, en connaissant leurs IP respectives, cette approche est irréaliste. Des éléments particuliers ont donc été définis, avec pour rôle l'authentification, le routage des requêtes et la vérification de leur validité, en terme d'autorisations de services dont dispose l'UAC. Ces UAS sont appelés des *proxy servers* ; lorsqu'ils supportent les requêtes `REGISTER`, on les appelle aussi *SIP registrars*. En pratique, lorsqu'un téléphone rejoint le réseau SIP dont il est membre, il va s'inscrire auprès du registrar. Cette inscription permet au registrar et aux autres proxy servers de le localiser et de router les requêtes en conséquence. Lorsqu'un téléphone SIP cherche à joindre ce nouveau membre, la requête SIP `INVITE` est envoyée au proxy server le plus proche, puis routée jusqu'au téléphone destination. Les données SDP de ces paquets `INVITE` circulant depuis l'appelant vers l'appelé, au travers des différents proxy servers, contiennent l'ensemble des informations nécessaires pour établir la conversation.

3.4 Autres protocoles

Quoiqu'il en soit, ces deux protocoles ne représentent qu'une infime partie des technologies utilisées par la VoIP, où des services tels que le 802.1x, le DHCP, le TFTP sont aussi présents.

[2] *Public Switched Telephone Network*. Le réseau téléphonique traditionnel mondial par commutation de circuits, utilisant principalement les protocoles de la famille SS7.

[3] La RFC ne dit cependant pas comment déterminer le MTU pour l'ensemble du parcours ; dans ce cas, la limite est fixée à 1300 octets. Les 200 octets de différence s'expliquent par le fait qu'un peu de place en plus est prévue pour la réponse, afin d'éviter un changement de protocole de transport en cours d'échange SIP...

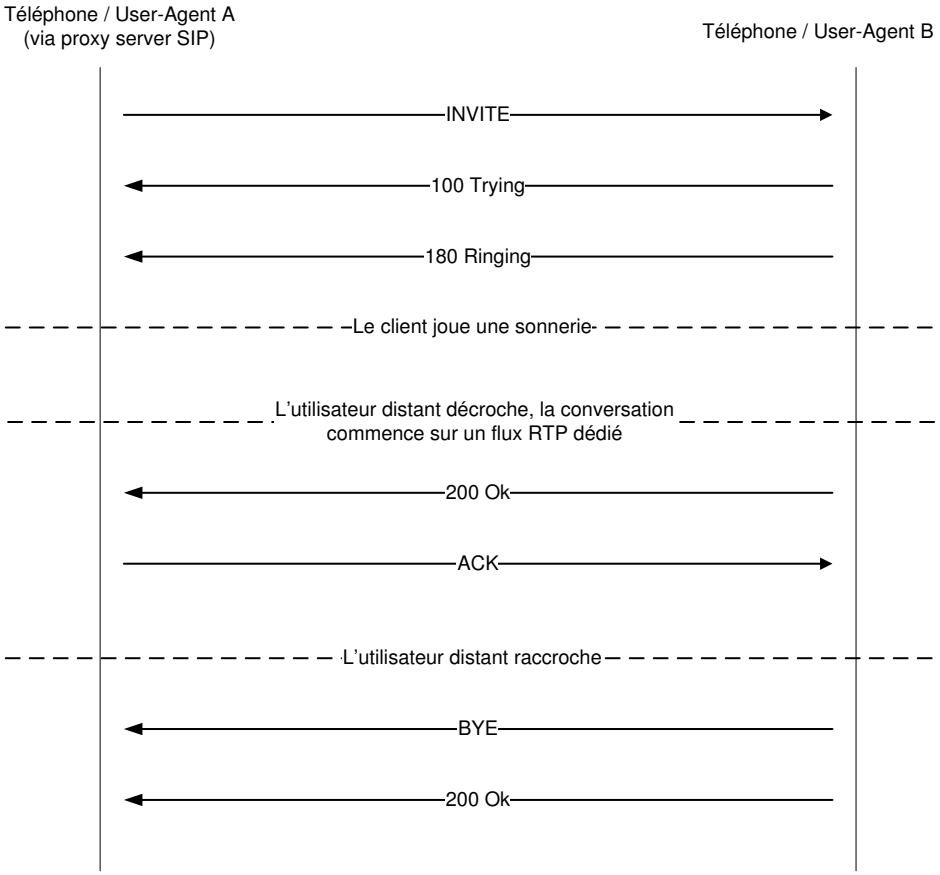

FIG. 1: Chronologie des messages SIP envoyés d'un user-agent A vers un user-agent B dans le cadre d'un établissement de conversation.

Ainsi, dans la plupart des déploiements, l'initialisation du téléphone se fait au moyen des protocoles DHCP/BOOTP, TFTP pour le téléchargement des configurations. TFTP est aussi mis en oeuvre dans les phases de mise à jour des firmwares. Les téléphones étant des entités IP, ils disposent de leur propre stack, embarquant généralement serveurs http, telnet et snmp pour l'administration distante. Lors d'interconnexions de réseaux SIP, la résolution des proxy servers SIP distants est effectuée par DNS, à la manière des records MX pour le SMTP. Supposons que le proxy server SIP d'une organisation cherche à localiser le proxy server d'une autre organisation, pour permettre l'appel d'un abonné de cette deuxième organisation, identifié par son URI `sip:john.doe@example.com`. Une requête DNS SRV portant sur `_sip._udp.example.com` sera émise par le proxy server de l'appelant. De la même manière, si l'appelant cherche à établir une session SIP sécurisée vers le

proxy server de l'organisation `example.com` (on verra qu'il est effectivement possible de tunneler SIP au travers de TLS), une requête DNS SRV portant sur `_sip._tls.example.com` sera effectuée.

3.5 Résumé

Les protocoles mis en oeuvre dans la Voix sur IP sont donc de deux types :

- **Les protocoles de signalisation, s'occupant notamment de la notification de présence, de l'établissement des conversations, de la négociation du trajet des flux de données temps réel et des IP/ports utilisés pour la communication.** Les protocoles les plus répandus pour ce faire sont SIP (standard IETF supposé devenir la référence à moyen terme, messages échangés en texte sur de l'UDP ou TCP), H.323 (famille de protocoles de l'ITU-T, beaucoup plus complet que SIP car implémentant l'ensemble des notions telco), SCCP (protocole propriétaire Cisco).

- **Les protocoles de données, s'occupant de la diffusion de flux temps réels et du contrôle de cette diffusion.** Le plus courant de ces protocoles est RTP, regroupant à la fois RTP lui-même (diffusion du flux) et RTCP (contrôle de la diffusion).

En pratique, de nombreux autres protocoles interviennent en support des précédents. Ainsi, DNS est utilisé pour la résolution des proxy servers distants, dans le cadre de communications entre organisations. Les infrastructures VoIP sont d'autre part très souvent placées dans des VLAN séparés de la bureautique. Dans ce cas, les technologies classiques de VLAN 802.1Q sont mises en oeuvre. Sur le VLAN VoIP circulent aussi des échanges DHCP, BOOTP et TFTP, pour l'assignation des adresses et le téléchargement des configurations. Finalement, des flux Web, Telnet, SNMP, peuvent aussi être présents dans le cadre de l'administration des user agents SIP.

4 Un exemple de déploiement de VoIP

Un exemple de déploiement VoIP classique (mais très modeste !) pourrait être le suivant : téléphones du vendeur X, PaBX IP (*IPBX*) basé sur une souche open-source Asterisk[4]. Afin d'éviter de nouveaux câblages, la VoIP utilise les infrastructures physiques du réseau corporate de la société. La séparation est faite logiquement, au moyen de VLAN (protocoles 802.1Q) séparés pour le trafic de voix et les échanges usuels. Chaque téléphone intègre pour ce faire un mini-switch, sur lequel l'utilisateur branche son poste PC ; l'étiquetage VLAN 802.1Q est donc fait par le téléphone lui-même. Les téléphones démarrent et obtiennent une adresse IP en DHCP, ainsi que leur configuration via BOOTP.

Le protocole SIP est utilisé pour la négociation des appels. Après négociation SIP auprès du PABX, la voix est transmise directement via le protocole RTP entre les deux postes communicant ensemble.

La VoIP n'est pour l'instant mise en oeuvre que pour les appels internes (LAN/WAN de la société). Les communications vers l'extérieur (réseau PSTN « classique ») sont effectuées via une carte dite *FXO*, permettant de connecter le PABX au réseau téléphonique public. Des évolutions vers du tout-IP sont considérées ; elles pourraient être implémentées en souscrivant un abonnement auprès d'un opérateur VoIP s'occupant de l'interconnexion PSTN. Une liaison (physique ou logique type IPSEC) serait alors établie vers les proxy servers SIP de cet opérateur.

[4] `http://www.asterisk.org/`

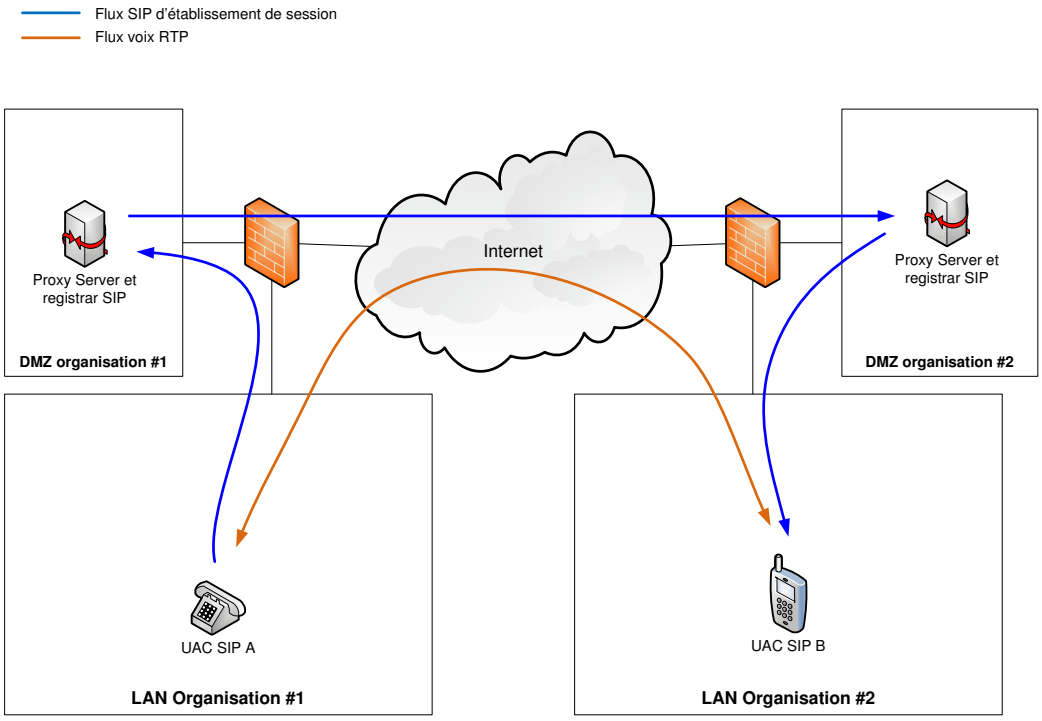

FIG. 2: Schéma réseau typique et flux SIP/RTP dans le cadre d'une conversation VoIP

5 Risques et vulnérabilités en VoIP

5.1 Les risques de sécurité en VoIP

La VoIP hérite de la plupart des risques déjà présents sur le POTS. Cependant, certains peuvent être exacerbés par l'utilisation de réseaux et protocoles informatiques classiques, sur lesquels les agresseurs sont beaucoup plus nombreux et la connaissance très accessible. En termes de risques, on considère souvent en premier lieu ceux pesant sur la disponibilité de la solution VoIP. Les fraudes téléphoniques sont ensuite adressées. On traite finalement les problèmes relatifs à la confidentialité des communications. Ces risques peuvent apparaître au travers de nombreuses vulnérabilités sur les infrastructures VoIP. Les vulnérabilités les plus courantes, ainsi que les profils de risques associés, sont détaillés dans les paragraphes qui suivent.

5.2 La disponibilité

Les risques considérés à juste titre en premier lieu dans les déploiements VoIP sont ceux affectant la disponibilité des services de téléphonie. En effet, alors que sur les POTS les utilisateurs se sont habitués à des taux de service très élevés (et des pannes somme toute très rares, sur les réseaux des opérateurs « historiques »), l'équivalent peut être très difficile à atteindre en VoIP. L'énergie

représente un premier point de panne. Les téléphones POTS peuvent directement disposer d'une source d'énergie dans la ligne téléphonique (on rappelle que les réseaux POTS délivrent dans la prise téléphonique de l'abonné, prise dite *FXS*, un courant de 48 V). Si une coupure électrique des bureaux peut faire tomber tous les équipements informatiques, les téléphones POTS resteront au moins utilisables pour les services d'urgence, l'énergie fournie par le réseau étant suffisante pour passer des appels.

Dans le cas de la VoIP, le réseau ne peut pas fournir l'énergie nécessaire au fonctionnement des équipements. En cas de panne électrique, la plupart des équipements VoIP sont donc complètement inopérants. Certains protocoles, tels que le fameux 802.3af, aussi connu sous le nom de *Power-over-Ethernet*, permettent de faire circuler de l'énergie sur deux paires filaires d'une prise Ethernet. Cependant, le PoE n'est aucunement destiné à fournir une alimentation fiable, constante vis-à-vis des problèmes énergétiques. Le rôle du PoE est principalement un rôle de confort, pour éviter la multiplication des petits blocs transformateurs dans les bureaux. En définitive, pour obtenir un même niveau de disponibilité que pour le POTS, la VoIP nécessite une gestion optimale de l'énergie au niveau de l'entreprise. Cette gestion de l'énergie est le principal challenge auquel doivent faire face les équipes déployant des infrastructures VoIP.

Là où les infrastructures POTS pouvaient se considérer au travers de quelques éléments critiques seulement, la VoIP nécessite un nombre important de « supports », que cela soit les serveurs VoIP eux-mêmes, mais aussi les équipements réseaux (coeurs de réseau, switchs d'étages...) ou les serveurs annexes (base LDAP pour l'annuaire ou l'authentification registrar...). La disponibilité de services VoIP est donc fonction de nombreux points de panne possibles, qu'il est nécessaire de connaître et adresser, avant même de songer aux éventuels problèmes de QoS.

5.3 Configurations par défaut des user-agents

Comme de nombreux services disposant d'une pile TCP/IP et exposant des services à l'extérieur, les équipements SIP, qu'ils soient simples téléphones ou serveurs d'infrastructures, sont livrés avec une configuration par défaut trop souvent laxiste d'un point de vue sécuritaire.

De nombreux fabricants fournissent notamment les services suivants, activés par défaut sur leurs téléphones : serveurs Web et telnet de configuration distante, agent SNMP, client DHCP, client et/ou serveur TFTP... Ces services sont bien évidemment protégés par des authentifiants « usines » très faibles, facilement trouvés sur Internet, dans les manuels des constructeurs ou dans les listes idoines de mots de passe par défaut.

Ces mots de passe par défaut sont rarement changés. Dans de trop nombreux déploiements, l'utilisation d'un VLAN séparé pour les équipements VoIP semble suffire pour garantir qu'ils ne feront l'objet d'aucune attaque... Ceci est d'autant plus problématique qu'outre les possibilités de déni de services résultant de la reconfiguration du téléphone, l'accès aux interfaces d'administration expose à de nombreux risques. La page de configuration du téléphone, sur le serveur Web d'administration de ce dernier, va notamment contenir les authentifiants de l'utilisateur (login / password SIP), « masquées » dans un champ password HTML mais immédiatement récupérables en clair dans le code source de la page. Chez certains fabricants, des fonctions avancées sont mêmes disponibles sur l'interface Web du téléphone. Ces fonctions permettent par exemple de numéroter et d'activer la fonction mains libres du téléphone, à distance.

Si le serveur HTTP représente une porte d'entrée plaisante, toutes les fonctions présentent dans les pages sont généralement accessibles via les autres interfaces, qu'elles soient telnet ou SNMP. Dans ce dernier cas, le peu de documentation des MIB SNMP des constructeurs n'offre qu'un bien modeste rempart...

5.4 À quand le retour des *boxes* ?

Le système téléphonique a toujours été source de curiosité pour certains individus, avides de connaissances sur le sujet et de moyens de le détourner à leur profit. Dès les années 1960, les *phreakers* utilisaient diverses astuces pour exploiter les vulnérabilités des réseaux téléphoniques. Là où l'on aurait maintenant parlé d'*exploits*, ils développaient des *boxes* pour faciliter voire automatiser ces arnaques. Ainsi, la *bluebox* permettait d'appeler gratuitement, en simulant sur la ligne la signalisation de réseau, tirant profit du fait que les réseaux téléphoniques faisaient circuler la signalisation sur la même ligne que la conversation elle-même[5].

À quand les *IPboxes* ? Alors que les phreakers et leurs boxes ont disparu avec la migration de la plupart des PSTN occidentaux vers des protocoles de signalisation hors bande type SS7, on peut imaginer l'arrivée de nouveaux phreakers, exploitant les vulnérabilités dans les infrastructures VoIP mal sécurisées.

Dans le protocole SIP, le téléphone qui souhaite émettre un appel doit préalablement s'authentifier auprès de son registrar. Cette authentification se fait dans la plupart des cas au moyen d'un simple login/password programmé dans la configuration du téléphone. L'obtention de ce login/password permettrait à un agresseur de s'enregistrer auprès du registrar sous l'identité de la victime, ouvrant la porte à de nombreuses attaques, de la fraude à la facturation jusqu'à l'*ingénierie sociale* vis-à-vis des correspondants légitimes de la victime. Même si le mot de passe ne circule pas en clair dans l'échange `REGISTER` (le protocole SIP interdit l'utilisation de l'authentification `basic`, et lui préfère le HTTP digest basé sur MD5), il est souvent aisé de récupérer ce mot de passe. En effet, comme on l'a vu précédemment, les téléphones offrent des interfaces d'administration rarement protégées ; nous avons même pu voir chez certains constructeurs le mot de passe SIP sortir au milieu d'une interrogation SNMP walk ! Dans les cas où le téléphone est bien protégé, mais où l'agresseur a accès au port SIP du proxy/registrar, une attaque par brute-force réseau peut être menée. Elle a d'autant plus de chances de succès que les mots de passe utilisés sur les téléphones répondent rarement aux attendus en matière de sécurité... On comprend aisément qu'il peut être très embêtant de taper un mot de passe complexe sur le clavier d'un téléphone ! Tout comme pour la téléphonie classique, la VoIP nécessite au niveau des équipements de sécurité - en l'occurrence les proxy servers, l'établissement de listes d'autorisation. Ces classes de services définissent les permissions de chaque utilisateur vis-à-vis de services tels que l'accès aux numéros externes ou aux services étendus de messagerie vocale, conférences... Les classes de services doivent être déployées au niveau de tous les proxy servers accessibles par les clients (légitimes ou non). Ainsi, nous avons vu des architectures dans lesquelles les permissions étaient bien validées au niveau des proxy servers configurés sur les postes, mais n'étaient aucunement validées sur les proxy servers upstream, qui se basaient uniquement sur le filtrage effectué par les premiers. Il suffisait alors aux utilisateurs de reconfigurer leur téléphone pour utiliser directement ces proxy servers upstream et bénéficier de services sans restrictions. Ces problèmes sont donc exacerbés par l'utilisation d'IP, son caractère universel et la difficulté qu'on peut parfois avoir à définir les frontières exactes des zones de sécurité[6].

[5] On remarquera à ce propos que de nombreux problèmes de sécurité sont liés à l'utilisation du même canal pour la signalisation et les données. Ainsi, la possibilité d'exploiter stack et heap overflows tient essentiellement du fait que données et informations de contrôle (headers de chunks ou adresses de retour des fonctions) sont stockées dans les mêmes zones mémoires.

[6] Le problème en question a été diagnostiqué comme mauvais filtrage : le proxy server SIP upstream était situé dans une DMZ, normalement inaccessible par les clients internes. Une erreur dans les règles de filtrage sortantes au niveau du firewall exposait les systèmes de la DMZ à l'ensemble du réseau interne.

Alors qu'auparavant les fraudeurs téléphoniques avaient peine à dissimuler leur localisation, l'automatisation des attaques et l'impunité à agir derrière des relais peut laisser envisager des fraudes massives aux numéros surtaxés, à l'usurpation d'identités.

5.5 Absence de confidentialité

Les infrastructures VoIP basées sur SIP/RTP ne proposent par défaut aucune confidentialité sur les flux de données voix/vidéo. Ces flux peuvent être interceptés et décodés par toute personne capable de sniffer en un point du chemin pris par les paquets RTP. Des logiciels existent dans le domaine public pour ce faire, tels les fameux *VoMIT* ou *VoIPong*, ce dernier ayant la capacité de détecter tout trafic RTP/RTCP et d'enregistrer la voix sur disque lorsqu'elle est encodée selon le standard ITU-T G.711.

L'absence par défaut de chiffrement est tout à fait justifiée. En effet, le chiffrement peut introduire un *overhead* au niveau des user agents. D'autre part, le chiffrement n'est nécessaire que dans certaines circonstances, pour certains utilisateurs : l'utilisation systématique du chiffrement serait inutile et rendrait plus difficile le diagnostic d'éventuels problèmes. Finalement, le chiffrement n'est pas forcément compatible avec l'ensemble des législations, certaines d'entre elles imposant même un enregistrement des conversations (cas par exemple des réglementations du gendarme américain de la bourse, la SEC, pour les appels passés depuis/vers des salles de trading). Le problème est en fait tout autre. En effet, les problèmes surviennent lorsqu'il est nécessaire de chiffrer : certaines conversations ont effectivement vocation à rester confidentielles, d'où la nécessité du chiffrement. Les utilisateurs et développeurs de solutions se retrouvent alors confrontés à la variété des solutions possibles, et évidemment à leur incompatibilité chronique. Le standard SRTP / SRTCP (Secure RTP) définit une extension des protocoles RTP/RTCP pour le chiffrement des données temps réel. Ce protocole repose sur un chiffrement unique, l'AES, utilisé dans un mode le transformant en stream cipher. Cependant, SRTP part du principe qu'un secret est déjà partagé entre les deux entités qui communiquent. Il convient donc, préalablement à l'établissement d'une conversation sécurisée, de négocier un secret partagé entre les deux entités. C'est sur ce point que de nombreuses variantes ont vu le jour, certaines ayant plus ou moins de succès, mais avec au final une conclusion dominante : il n'existe pas pour l'instant de réel standard pour la négociation des clefs secrètes. Certains vendeurs font transiter le secret AES encodé en base64 dans la signalisation SIP. Cette méthode présente deux problèmes : tout d'abord, elle part du principe que le flux de signalisation est sécurisé, et oblige donc à utiliser un protocole tel que le SIP over TLS. D'autre part, elle n'aboutit pas vraiment à un chiffrement complet de bout en bout de la conversation. En effet, le téléphone appelant va contacter en TLS le proxy auquel il est rattaché, qui aura accès à cette fameuse clef de session AES, et devra la retransmettre au proxy suivant de manière sécurisée : les intermédiaires connaissent donc la clef de session.

Pour pallier ces problèmes, plusieurs protocoles de négociation des clefs de session SRTP ont vu le jour. Les plus courants sont actuellement MIKEY, SDES et ZRTP, chacun utilisant une approche différente, avec ses avantages et inconvénients. Plusieurs questions importantes doivent être adressées lors de la conception de protocoles de négociation de clefs dans la VoIP. La négociation de sécurité (c'est-à-dire le choix des algorithmes, l'établissement sécurisé des quantités secrètes et si besoin l'authentification des pairs) ne doit pas avoir d'effet perceptible notable sur l'appel. Le surplus de messages ou d'octets qu'il induit doit pour ceci être limité au maximum. Le protocole doit pouvoir être traversé sans difficulté par les proxy servers SIP ; il doit supporter un certain nombre des fonctionnalités avancées souhaitées par les utilisateurs, notamment les conférences multipartites,

ainsi que les redirections, messages d'attente etc. Les approches sont tout aussi nombreuses. Alors que certains protocoles (SDES, MIKEY) font passer leurs échanges dans le flux de signalisation SIP, ZRTP le fait uniquement dans le flux RTP, en utilisant les options de ce protocole. Autant de possibilités offertes, pour l'instant loin d'être adoptées par l'industrie et réalistes en termes de déploiement.

5.6 Simple is beautiful

Les développements que nous avons menés autour de SIP et ses protocoles, dans le cadre de la conception de produits de sécurité VoIP, nous ont amené au constat que SIP est d'une complexité allant à l'encontre de la simplicité recherchée en sécurité. En effet, SIP a été conçu comme un protocole d'établissement complètement générique, utilisable non seulement pour la VoIP mais pour toute autre application faisant appel au concept de sessions de communication pair à pair.

Si cette vue générique a du sens en théorie, elle oublie malheureusement que ses seules applications pratiques, qui fonctionnent actuellement sur Internet, sont la VoIP et ses dérivés. L'avalanche de protocoles qui s'en suit (il faut bien à un moment définir comment SIP sera utilisé réellement pour la VoIP, ce qui est fait dans une multitude d'autres drafts !) apporte une importante complexité, qui s'ajoute aux difficultés rencontrées par les concepteurs d'équipements VoIP. La majeure partie du temps dans les projets de développements VoIP est donc passée d'une part sur la compréhension des protocoles, sur l'analyse inversée de ce qui existe déjà, et d'autre part sur les tests d'interopérabilités, quelque peu aléatoires quand on sait comment chaque vendeur a implémenté ce qu'il avait compris des protocoles. Rappelons par exemple qu'outre les 270 pages de la RFC 3261 qui définit SIP, qui vient d'ailleurs en clarification de la RFC 2543 d'origine, environ 150 drafts viennent apporter à SIP telles ou telles fonctionnalités nécessaires aux applications de la « vraie vie » !

La structure même du protocole SIP, basée exclusivement sur des échanges en texte UTF-8, est un autre problème auquel doivent faire face les développeurs de stacks SIP. Un protocole texte est effectivement appréciable pour l'utilisateur, l'administrateur ou l'étudiant, qui peuvent facilement voir à l'analyseur réseau les trames protocolaires circulant sur le réseau. Cependant, du point de vue du programmeur, l'implémentation de parseurs textes est beaucoup plus complexe et sujette à une multitude de bugs, ayant pour conséquence le déni de service jusqu'à la faille de sécurité exploitable (overflow dans le parseur, par exemple). La capture suivante illustre le type de messages que doit traiter une implémentation SIP. Tiré d'un draft IETF, elle donne trois messages SIP classiques, `INVITE` qui définit l'appel d'un poste `UserA@atlanta.com` vers `UserB@biloxi.com`, un message de retour `180 Ringing` puis un dernier message `200 OK` indiquant que le correspondant a décroché. On constate la structure similaire à HTTP des en-têtes, ainsi que le payload en format SDP. Notons que contrairement à certains protocoles où les en-têtes sont utilisées uniquement pour du diagnostic (SMTP), tous les champs de ce paquet (en-têtes SIP puis contenu SDP) sont utiles et ont vocation à être parsés par la stack SIP/SDP distante.

Courant 2005, le projet de PROTOS s'est penché sur la résistance de quelques implémentations commerciales de SIP. Monté par des chercheurs de l'université finlandaise d'Oulu sponsorisés par quelques industriels, ce projet a pour objectif de systématiser les tests « boite noire » contre les implémentations de protocoles courants. Même si leurs tests SIP portaient uniquement sur la gestion des messages `INVITE` par neuf implémentations, leur conclusion était sans appel, illustrant la réalité du risque sur les implémentations SIP : « *Although the test-material was designed as simple exercise of headers and fields in isolation, the failure rate was alarming. Only one from the sample of nine implementations survived the test-material as it is. This calls for a more comprehensive test-suite to be developed as the SIP scene matures.* »

```
-- Message d'appel (invite) de UserA@atlanta.com vers UserB@biloxi.com

  INVITE sip:UserB@biloxi.com SIP/2.0
  Via: SIP/2.0/TCP client.atlanta.com:5060;branch=z9hG4bK74bf9
  Max-Forwards: 70
  From: BigGuy <sip:UserA@atlanta.com>;tag=9fxced76sl
  To: LittleGuy <sip:UserB@biloxi.com>
  Call-ID: 3848276298220188511@atlanta.com
  CSeq: 1 INVITE
  Contact: <sip:UserA@client.atlanta.com;transport=tcp>
  Content-Type: application/sdp
  Content-Length: 143

  v=0
  o=UserA 2890844526 2890844526 IN IP4 client.atlanta.com
  s=-
  c=IN IP4 192.0.2.101
  t=0 0
  m=audio 49172 RTP/AVP 0
  a=rtpmap:0 PCMU/8000


-- Message indiquant que l'appel a été reçu et que le téléphone sonne

  SIP/2.0 180 Ringing
  Via: SIP/2.0/TCP client.atlanta.com:5060;branch=z9hG4bK74bf9
   ;received=192.0.2.101
  From: BigGuy <sip:UserA@atlanta.com>;tag=9fxced76sl
  To: LittleGuy <sip:UserB@biloxi.com>;tag=8321234356
  Call-ID: 3848276298220188511@atlanta.com
  CSeq: 1 INVITE
  Contact: <sip:UserB@client.biloxi.com;transport=tcp>
  Content-Length: 0


-- Message indiquant que le correspondant a décroché, début de l'échange de données

  SIP/2.0 200 OK
  Via: SIP/2.0/TCP client.atlanta.com:5060;branch=z9hG4bK74bf9
   ;received=192.0.2.101
  From: BigGuy <sip:UserA@atlanta.com>;tag=9fxced76sl
  To: LittleGuy <sip:UserB@biloxi.com>;tag=8321234356
  Call-ID: 3848276298220188511@atlanta.com
  CSeq: 1 INVITE
  Contact: <sip:UserB@client.biloxi.com;transport=tcp>
  Content-Type: application/sdp
  Content-Length: 141

  v=0
  o=UserB 2890844527 2890844527 IN IP4 client.biloxi.com
  s=-
  c=IN IP4 192.0.2.201
  t=0 0
  m=audio 3456 RTP/AVP 0
```

FIG. 3: Capture de quelques messages SIP échangés lors d'une négociation de session (source : RFC).

Comme le résume parfaitement Arjun Roychowdhury, spécialiste des infrastructures VoIP et Broadband dans un post[7] sur son blog fin 2005, « *As far as I am concerned, today, SIP stands in the market as a 'HTTP similar expandable protocol' but to the developer, stands as a "Massive hack of spaghetti headers and rules"* ».

[7] `http://corporaterat.blogspot.com/2005/12/developer-perspective-painful.html`

Pour appuyer ce constat, présentons un extrait de la RFC 3261 définissant SIP. Cette capture donne trois constructions strictement équivalentes pour un message SIP. Le parseur devra être capable d'analyser ce message sans ambiguïté, quelle que soit la construction faite par le user-agent distant. On remarquera notamment la possibilité d'utiliser des lignes multiples pour les données d'un même en-tête, la possibilité de croiser les en-têtes entre eux etc. Combien de `strcpy`, `strcat`, `sscanf` dans les parseurs... ?
Exemple tiré de la RFC3261 montrant une série d'en-têtes SIP parfaitement équivalents

```
Route: <sip:alice@atlanta.com>
Subject: Lunch
Route: <sip:bob@biloxi.com>
Route: <sip:carol@chicago.com>

Route: <sip:alice@atlanta.com>, <sip:bob@biloxi.com>
Route: <sip:carol@chicago.com>
Subject: Lunch

Subject: Lunch
Route: <sip:alice@atlanta.com>, <sip:bob@biloxi.com>,
 <sip:carol@chicago.com>
```

Le problème de la robustesse des implémentations VoIP n'a pas seulement des conséquences pour la sécurité des architectures VoIP. Alors que les best practices recommandent la ségrégation des mondes VoIP et bureautiques classiques, l'étanchéité entre les deux est en pratique très relative. En effet, il est tout à fait illusoire de penser qu'il est possible de cloisonner VoIP et réseaux bureautiques lorsqu'on sait qu'un des avantages de la VoIP est justement la réutilisation des infrastructures et des liaisons, la réutilisation de services, d'annuaires, de bases d'authentification existantes. Ainsi, lorsque de nombreuses sociétés se seront dotées d'une présence SIP sur Internet, en exposant un proxy server sur une de leurs DMZ, il serait bien étonnant d'avoir une liaison Internet dédiée pour ce proxy server, d'avoir des bases d'authentification et d'information autres que celles de l'Active Directory, de ne pas réutiliser ces mêmes informations pour la messagerie instantanée présente sur les stations de travail.

6 Notre approche pour la sécurisation des conversations

6.1 Besoin et marché

Si l'inventaire des failles a effectivement de quoi faire peur, il est bon de se rappeler que la plupart des attaques étaient déjà possibles en POTS. A prioiri, les seuls changements concernent le nombre d'agresseurs, leur niveau global de connaissance et leur outillage. D'autre part, l'industrie de la sécurité a toujours réussi à apporter des solutions à la mesure des enjeux. Ainsi, le raffinement des attaques dont on peut avoir un aperçu par la connaissance publique n'a d'égal que l'étendue des possibilités actuelles pour sécuriser les réseaux.

Le constat que nous avons fait en tant que vendeur était justement que la VoIP, et donc les échanges téléphoniques de nos clients, allaient pouvoir bénéficier des avancées en matière de sécurité des systèmes d'information. Alors que le monde de la téléphonie restait jusqu'alors quelque peu

hermétique pour une personne venant de la SSI, la VoIP permettrait l'application des grands principes de sécurité, l'arrivée d'une offre produits dédiée.

Forts d'une expérience précédente dans la réalisation de logiciels de chiffrement pour messageries instantanées, nous avons décidé d'adresser le problème du chiffrement de bout en bout des conversations. Les parties précédentes de ce document ont en effet montré l'existence du problème de la confidentialité. Alors que des solutions protocolaires existent, aucune solution viable, facile à mettre en place au niveau de l'utilisateur, ne semble actuellement proposée dans les solutions des grands vendeurs VoIP[8].

Le besoin était donc bien réel. Restait à confirmer l'existence d'un marché. Validé par un certain nombre de clients potentiels, notre cahier des charges a pris pour objectif la conception et la réalisation d'un module de chiffrement bout-en-bout des conversations VoIP. Destiné à des utilisateurs privilégiés en entreprise (VIP tels que directions générales, directions financières ou autres utilisateurs clefs de grands groupes) ce module de chiffrement assurerait la confidentialité et l'intégrité des conversations, de téléphone à téléphone. Etant donné le type d'utilisateurs ciblés, il a été décidé que le module nécessiterait un minimum d'interactions et de configuration. Dans l'idéal, ce module pourrait être utilisable immédiatement après réception et formation minimale, sans aucune configuration, que ce soit de la part de l'utilisateur ou des personnels opérant les infrastructures VoIP. Une direction générale négociant des opérations stratégiques pourrait donc faire l'acquisition de notre produit et protéger ses conversations VoIP, sans même avoir à faire intervenir l'informatique interne.

6.2 Un boitier matériel pour le chiffrement des conversations

En raison du caractère confidentiel du marché des softphones (téléphones IP logiciels, embarqués sur le PC de l'utilisateur) par rapport aux téléphones matériels, nous avons souhaité que notre module de chiffrement puisse sécuriser les postes téléphoniques IP les plus courants. Aucun standard ne semblait cependant exister pour développer des extensions pour les téléphones. Nous avons donc opté pour une approche calquée sur celle employée par les chiffreurs de liens militaires : notre module de chiffrement se présenterait sous la forme d'un petit boitier matériel d'une dizaine de centimètres, que l'utilisateur poserait sur son bureau, et qui interviendrait en coupure du flux Ethernet entre téléphone et prise RJ45 murale. Ce petit boitier plastique comporterait une prise d'alimentation, deux prises réseaux, un afficheur type LCD et éventuellement quelques boutons (nombre limité au strict minimum) pour l'interaction avec l'utilisateur.

6.3 Le support des protocoles

Concernant les protocoles supportés, nos recherches ne nous ont pas permis de déterminer de véritables tendances sur les parcs actuellement déployés et à venir. Il est cependant apparu que si SIP a vocation à gagner des parts de marché ces prochaines années, les protocoles propriétaires tels SCCP (Cisco) sont aussi à considérer pour que notre boitier soit supporté par la majorité des déploiements VoIP d'entreprise. Dans un premier temps, SIP et SCCP ont donc été choisi, H.323 étant temporairement abandonné car complexe à implémenter, en plus d'être présent en majorité dans le segment des telcos, qui n'était pas visé par notre cahier des charges.

[8] Cisco et quelques autres pourraient légitimement discuter cette affirmation. Notre objectif était cependant de rendre le chiffrement de bout en bout possible pour un utilisateur, sans que des changements de configuration dussent être appliqués par les services informatique ou téléphonie.

Le chiffrement de bout en bout ne peut se faire qu'en présence d'entités capables de comprendre le chiffrement, de chaque côté de la conversation. Dans le cas d'un parc homogène, on pourrait imaginer que le client se dote exclusivement de nos boitiers pour assurer la sécurité des communications entre ses utilisateurs VIP. Un protocole de chiffrement propriétaire du flux aurait donc été envisageable, tant qu'il n'altère pas les propriétés RTP et QoS nécessaires aux infrastructures. Conscients que la compatibilité de notre produit avec des téléphones évolués, qui intègreraient déjà un support du chiffrement, serait un plus, nous avons décidé d'opter pour un protocole de chiffrement standard. Le monopole de fait donné à SRTP en la matière a facilité le choix de ce protocole ; restait ensuite à déterminer le mode de négociation des associations de sécurité (ie. des clefs secrètes). Notre choix s'est porté dans un premier temps sur MIKEY. Après quelques surprises lors des tests de compatibilité sur ce protocole, nous avons finalement opté pour ZRTP, draft proposé par Philip Zimmermann, le créateur de PGP. ZRTP a, entre autres avantages, la particularité de ne pas dépendre du protocole de signalisation (toute la négociation est faite dans le flux RTP), ni de l'existence d'une PKI.

6.4 Le développement

Notre métier est la réalisation de logiciels informatiques. Nous n'étions pas en mesure de concevoir l'architecture matérielle complète pour ce produit. Nous souhaitions nous concentrer sur notre savoir-faire, à savoir l'implémentation informatique du chiffrement. Nous avons donc décidé de faire tourner notre produit sur une architecture embarquée, qui nous permettrait de nous retrouver très rapidement dans la situation connue d'un développement logiciel. Une étude de marché nous a permis d'identifier un certain nombre de vendeurs proposant des mini cartes-mères, la plupart RISC, suffisamment dimensionnées pour recevoir notre application. Nos contraintes de départ étaient une plateforme matérielle de petite taille (pas plus d'une douzaine de centimètres de longueur), dotée d'un processeur capable de supporter des opérations assez lourdes (notamment pour la génération des paramètres de chiffrement asymétrique), de suffisamment de RAM et de mémoire non volatile, le tout embarquant un système type Linux ou BSD. Deux cartes réseau Ethernet 100Base-T étaient aussi nécessaires, l'une pour la connexion du boitier vers le téléphone, l'autre pour relier le boitier à la prise RJ45 murale. Notre choix s'est porté sur les produits proposés par la société Gumstix. Pour un coût relativement modeste (de l'ordre de 200 US$), ce constructeur propose des plateformes de taille 12 x 2 cm, architecturées autour d'un processeur Intel XScale (type ARM) à 400 MHz, 64 Mo de RAM, 16 Mo de Strataflash et deux interfaces Ethernet contrôlées par des chipsets SMSC. Passant avec succès tous nos tests de fiabilité logicielle et matérielle, ces cartes présentaient la particularité d'être entièrement compatibles Linux 2.6 et de bénéficier du processeur Intel XScale, comportant notamment un contrôleur LCD embarqué et de suffisamment de ports d'entrée/sortie versatiles (GPIO) pour nous permettre des extensions telles que des boutons pour l'interaction utilisateur. Tous nos développements, de l'écriture du code jusqu'aux tests, ont été réalisés sur des PC sous GNU/Linux équipés du toolchain binutils / gcc / g++. La transition vers la plateforme embarquée n'entrainait en fait quasiment aucun changement notable, si ce n'est l'utilisation d'un cross-compilateur sur le PC pour produire des binaires ARM, envoyés ensuite en SSH ou via liaison série sur la plateforme.

D'un point de vue architecture logicielle, les boitiers sont équipés d'un kernel Linux 2.6 standard, compilé avec une série minimum d'options telles que le support bridge et netfilter. Notre contrainte initiale d'interférer le moins possible avec l'existant chez le client nous imposait d'opter pour une approche non intrusive. Nous souhaitions négocier et activer le chiffrement des conversations de

manière complètement transparente, le boitier se chargeant de détecter automatiquement un appel entrant ou sortant du téléphone, puis d'activer et réaliser le chiffrement dans le cas où un boitier était aussi présent à l'autre bout de la conversation. L'approche pour ce faire a donc été de configurer le Linux embarqué en mode Bridge, sans même assigner d'adresse IP aux interfaces, et d'utiliser les services du kernel pour intercepter tout trafic VoIP rentrant ou sortant du téléphone. Afin de garantir la stabilité de la plateforme, et éviter qu'un crash logiciel n'entraine une coupure complète du téléphone, nous avons opté pour un positionnement de notre code en userland. Aucun module additionnel, aucun patch propriétaire n'a été appliqué au kernel. Lancé par les scripts d'initialisation, immédiatement après le démarrage du système, notre logiciel embarqué sur le boitier positionne des hooks au moyen de l'API libnetfilter_queue et récupère les paquets interceptés par le kernel et répondant à un ensemble de règles de filtrage spécifiques au trafic VoIP. Le logiciel analyse ces paquets VoIP, les chiffre si besoin, et les réinjecte vers le kernel pour réémission sur l'autre interface du bridge constitué par le boitier. En cas de crash du logiciel, l'intégrité du système (kernel) est préservée et le trafic peut continuer de traverser le bridge, certes sans chiffrement.

Afin d'optimiser le traitement et éviter que tout paquet émis depuis ou vers le téléphone soit intercepté et analysé (on rappelle que les téléphones incluent souvent un mini-switch sur lequel un poste PC peut être branché), nous décodons le protocole de signalisation passant sur son port standard, par exemple 5060 pour SIP. Dans le cas de paquets SIP caractéristiques d'un appel, nous extrayons les données SDP et en déduisons les numéros de ports qui seront utilisés pour le trafic RTP/RTCP. Le logiciel sait alors quels paquets il doit intercepter pour avoir accès aux données de la conversation. Ces paquets RTP sont attrapés, chiffrés après négociation MIKEY ou ZRTP, puis réinjectés. Un message affiché sur un petit écran LCD présent sur le boitier indique à l'utilisateur que la conversation est chiffrée. Les protocoles de négociation des associations de sécurité, notamment MIKEY, ont apporté de nombreuses surprises, certaines particulièrement désagréables. En effet, MIKEY (et d'autres) transportent les informations relatives à la négociation dans le flux de signalisation SIP. Or, il n'existe presque jamais de flux SIP direct entre les deux entités qui souhaitent communiquer. Elles passent au travers d'au moins un proxy server SIP, qui agit comme proxy applicatif vis-à-vis de l'échange SIP. Nos tests ont montré que de nombreux proxy servers SIP du marché, et notamment certains IPBX largement répandus sur le marché, réécrivent complètement les messages SIP qu'ils retransmettent, en omettant évidemment les extensions MIKEY. Ainsi, dès lors qu'un de ces proxy servers était utilisé, il devenait impossible de faire passer la négociation de clef. D'autres problèmes ont été trouvés sur MIKEY. Si l'on souhaite mieux que sa version à secret partagé (PSK), qui nécessite la configuration manuelle d'une clef secrète de part et d'autre de la conversation, il faut avoir recours à des certificats devant être signés par une PKI. Cette approche était peu pratique, eu égard à deux problèmes : premièrement, nous avons considéré que la nécessité d'une PKI était contraire à notre ambition d'un produit simple, autonome ; ensuite, le fait de placer ces certificats dans les échanges SIP augmentait la taille des paquets à un point tel qu'ils dépassaient immédiatement le MTU, obligeant à basculer dynamiquement d'un transport UDP vers un transport TCP.

Le protocole ZRTP, développé par Zimmermann, a donc été notre favori. Extension du RTP faisant l'objet d'un draft IETF, le ZRTP place toute la négociation de la clef SRTP dans le flux de données temps réel RTP lui-même. Il ne nécessite aucune PKI, et supporte tout à fait l'absence de chiffrement si une des parties n'est pas compatible. Il est en théorie indépendant des protocoles de signalisation et des proxy servers. Il propose des services de confidentialité, d'intégrité, d'anti-replay sur le flux. Il apporte une protection contre le Man-in-the-Middle en affichant de chaque côté de la conversation un hash des quantités Diffie-Hellman (ces hashes sont à lire à haute voix par les

utilisateurs pour voir s'il y a correspondance et donc continuité cryptographique). En pratique, nous avons constaté que malgré ses nombreux avantages, le ZRTP n'était pas dénué d'incompatibilités. Ainsi, certains proxy servers font non seulement du proxying pour le flux de signalisation, mais aussi pour le flux de données. C'est le cas notamment des passerelles entre plusieurs réseaux de VoIP n'utilisant pas les mêmes codecs vocaux. Dans ces circonstances, les chances de pouvoir faire du chiffrement de bout en bout nous semblent assez minces, que ZRTP soit utilisé ou non.

7 Conclusion

Nonobstant la complexité des protocoles et l'absence de standard de sécurité clairement établi, nous pensons avoir apporté aux utilisateurs de la VoIP une solution efficace pour la confidentialité de bout en bout de leurs conversations. Construite autour d'une plateforme matérielle embarquée, cette solution réussit le pari de la simplicité : fonctionnelle sans configuration, dès les branchements effectués, elle procure une vue immédiate de l'état (sécurisé ou non) de la conversation. Si ce document a principalement porté sur les vulnérabilités des systèmes de VoIP, n'oublions pas que cette technologie apporte une valeur ajoutée réelle, qui se matérialise par des économies pouvant être substantielles, mais aussi par des services nouveaux, dont les utilisateurs pourront bientôt difficilement se passer.