C&ESAR 2011

Computer & **E**lectronics
Security **A**pplications
Rendez-vous

28 - 29 - 30 novembre 2011
Rennes - France

http://www.cesar-conference.fr/

C&ESAR 2011 : mobilité et sécurité

Cette 18ième édition des journées SSI de la Défense, aussi appelées C&ESAR (Computer & Electronics Security Applications Rendez-vous), va revenir sur un thème déjà abordé en 2002, celui de la mobilité, pour en décliner certains aspects à la lumière des évolutions récentes. En effet, la sécurité des équipements mobiles ou de la téléphonie avait été traitée, mais les technologies progressant, les usages ont évolué, conduisant à une multiplication des objets communicants et de leurs fonctionnalités. La frontière devient ténue entre ce que nous appelions ordinateur portable, assistant personnel ou téléphone mobile à l'époque. Les fonctionnalités se retrouvent maintenant toutes regroupées au sein d'un même équipement, qu'il soit smartphone (dont la traduction officielle – ordiphone – suffit à illustrer cette disparition des frontières) ou tablette (qui n'a guère que sa taille pour la différencier du smartphone).

L'apparition de ces nouvelles machines s'est accompagnée de celle de nouveaux systèmes d'exploitation, plus flexibles et plus évolutifs, et de milliers d'applications destinées à tirer le maximum de leurs fonctionnalités, souvent au détriment de la sécurité. Tous ces objets nomades peuvent être employés de manière autonome ou connectés, protégés ou non. Leur versatilité rend de plus en plus perméable la frontière entre les sphères professionnelle et personnelle.

Du point de vue de la sécurité, nous assistons donc, sans véritable surprise, au franchissement de nouvelles étapes dans l'évolution du paradigme de la forteresse vers le partage universel de l'information... Il est fréquent d'apprendre que tel système ou telle application transmet des données personnelles de tout type. Avec ces nouveaux usages apparaissent de nouveaux défis, qui mettent une fois de plus en exergue l'opposition entre fonctionnalité et sécurité !

Neuf ans après, un nouveau point s'imposait donc sur la question.

Suivant le plan habituel, la première journée nous donnera un aperçu de la menace technique (smartphones, radio), la seconde journée fournira quelques éléments de solution, eux aussi techniques (sécurité des systèmes, des applications, et des réseaux), tandis que la troisième journée apportera des compléments sur les usages, les risques et les perspectives de ces systèmes mobiles.

Nous remercions les membres du comité de programme, les auteurs des diverses communications, et tous les acteurs qui contribuent au succès de cette manifestation. Sans oublier nos sponsors (DGA, DGSIC, DIRISI, Orange, Technicolor) qui rendent possible ce rendez-vous annuel de la communauté SSI.

Bonne conférence C&ESAR 2011 à toutes et à tous !

Yves Correc (DGA-MI), Président du comité d'organisation.
José Araujo (ANSSI), Président du comité de programme.

Comité d'organisation

Florent Chabaud	DGSIC
Pascal Chour	SGDSN/ANSSI
Yves Correc	DGA-MI, président du comité d'organisation
Olivier Heen	Technicolor, directeur de la publication
Ludovic Mé	Supélec
Eric Wiatrowski	Orange Business Services

Comité de programme

José Araujo	SGDSN/ANSSI,président
Christophe Bidan	Supelec
Laurent Butti	Orange Labs
Claude Castelluccia	INRIA
Yves Correc	DGA-MI
Gilles Courcoux	Alcatel-Lucent
Eric Freyssinet LCL	Gendarmerie
Olivier Heen	Technicolor
Maryline Laurent	Telecom Sud-Paris
Thierry Lestable	Sagemcom SAS
Alexandre Mallard	Ecole des mines de Paris
Refik Molva	Eurecom
Nicolas Ruff	EADS
Jean-Pierre Tual	Gemalto

Site officiel : `http://www.cesar-conference.fr/`

Table des matières

I 28 novembre 2011

De l'importance de prendre en compte les attaques par injection de fautes sur plateformes mobiles 1
Joan Mazenc, Julien Badoules, Sébastien Valette, Jean-Christophe Courrège

Fuzzing on the HTTP protocol implementation in mobile embedded web server 14
Matthieu Barreaud, Guillaume Bouffard, Nassima Kamel, and Jean-Louis Lanet

Smartphones : La sécurité de votre poche... 28
Ivan Fontarensky

État de l'art de la prise d'empreinte 802.11 40
Olivier Heen, Christoph Neumann, Stéphane Onno

De la radio matérielle à la radio logicielle : impact sur l'étude de la sécurité des réseaux sans fil 45
Chaouki Kasmi, Arnaud Ebalard et Pierre-Michel Ricordel

II 29 novembre 2011

Mise en œuvre de politiques de protection des flux d'information dans l'environnement Android 65
Valérie Viet Triem Tong, Radoniaina Andriatsimandefitra, Stéphane Geller, Simon Boche, Frédéric Tronel, Christophe Hauser

Etablissement de Clé de Session en Environnement M2M entre Nœuds à Ressources Fortement Hétérogènes 81
Yosra Ben Saied, Alexis Olivereau

État des lieux de la sécurité des communications cellulaires 102
Chaouki Kasmi et Benjamin Morin

XSS Test Driver et les navigateurs web sur mobile 121
Erwan Abgrall, Yves Le Traon, Sylvain Gombault, and Alain Ribault

TazSecure, système d'authentification forte basé sur la biométrie et un composant sécurisé 143
Ismail SABRY

III 30 novembre 2011

Comment allier objectifs d'usage et objectifs de sécurité ? 153
Pierre-Yves Gouardin, Sahra Zaim, Charles Capron, Claire Premont

Vers une politique de mobilité 167
Pierre-Yves Bouf

Security in navy systems, Industrial point of view 170
Patrick Hébrard, Laurent Comte

Communications Opportunistes : Défis de Sécurité 172
Abdullatif Shikfa, Melek Önen et Refik Molva

Première partie

28 novembre 2011

De l'importance de prendre en compte les attaques par injection de fautes sur plateformes mobiles

Joan Mazenc, Julien Badoules, Sébastien Valette, Jean-Christophe Courrège

CESTI Thales (CEACI)
{joan.mazenc,julien.badoules,sebastien.valette,
jean-christophe.courrege}(@)thalesgroup.com

Résumé Nous présentons dans ce document les premiers cas d'injections et d'exploitations de fautes matérielles sur environnement mobile. Nous avons choisi de cibler la plateforme Android, dont la popularité ne cesse de croître. La bibliothèque de calcul RSA, située au cœur de toutes les transactions sécurisées, apparait vulnérable à ce type d'attaque. D'autre part, les applications utilisateur (non natives) peuvent elles aussi être vulnérables. Deux preuves de concept sont détaillées. Ces résultats démontrent la nécessité de prendre en compte ces menaces dans le cycle de développement logiciel sur plateforme mobile.

Mots-clés: Android, attaques matérielles, RSA, DFA, tablette tactile.

1 Introduction

Depuis la démocratisation du téléphone portable, ce dernier a subi de nombreuses modifications alliant à sa fonction de base, des fonctionnalités complémentaires aussi bien personnelles que professionnelles. De nos jours, ces téléphones intelligents, rejoints par les tablettes numériques, nous permettent d'effectuer certaines tâches quotidiennes comme la lecture de mails, le stockage de données confidentielles, des paiements en ligne ou via une application... Le développement de ces applications fait des ces ordiphones une cible privilégiée. Les attaques publiées à ce jour sont basées sur la recherche et l'exploitation de vulnérabilités logicielles que ce soit à distance ou en intervenant directement sur le produit[10]. Les attaques physiques ont démontré leur efficacité que ce soit dans le domaine de la carte à puce ou pour contourner des fonctions de sécurité sur des équipements plus complexes comme les consoles de jeux [11]. Cette famille d'attaques peut s'avérer tout aussi pertinente pour casser des fonctions de sécurité offertes par un ordiphone.

Aujourd'hui, il existe plusieurs systèmes d'exploitations qui équipent les ordiphones et les tablettes tel iOS d'Apple, Android de Google, Palm OS... Depuis son lancement en 2008 par Google, Android est devenu le système d'exploitation mobile pour ordiphone le plus représenté, dépassant même le populaire iOS d'Apple. La plateforme Android s'appuie sur un noyau linux et dispose d'une machine virtuelle java (*Dalvik*) pour l'exécution des applications.

Android est proposé sous forme de projet open source, les développeurs peuvent facilement et surtout gratuitement créer leur application en JAVA au travers du SDK Android, mais aussi implémenter des briques logicielles natives via le *Native Development Kit* (NDK) pour obtenir de meilleures performances. Tout ces avantages nous ont orienté vers cette plateforme pour pouvoir réaliser nos différents tests d'attaques.

La richesse de la bibliothèque d'applications *Android Market* incite les fabricants d'électronique grand public à utiliser, supporter et contribuer à l'expansion de l'écosystème Android. Ainsi, RIM a choisi de rendre possible l'exécution d'applications provenant de l'*Android Market* sur sa tablette *Blackberry Playbook*, Intel est en cours de développement d'une version x86 d'Android 3.0 et Google entend faire d'Android un système d'exploitation pour la domotique.

L'omniprésence programmée d'équipements sous Android n'a pas échappé à l'analyse des chercheurs en sécurité et la disponibilité d'une grande partie du code source, facilite la recherche de vulnérabilité. Depuis quelques mois, de nombreuses anomalies ont été remontées, certaines affectent directement le cœur du système, comme la collecte des coordonnées GPS [5] ou l'envoi en clair des identifiants Google Calendar [9] alors que d'autres concernent des applications commerciales (Skype) [1].

Les besoins de sécurité sont de plus en plus importants et présentent de fortes similitudes avec ceux d'une carte à puce. Les menaces sont quasiment identiques de par la nature portable et accessible de l'équipement. Les biens à protéger ne sont pas de la même nature mais peuvent être vus, dans les deux cas, comme des informations à protéger en intégrité et en confidentialité.

L'analyse sécuritaire d'Android ne se résume pour l'instant qu'à l'indentification d'attaques logicielles qu'elles soient distantes ou locales. Lors de l'évaluation sécuritaire d'une carte à puce, toutes les menaces sont prises en compte et la résistance aux attaques physiques est primordiale. Notre démarche a consisté à évaluer la résistance d'Android vis-à-vis d'une classe particulière de ce type d'attaques : l'injection de fautes. En effet, nous pensons que ce type d'attaques n'est pas

pris en compte dans le modèle de sécurité Android, ni audité par les chercheurs. Ces ordiphones ou tablettes ne sont pas conçus pour résister à ce type d'attaques.

Une injection de faute consiste à perturber physiquement un équipement lorsqu'il est en train d'exécuter une opération critique, sécuritairement parlant. La faute peut viser une copie en mémoire, un calcul cryptographique, la vérification d'un code secret, etc.

Ce document présentera dans un premier temps notre méthode d'injection de fautes et les effets attendus. Dans une seconde partie, l'accent sera mis sur les méthodes de recherche de vulnérabilité vis-à-vis des attaques par faute. Enfin, deux preuves de concept seront présentées sur un produit commercial grand public suivies par un scénario d'attaque les utilisant. Il s'agit à notre connaissance du premier exemple d'attaques par fautes sur système d'exploitation mobile.

2 Injection de fautes

L'injection de fautes est un procédé semi invasif, il requiert un accès physique au système. Plusieurs techniques permettent de perturber un circuit. Nous avons choisi l'injection électromagnétique, qui consiste à envoyer sur le circuit en fonctionnement une impulsion électromagnétique directionnelle de faible puissance et de très faible durée.

La sonde d'injection en champs proche étant directionnelle, il nous est possible de perturber une partie bien spécifique du circuit et la nature des ondes émises permet de procéder à l'injection de fautes sans modification/préparation des composants ciblés.

L'injection d'une faute permet de provoquer une perturbation transitoire du circuit pendant quelques microsecondes. Une campagne d'injection de fautes peut être menée sur plusieurs zones. L'attaque peut par exemple viser :

- le processeur applicatif, pour perturber l'exécution du code ou les périphériques embarqués (accélérateur cryptographique, interface de communication...)
- la mémoire volatile (RAM), pour causer des erreurs lors d'écritures/lectures de tableaux
- la mémoire non volatile : pour affecter le chargement du code, la lecture de fichiers de configuration, l'écriture de journaux...

Si les fautes laser produisent un effet relativement maitrisé, l'injection électro-

magnétique n'a pas encore atteint le même niveau de maturité et il est encore difficile d'en caractériser ses effets. Pour que l'injection de faute fonctionne et étant donné que le phénomène est transitoire, il convient de déclencher l'injection électromagnétique avec une bonne précision temporelle. Pour finir la puissance de l'impulsion et sa polarité doivent aussi être paramétrées pour causer une perturbation transitoire.

En résumé, l'attaquant doit être capable de trouver la bonne combinaison de paramètres pour :

- obtenir l'effet désiré
- et surtout, ne pas causer d'effets secondaires (erreurs, destruction de l'équipement).

La localisation temporelle est la plus difficile à obtenir. Ce paramètre est obtenu par rétro conception en observant d'une part le code source à perturber et d'autre part, tous les canaux auxiliaires possibles. Dans le cas d'un équipement mobile tel qu'un ordiphone ou une tablette tactile, on peut se fier aux informations suivantes :

- retours du programme : erreurs ou non
- journaux systèmes : logcat
- consommation en courant du circuit : lignes d'alimentation du cœur
- mesure de canaux auxiliaires sur les composants (CPU,RAM, flash)
- signaux observés sur des bus ou des points de test

Outre la partie rétro conception qui permet de déterminer le moment précis de l'injection, ces informations vont aussi être utilisées pour commander l'injection.

3 Recherche de vulnérabilités

3.1 Méthode d'analyse

La recherche de vulnérabilité en vue d'une injection de faute peut se faire en boite noire, ou en boite blanche. Nous allons nous concentrer sur l'approche boite blanche. Notre méthode d'analyse dérive des critères communs, où tout commence par une identification des menaces, des biens à protéger et des fonctions de sécurité implémentées par le système. Puis vient la recherche de chemins d'attaque basée sur ces informations. L'analyse des scénarios d'attaque envisagés mène à un audit de code et/ou à une caractérisation du système qui viennent confirmer ou infirmer la présence d'une faiblesse. Si l'exploitation de cette dernière fonctionne, la vulnérabilité est avérée [8].

Contrairement à une méthode de recherche de vulnérabilités logicielles, où l'on prend comme postulat que le code est immuable et seules les données passées au programme peuvent être manipulées par l'utilisateur, l'approche injection de faute est plus libre. Sous l'influence d'une faute, on peut par exemple observer la non exécution d'une instruction assembleur, la mauvaise lecture/écriture d'une donnée...

La vulnérabilité typique d'un code vis-à-vis d'une injection de faute est le "simple test".

```
if (!memcmp(code_fourni,code,taille_code) {
       printf("gagné");
}
else {
       printf("perdu");
}
```

Quel que soit le code assembleur généré, il y aura plusieurs moyens pour atteindre la section critique au travers d'une simple faute.

- Si la longueur de comparaison est fautée à 0 : passage de paramètre à memcmp, lecture/utilisation du paramètre/registre dans memcmp.
- Si le retour de memcmp est fauté à 0.
- Si l'instruction de branchement pour sauter à "perdu" en cas d'erreur n'est pas exécutée.

En effectuant un survol du code de la plateforme, nous avons pu constater que les menaces physiques de types injection de faute ne sont, globalement, pas prises en compte.

3.2 Identification d'une vulnérabilité

Dans le but de réaliser une preuve de concept, nous avons cherché un mécanisme qui est largement utilisé, qui a de fortes probabilités de ne pas être protégé vis-à-vis des fautes et qui peut conduire à un résultat probant, même si la faute n'est pas extrêmement précise, spatialement et temporellement.

Nous avons donc opté pour l'analyse de fonction de signature RSA-CRT mise à disposition par la bibliothèque open source *Bouncy Castle*. La fonction est largement utilisée, que se soit pour assurer une garantie d'authenticité (bootloader par exemple), ou protéger en confidentialité et en authenticité des transactions et des données (SSL). Pour une signature avec un exposant secret de 1024 bits, on peut s'attendre à un temps de calcul de plusieurs millisecondes. Enfin, l'attaque d'un RSA CRT par les laboratoires Bellcore en 1997 [4] décrit

que par perturbation de l'une des deux exponentiations modulaires (à exposant secret) requises, la signature fautée obtenue peut être exploitée pour retrouver l'exposant de l'exponentiation non perturbée. La technique d'attaque mise en oeuvre est appelée DFA pour *Differential Fault Analysis* car elle exploite une différence entre la signature originale et la signature fautée.

La faute peut survenir à tout moment durant l'une des deux exponentiations modulaires. En d'autres termes, quel que soit le résultat de l'exponentiation fautée, la signature générée permet de retrouver l'autre exposant secret par simple PGCD et donc l'intégralité des paramètres secrets de signature. Le listing suivant est extrait du fichier *RSACoreEngine.java* de la bibliothèque *Bouncy Castle* :

```
180             // mP = ((input mod p) ^ dP)) mod p
181             mP = (input.remainder(p)).modPow(dP, p);
182
183             // mQ = ((input mod q) ^ dQ)) mod q
184             mQ = (input.remainder(q)).modPow(dQ, q);
185
186             // h = qInv * (mP - mQ) mod p
187             h = mP.subtract(mQ);
188             h = h.multiply(qInv);
189             h = h.mod(p);
190
191             // m = h * q + mQ
192             m = h.multiply(q);
193             m = m.add(mQ);
194             return m;
```

La bibliothèque implémente le calcul de manière naturelle. Les deux exponentiations modulaires pour les calculs de mP et mQ utilisent la fonction *modPow*. Cette dernière invoque l'exponentiation modulaire *BN_mod_exp* implémentée par *OpenSSL*. Une fois ces calculs effectués, les deux parties sont recombinées et le résultat est retourné à l'utilisateur. Un guide simple aux développeurs d'application utilisant la bibliothèque *Bouncy Castle* ou autre appelant *OpenSSL* serait de vérifier la signature générée avant de l'exploiter et de la transférer à l'utilisateur. Nous allons montrer, dans la suite de ce document, ce que donnerait une attaque en faute, si une telle contre mesure n'était pas suivie.

4 Exploitation d'une vulnérabilité

4.1 Choix de l'équipement à tester

Pour effectuer nos tests, nous avons choisi une tablette tactile Android grand public. Cette tablette est construite autour d'un processeur ARM cadencé à 1GHz, Android est en version 2.2.1. Nous l'avons choisie pour son coût raisonnable et son succès commercial (même si très loin de celui de l'iPad d'Apple).

4.2 Identification des composants de la tablette

La photo 1 représente l'identification des composants nécessaire à la mise en place d'une attaque par injection de fautes. On identifie clairement le processeur et la mémoire flash. Sur cette tablette Android, les composants sont facilement accessibles et reconnaissables.

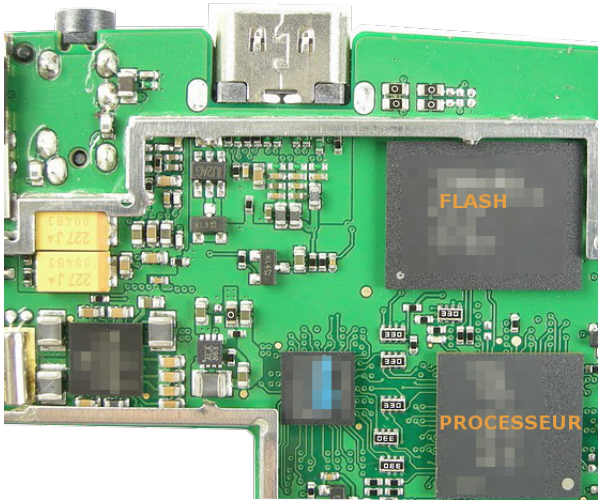


Figure 1. Identification des composants.

Une fois la tablette ouverte, nous avons analysé les canaux auxiliaires des différents composants. Après plusieurs observations sur la mémoire flash, nous avons repéré un motif assez régulier correspondant à des écritures de fichiers.

4.3 Instrumentation du code

Le but de cette attaque est de mettre en défaut la bibliothèque *Bouncy Castle*. Au lieu de chercher à exploiter la vulnérabilité identifiée en audit de code au travers d'une application commerciale, nous avons choisi d'implémenter notre

propre application de test. Cette dernière utilise le mécanisme de signature RSA. Le but étant de réaliser une preuve de concept, nous disposons de toute la latitude nécessaire pour instrumenter le code. Afin d'injecter une faute, il est nécessaire de repérer les exponentiations mises en oeuvre par la signature RSA CRT à attaquer. La grande difficulté est d'identifier les évènements pertinents. Nous avons donc encadré le calcul de la signature par deux évènements identifiables comme par exemple :

1. Lecture d'un son.
2. Mise sous tension de la diode de chargement.
3. Manipulation de la mémoire non volatile.

Cependant, d'autres évènements sont déclenchés par le système Android. Pour pallier à ce problème, une lecture audio est effectuée avant l'encadrement de la signature. Cette méthode a l'avantage d'être facilement observable depuis l'extérieur en utilisant le connecteur casque audio. L'application se déroule en cinq grandes étapes :

1. Lecture d'un son.
2. Lecture d'un message à signer dans un fichier texte.
3. Signature du message encadré par deux évènements identifiable.
4. Ecriture du résultat dans un fichier texte.
5. Vérification de la signature et détection d'une erreur éventuelle.

La partie située entre ces deux évènements est une image de la signature RSA. Ces courbes représentant l'activité électromagnétique de la mémoire flash, nous ne sommes pas en mesure d'observer les exponentiations calculées par le processeur. Nous pouvons cependant faire l'hypothèse que ces exponentiations vont occuper une partie non négligeable du temps de calcul. Par conséquent, l'injection sera déclenchées en temps variable par rapport à la fin de la première écriture.

4.4 Exploitation

En quelques heures, nous trouvons la combinaison de paramètres permettant d'obtenir le modèle de faute ciblé. La faute est injectée dans le processeur et permet d'obtenir les résultats suivants :

```
Signature correcte :
1F64C39835B2B7F2104078144CBF2D89C56E5F013E8A33C18CA395541502CB20
25F3D1F566FCE06041F48BB21380B94A4EC9B47C2F4A416619EF8781F26DC357
F097BFB8116612213B77039A68084CB3EE1536A39C280A8F28A42A021A4E66D4
```

Figure 2. Identification sommaire du RSA par encadrement

```
1543F4322FD4B13E220CCF3DC0A3287B1D9E5AC855A2CE29B9EDEE3BC63B91C1
```

```
Signature erronée :
5F284D29FBF8AB0059F9756F9E0AE6FEA9DA173AB598BFB7E6F08324D36755AD
3FF1A78BD9C6F9D7FD20F942A8F2B4592173174C709D5EE80C318B514FD00983
42B4304E5A77440047C2B4380A4CDC04708D2C42D3398BC16CBA6E2F3AEBC81E
8D5C55F32058C023A72E9B68D8D5F505E02A2B93B5D3E59403FD1E923695EE53
```

Par simple calcul de PGCD, nous retrouvons un des paramètres secrets qui suffit à retrouver l'intégralité de la clé secrète.

Grâce aux informations des journaux système, nous avons pu constater que la faute avait engendré une perturbation dans le calcul de l'exponentiation par la bibliothèque *OpenSSL*. On peut donc affirmer que la perturbation s'est faite au niveau de la partie native du système d'exploitation. Il reste aussi à démontrer que l'injection de faute est aussi réalisable sur des portions de code applicatif implémentées en Java et exécutées par la machine virtuelle *Dalvik*.
Sur la même base, nous avons remplacé la signature RSA par une simple boucle

de décrémentation. Le but étant de sortir prématurément de cette boucle en provoquant une faute.

```
//Test_boucle.apk
int i = 1000000;
while(i != 0){
     i--;
}
Log.d("VALEUR DE I : ",""+i);
```

En appliquant la même technique d'injection de faute et sensiblement les même paramètres, nous sommes capables de sortir de la boucle avec une valeur de compteur différente de 0. Les deux images suivantes montrent la différence entre une exécution normale (à gauche) et une exécution fautée (à droite).

Figure 3. Preuve de concept : injection de faute au niveau machine virtuelle

Extrait des journaux systèmes :

```
W/MediaPlayer-cpp( 1592): info/warning (1, 44)
I/global  ( 1592): Default buffer size used in BufferedWriter constructor. It wo
uld be better to be explicit if an 8k-char buffer is required.
D/VALEUR DE I : ( 1592): 954989
I/global  ( 1592): Default buffer size used in BufferedWriter constructor. It wo
uld be better to be explicit if an 8k-char buffer is required.
```

5 Etude de cas

Les deux preuves de concept présentées peuvent être mises en application dans un scénario imaginaire mais réaliste. L'exemple suivant décrit le cas d'une attaque visant une application mobile d'authentification de transaction bancaire

à deux facteurs.

L'application de sécurité Android SecureBankX fournit à l'utilisateur un moyen d'authentifier une transaction bancaire. A partir du contenu d'un SMS émis par sa banque, suite à un achat sur internet initié depuis un ordinateur de bureau, l'utilisateur calcule un cryptogramme grâce à SecureBankX. Ce cryptogramme est ensuite saisi par l'utilisateur sur la page web de confirmation de sa banque et peut ainsi finaliser la transaction. Pour que seul l'utilisateur légitime puisse utiliser ce service, l'application SecureBankX est protégée par un code PIN. Le cryptogramme, est généré par une opération de signature RSA sur le challenge fourni. La clef RSA est unique, elle a été générée par la banque ou le terminal et est inconnue de l'utilisateur. Elle est stockée dans une zone de mémoire sécurisée du téléphone Android, protégée en intégrité et en confidentialité.

Nous pouvons distinguer deux attaquants potentiels cherchant à atteindre des buts différents :

1. Un utilisateur illégitime qui souhaite réaliser des achats avec un couple téléphone/carte bleue volés aux dépends de l'utilisateur légitime.

2. Un utilisateur légitime qui souhaite attaquer "le système" pour créer un clone logiciel de l'application de signature. Ce clone permettrait par exemple, de calculer des cryptogrammes sur des données arbitraires et ainsi briser le principe de non répudiation des transactions (aux dépends de la banque). Par exemple : deux transactions sont effectuées par l'attaquant et son complice, l'une en France (avec le mobile) et l'autre à l'étranger (avec le clone) le même jour. L'attaquant demande à la banque réparation pour l'une des deux transactions en mettant en cause une éventuelle fraude. Les deux cryptogrammes générés étant authentiques, la banque n'a d'autre choix que d'accéder à la demande.

Dans le premier cas, la vérification du code PIN doit être attaquée. Sans contre mesures spécifiques vis-à-vis des attaques physiques, un attaquant visant l'exécution du bytecode Dalvik peut outrepasser le mécanisme d'identification.
Pour le dernier cas, l'attaque par fautes sur le RSA présentée précédemment, permet à un attaquant d'obtenir la partie privée de sa clef et ainsi de pouvoir constituer des clones de l'application de signature.

6 Travaux futurs

L'application possible de cette classe d'attaques aux plateformes mobiles élargit grandement le spectre des capacités d'un attaquant. Nous avons ainsi décidé

de continuer nos travaux pour prouver que la faisabilité des attaques n'est pas liée à un équipement en particulier. Les mêmes tests seront donc réalisés sur différentes plateformes mobiles. Enfin, nous étudions aussi la combinaison attaques logicielles et attaques matérielles, l'objectif étant de profiter de l'effet transitoire d'une injection de faute pour rendre possible une attaque logicielle.

7 Conclusion

Les deux preuves de concept présentées démontrent la possibilité et la relative facilité, de perturber du code natif mais aussi du bytecode interprété par la machine virtuelle *Dalvik*. La primitive de signature RSA-CRT offerte par la bibliothèque open source *Bouncy Castle* est vulnérable aux injections de fautes. Cette bibliothèque ne revendique aucune résistance vis-à-vis des fautes.
Il apparait que cette classe d'attaque est tout simplement ignorée par les développeurs de systèmes d'exploitation et d'applications mobile et seule une sécurisation globale de la plateforme, en partant des couches les plus basses, permettrait de garantir un système résistant aux attaques par faute.

Références

1. Vulnerability in skype for android is exposing your name, phone number, chat logs, and a lot more. April 2011.
2. Charles Arthur. Exploratory android surgery. 2011.
3. Hagai BarEl, Hamid Choukri, David Naccache, Michael Tunstall, and Claire Whelan. The sorcerer's apprentice guide to fault attacks. Cryptology ePrint Archive, Report 2004/100, 2004. `http://eprint.iacr.org/`.
4. Dan Boneh, Richard A. Demillo, and Richard J. Lipton. On the importance of checking cryptographic protocols for faults. pages 37–51. Springer-Verlag, 1997.
5. Jesse Burns. Android phones record user-location according to research. 2009.
6. Jesse Burns. Mobile application security on android. June 2009. Black Hat USA.
7. Christophe Giraud and Hugues Thiebeauld. A survey on fault attacks. In *CARDIS'04*, pages 159–176, 2004.
8. Hagai Bar-El Hamid, Hamid Choukri, David Naccache Michael Tunstall, and Claire Whelan. The sorcerer's apprentice guide to fault attacks. 2004.
9. Bastian Könings, Jens Nickels, and Florian Schaub. Catching authtokens in the wild the insecurity of google's clientlogin protocol. May 2011. `http://www.uni-ulm.de/en/in/mi/staff/koenings/catching-authtokens.html`.
10. John Leyden. iphone 4 jailbreak banks on browser exploit. August 2011. `http://www.thregister.co.uk/2011/08/02/iphone_4_jailbreak/`.
11. John Leyden. Tricky xbox 360 hack claimed to work 1 try in 4. September 2011. `http://www.thregister.co.uk/2011/09/01/xbox_reset_glitch_hack/`.
12. Collin Mulliner and Charlie Miller. Fuzzing the phone in your phone (black hat usa 2009). June 2009. Black Hat Abu Dhabi.
13. Nils. Building android sandcastles in android's sandbox. October 2010. Black Hat Abu Dhabi.

14. Jean-Jacques Quisquater, Pierre Paradinas, Yves Deswarte, and Anas Abou El Kalam. Smart card research and advanced applications vi, ifip 18th world computer congress, tc8/wg8.8 & tc11/wg11.2 sixth international conference on smart card research and advanced applications (cardis), 22-27 august 2004, toulouse, france. In *CARDIS*, 2004.
15. Jean-Jacques Quisquater and David Samyde. Eddy current for Magnetic Analysis with Active Sensor. In *Esmart 2002, Nice, France*, 9 2002.
16. François xavier Standaert (ucl). Electromagnetic analysis and fault attacks : State of the art, 2005.

Fuzzing on the HTTP protocol implementation in mobile embedded web server

Matthieu Barreaud, Guillaume Bouffard, Nassima Kamel, and Jean-Louis Lanet

Smart Secure Devices (SSD) Team – XLIM Labs, Université de Limoges
83 Rue d'Isle, 87000 Limoges, France
{matthieu.barreaud,guillaume.bouffard,nassima.kamel,jean-louis.lanet}(@)xlim.fr

Abstract The fuzzing is a technique which allows to generate invalid, unexpected, or random data to supply them in the various inputs of the software or the protocol to be tested. That allows to find situations not expected by the programmers and sometimes to influence the functioning of the target. Our work aims to check implementations of the HTTP protocol in smart card embedded web servers. For that, we have used the fuzzing method to found vulnerabilities and compliance of this sort of web server. Moreover, working on black box forced us to use PyHAT to collect a maximum of information of the target features. Thus, we can reduce the amount of properties to analyze. Our fuzzing program is based on the Peach framework adapted to our needs. Then, with data model and state model of the target, Peach will generate the random data. We have also defined mutators to represent the mutations types used. Results generated by logs files are finally automatically analyzed to understand the behavior of the application and to detect if some fuzzed data succeed to take up vulnerabilities.

Keywords: smart card web server, fuzzing, embedded HTTP protocol, mobile, security

1 Introduction

The recent years, the evolution of embedded systems and their complexity have increased steadily. Nowadays, it is possible to embed a web server in a smart card. This technology uses the HTTP protocol and allows the holder to provide services and custom interfaces. Due to constrained resources systems in which this technology runs, it should necessary to test it in order to discover bugs and other vulnerabilities.

This new technology, specified by Open Mobile Alliance (OMA), describes the Smart Card Web Server (SCWS) specification based on the version 1.1 of the HTTP protocol for the smart cards mobile phones Java Card 2.2 platform and for low-cost devices. In the other hand Oracle (formerly Sun Microsystem) proposes the Java Card 3 platform which also offers an embedded web server and new features through the enhancement of the framework with new supported Java API and programming. However, it is dedicated to high-cost devices.

Our work aims to test the compliance and the robustness of the HTTP protocol implementation of embedded web servers. Thus, we have choose the fuzzing

technique for its effectiveness in auditing different types of applications and systems. Based on a black box model, we have no knowledge of the target implementation. We are mainly interested to test the HTTP protocol implementation. In a previous work [4], we have presented the effectiveness of this technique in the verification of compliance of the BIP protocol (the transport layer) available on a smart card with the specification defined by ETSI (European Telecommunications Standards Institute) [9].

In this paper, we first explain the architecture of the SCWS and the embedded HTTP protocol then we present a state of the art of the fuzzing technique. In sections 4 and 5 we describe our contribution and explain the main points of our tool, namely : The PyHAT strategy to discover the supported web server features, how to generate testing data and the logging interpretation. We finish by presenting some experimental results and we conclude.

2 SCWS architecture

2.1 Smart Card Web Server

The SCWS is a HTTP/1.1 web server specified by the RFC 2616 standard [10] for embedded smart cards. It is used to provide services, to personalize mobile interface and to facilitate the card administration. The SCWS is both a server, and a client application. In the server mode, it is used by the subscriber with a WAP browser whereas in client mode, a Card Issuer may administrate the SCWS with an (Over The Air)OTA a server. [3,7].

In the server mode, the card communicates *via* a Bearer Independent Protocol (BIP) channel [6,9], which is not the Card Issuer communication channel, allowing the SCWS to be run independently from the Card Issuer network (figure 1). To communicate with the mobile, the card uses following BIP commands :

- `OpenChannel` : opens a communication channel between the card and the mobile,
- `GetChannelStatus` : asks to the mobile the communication channel state,
- `SendData` : sends data to the mobile,
- `ReceiveData` : receives data from the mobile,
- `CloseChannel` : closes a communication channel.

To communicate with the card, the mobile uses two events :

- `DataAvailable` : informs the card that the mobile wants to send data,
- `ChannelStatus` : asks to the card the communication channel state.

These commands are composed of one or many TLV (Tag, Length and Value). They belong to the Subscriber Identity Module (SIM) Application Toolkit (SAT) technology defined by the ETSI [9]. The SAT consists of a set of commands programmed into the SIM which defines how the SIM should interact with the

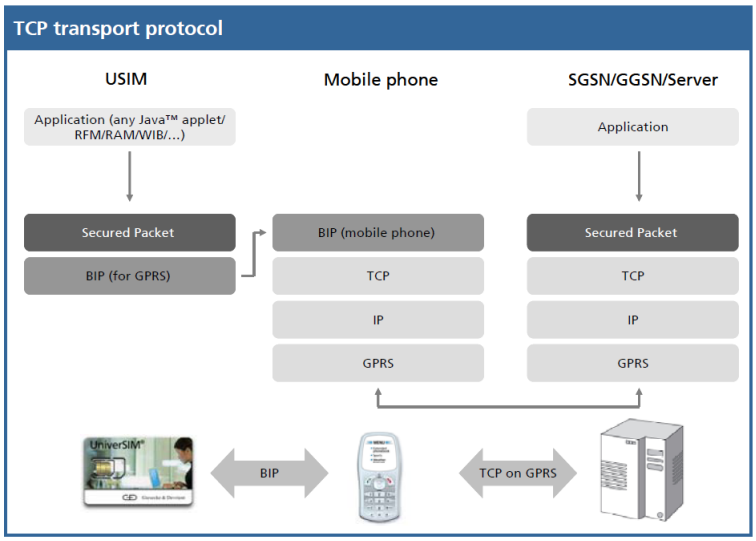


Figure 1. TCP transport protocol

outside. Moreover, SAT initiates commands independently of the handset and the network (proactive commands). In addition to the SCWS application, the whole environment is composed of (figure 2) :

- A handset, in which are implemented :
  - A BIP Gateway to ensure the translation of the data format between the SCWS and the WAP browser,
  - The HTTP application or WAP browser which sends requests to the SCWS via the BIP Gateway, at the subscriber requirement.
- The OTA server which stores the Card Issuer data, needed to remotely administrate the SCWS.

2.2 HTTP Protocol

The HTTP protocol is encapsulated into the BIP transport protocol. A HTTP message has the following components [10] :

- An URI used to identify the requested resource. The URI must be written as defined in the RFC 2616 format :
  `http_URL = "http:" "//" host [":" port] [abs_path ["?" query]]`
- A header which contains information as :
  `Message header = field-name ":" [field-value]`
- A body which contains the HTTP standard methods :
  - The `GET` method is used to retrieve the resource (in the form of an entity) identified by the `Request-URI`.

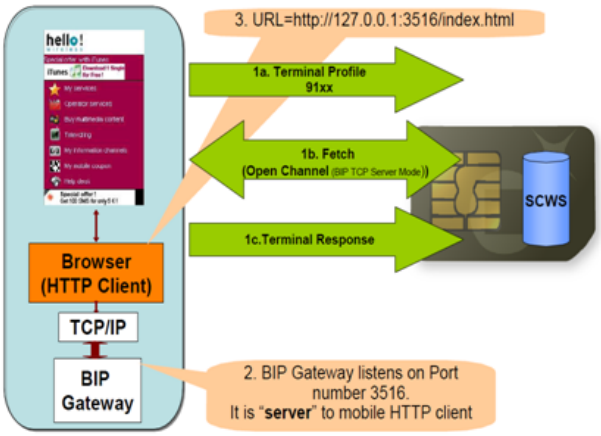


Figure 2. Communication between the mobile and the SCWS

- The `HEAD` method is same as GET except that the server does not return a body message in the response.
- The `POST` method is used to request the origin server accept the resource enclosed in the request as a new subordinate of the resource identified by the `Request-URI` in the `Request-Line`.
- The `PUT` method requests enclosed static resource be stored under the supplied `Request-URI`. This method is only used for the static resources.
- The `DELETE` method allows to delete the resource identified by the `Request-URI` field.
- The `OPTIONS` method represents a request to obtain information about the communication options available on the request/response chain identified by the `Request-URI`.

The `TRACE` and `CONNECT` methods are specific to SCWS :

- The `TRACE` method is used to invoke a remote application-layer loop-back of the request message.
- The `CONNECT` method is used with a proxy which may dynamically switch to be a tunnel.

The first line of the response is always the `Status-Line`, which consists in the protocol version followed by a numeric status code and its associated textual meaning :

- The status code is a 3-digit number, indicating if the request is successfully executed or not :
  - Informational `1xx` indicates a provisional response ;
  - Successful `2xx` indicates that the client's request is successfully received, understood and accepted ;
  - Redirection `3xx` is not supported ;

  - Client error `4xx` is returned for cases in which the client seems to be erred ;
  - Server error `5xx` is returned when the SCWS server has erred or is incapable to perform the request.
- The reason gives a short textual description of the status code
- A text message indicates the error type (same as standard reason-phrase)
- An error style sheet (XML output)

3 State of the art

3.1 The fuzzing technique

Fuzzing is a technique whose aims to find errors in software implementations by injecting invalid data [5,15]. The main goal of fuzzers is to crash the machine, application, protocol, etc. The main advantage is to search vulnerabilities with a low-cost material. Fuzzing data can be generated in three different ways [13] :

- Random data generation which has the inconvenient to be blind and, in most cases, these data are filtered and rejected by the target.
- The fuzzer generates invalid data from a data model created by the user. It sends them to the application or protocol to test. This method is time consuming because it needs to know the protocol integrally but it is the best effective method.
- The mutation fuzzer takes a well known and valid session like a file or a network capture to mutate and send it to the fuzzed application or protocol. This method needs not target knowledge but is limited by treated test file cases.

Most existent fuzzers are designed to test network protocols such as in [16] that provides a tool to test TCP/IP layer, but their use is not limited to this unique area. This technique is also used for testing protocols implemented on smart cards. In [4] we presented a fuzzer wich verify the compliance of the BIP protocol implementation with specification defined by ETSI. [12] presented a fuzzer based on the Sully framework[14], and dedicated to test and find flaws in smart card secure payment protocols(EMV).

There are many fuzzing frameworks based on APIs that my be used to implement tools for auditing at different levels (files, API, arguments for a command line, standard input, etc.) as :

- SPIKE [2] is one of a first fuzzing framework ; it is written in C and dedicated in particular for developing network protocol fuzzers. It introduces a concept of “blocks”, the data structures are broken and represented as a block that allows for abstracted construction of various protocol layers with automatic size calculations.

- Peach [8] is an advanced and robust fuzzing framework written in Python. It provides an XML file to create a data model and state model definition which is used to create data mutations. It is a flexible framework that allows to define our fuzzing strategy (modify one or many data elements at a time, define mutators to use, modify some parts of our model, change the flow of our state model,etc).
- Sulley [14] is based on SPIKE technique generation of data. It also includes many other important features as : managing the different states of protocols, using agents to monitor a target (monitoring network communications, detecting eventual faults, VMWare Control), and offers some drive that facilitate the use of Sulley.
- Fusil [11] is a Python library used to write fuzzing programs, it was initially dedicated to GNU/Linux programs on command line. But it can also be used in different environment like web browsers, GUI applications, etc. It provides a set of tools to attack a target (start process, start network client or server, and create mangled files, etc).

In our researches we have chosen Peach thanks to its flexibility, using both data and state models, and mutation strategy. This choice allows us to parallelize the fuzzing execution which reduces the execution of the test suites. It is very important, in particular while using smart cards low-bandwidth communication.

4 PyHAT

If we fuzz each method and their associated header, we have a huge amount of tests. These tests may spend a lot of time with a low bandwidth device, like a smart card. In order to decrease this amount of tests, we developed a tool which searches the implemented HTTP features in the server. This tool, PyHAT, for Python HTTP Assessment Tool, may :

- Detect implemented HTTP methods. The main feature of PyHAT is to find them,
- Detect case sensitive requests : the HTTP protocol specification describes that the sent requests to the server must be case sensitive.
- Detect the supported HTTP versions,
- Detect supported encoding data methods : to save bandwidth, the HTTP protocol allows to encode the resource within an encoding method chosen by the client. The RFC defines `gzip`, `compress` and `deflate` algorithms. The `identity` method is used to get a resource without encoding,
- Detect server analysis request fields : previously, we explained how a HTTP request is formed. In addition of the first HTTP request line, other fields may be used but they may not be parsed by the HTTP server,
- Find the resources contained in the web-server,

– Output the result in a XML file which may be parsed by our fuzzer.

Analysis strategy This part aims to define the strategies used for each previously functions and the expected results.

As we explained before, the HTTP protocol defines some return values. We will see that these values may be interesting and they may characterize the tested HTTP server

Detect implemented HTTP methods : The RFC 2616 [10] describes :

> *"A server which receives an entity-body with a transfer-coding it does not understand SHOULD return 501 (Unimplemented), and close the connection."*

The strategy chosen to determine the methods implemented by the HTTP server is based on this response numeric status code. In practice, for each method, we define a request which respects the RFC 2616 explained previously. In the response, only the first line may be isolated to obtain the return code and to determine if the method is implemented or not.

Detect request case sensitive : The test case sensitive is used to the requested method and fields where the specification does not define any specific rule. To perform these tests, we define a first request in which the method is written in lower-case. In a second step, the method is formatted in accordance with the specification, but other fields are written with alternately of lower and upper letter case. In each case, analysis of the response is due to return code. Indeed, if the server responds with a code `200` (OK), we deduce that there is no distinction between upper-case and lower-case.

Detect the supported HTTP version : Finding the supported protocol version of a HTTP server is not a friendly task. Indeed, when we send a request to a HTTP server it should, when possible, use the same version as the client. Our analysis is therefore based on this fact and follows this algorithm :

– Send a request with a specific HTTP version,
– Analyze the HTTP version used in the response,
– If the HTTP version is the same, this version is supported by the HTTP server else, it is not supported.

We try this algorithm for each HTTP version(`0.9`, `1.0` and `1.1`). The RFC 2616 defines a return value (505) if a HTTP version is not supported. In fact, we note that some servers do not return an error. This is why we choose to perform this test with the algorithm that works for all cases.

Detect supported encoding data methods : According to the content-encoding header managed by the client, the server must respond if possible with the encoding supported by it and no encoding else. Currently, when a client specifies an encoding, it informs the field `Accept-Encoding`. The server receives this request and analyzes the field. If this value is an encoding mechanism managed by the HTTP server, it will respond in the field `Content-Encoding` whose value and that of the encoding used.

There are four defined encodings :
- `gzip`
- `compress`
- `deflate`
- `identity`

In order to evaluate the encoding supported by the HTTP server, we send a request for each encoding value and we analyze the field `Content-Encoding` in the response request. There are then two possible cases :
- There is no encoding field in the server response :
  - The encoding format is not supported,
  - The encoding format is implicit and is the `identity`.
- The value of field `Content-Encoding` is the same as the client request. We conclude the encoding is managed by the HTTP server.

Detect server analysis request fields : In contrast to previously analysis, the detection of the fields parsed by the server is a hard problem. Indeed, we do not have all knowledge to determine if a field is expected or not.

The strategy chosen is to send a valid request (defined by the HTTP specification) with a specific field. Moreover, according to the value of this field, the RFC 2616 describes the expected value from the server. For the fields where there is no defined response in the specification we suppose by default that are analyzed. The table 1 summarizes the strategy used for each field.

Description : We describe in the following each field listed on the table :

1. Forbidding to receive HTML content for a `Content-Type` set up to text should provide in the server sending a response which informs its available to provide such data (code `406`).
2. If the server may not send a response with an requested encoding by the client, it should return the expected response `406`.
3. Give an undefined value of the expected field should hold the HTTP server to return a `304`. error class value.

Table 1. Request fields analysis strategies

id	Field	Value	Expected value
1	Accept	text/plain ;q=1, text/html ;q=0	Return code 406
2	Accept-Charset	ISO-8859-1,utf-8 ;q=0	Return code 406
3	Expect	10-continue	Return code 4xx
4	If-Modified-Since	Sun, 06 Nov 2014 08 :49 :37 GMT	Return code 200
5	If-Range	Random UUID	Return code 200
6	If-Unmodified	Sun, 06 Nov 1970 08 :49 :37 GMT	Return code 412
7	Range	1-30	Return code 206
8	Content-Length	1 000 000 000 000 000 000 000 000 000	Return code 413

4. If the requested resource has not been changed since the specified date, the server must return a code 304. In our case, this date is invalid. So, the server should return a code `200`.
5. The `If-range` field value is randomly generated, it is improbably match to an entity on the server. The expected code is `200`.
6. A requested unmodified resource since the specified time must response a return code `412` by the server.
7. A partial data response is of type `206`, else, the requested range is not available or this field is not analyzed.
8. If a server receives a request with a content length greater than it can handle, it must return code `413`.

Find the resources contained in the web server : To discover all URI of a website, we use the same way of any website mirroring software. Starting with the index page of the website, we collect all strings that match to a certain regular expression (in our case, the attribute `href`). Then we restart the operation for each discovered URI and we do some checks :

- This URI is new,
- If this URI refers to an external resource. We save it only if it has the same IP address of the analyzed HTTP server.

Fuzzing-FE : In order to make this tools user-friendly, we have developed, a user interface for Smart Fuzz. This interface is based on QT4.

This user interface provides a textual display of the analysis of PyHAT and fuzzing tool. A tabular is created of each output. The device may be chosen by the user, in the specific list to fuzz on a SCWS or on any other HTTP servers (including JC3). In addition, the fuzzing on smart card may be parallelized to reduce the time needs to make all tests.

5 The Smart Fuzz tool

5.1 Aims

We want verify the conformity of the HTTP protocol implementation and analyze the behaviour of the card in the case of invalid inputs : card crash or has an unexpected behaviour. For that, we analyze the status word and the returned response. The status word corresponds to the card state after execution of the request.

Moreover, we want to have a generic data model wich represents the HTTP protocol implemented either in the SCWS, in JC3 connected edition or in a general purpose server. Indeed, our fuzzing tool can fuzz a web server defined by an IP address and a port or an embedded web server in a smart card. Finally, we choose the mutation type by defining an array of string values.

5.2 Architecture

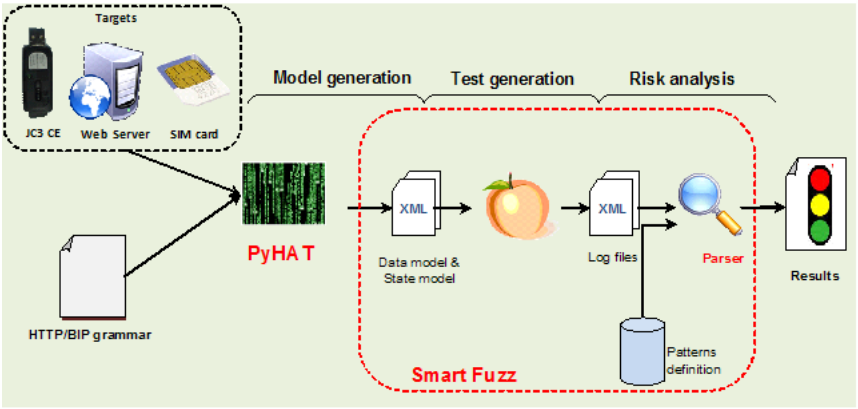


Figure 3. Our fuzzing architecture

The first step is to use our application PyHAT which discovers dynamically the implemented features on the card (figure 3) by comparing the responses to the specification. That reduces the number of cases to evaluate. This tool creates a XML file used by the fuzzer Peach [8]. Then, we have an interface which makes the link between :

- PyHAT, the fuzzer Peach and the targets,
- The fuzzer Peach, the targets and the logs.

Finally, we have risk analysis tool which identifies the vulnerabilities found in the logs. The principle is to look for predefined patterns. It helps the developer to find, in the logs, the searched patterns.

5.3 Test generation

Peach Peach intends to generate invalid data on some protocols. For that purpose, it uses XML file containing :

- the data model, representing the structure of the protocol,
- the state model defining the basic logic states needed to test the protocol,
- the publisher describing where the data are sent,
- the mutator representing the mutation types to use.

We modified Peach to meet our expectations. We created a publisher by using the library Pyscard [1] because Peach does not permit to communicate with smart cards. This publisher encapsulates the HTTP request in the BIP protocol automatically. It logs all the APDUs sent and received by the smart card. Moreover, we have implemented basic functions in order to fuzz the HTTP protocol :

- A `Choice` tag permitting to choose automatically a value among a value set,
- A `minOccurs` and `maxOccurs` attributes permitting to define the minimum and maximum number of element occurrence.

Peach allows to parallelize the test execution using the `-p` argument.We need to have for each card a XML files where we indicate in the publisher the card to test. The parallelization is done automatically by Peach, it is a way to reduce the computational time of a fuzzing session.

Fuzzing strategy We automate the generation of data and state models corresponding to methods and headers of the XML file created by PyHAT. This tool creates several files, one for each method, for each header and for each card to fuzz. Then, we use the Peach fuzzer with our own modifications to generate random data corresponding to the data models generated. To create invalid data we use string mutations. We have a static array of string containing several special characters. Moreover, we mutate the default string to create an unexpected result.

5.4 Logs analysis strategy

In order to find some unexpected behavior, we store each request and response in ordered log files. We have a log for each fuzzed method and for each header. Moreover, for each request, we store :

- the HTTP sent and response requests,
- the BIP commands, if we have an overflow inter-layer.

We designed a risk analysis program that analyses in a log file patterns corresponding to unexpected properties. It looks for path of log folder root on peach pit files. Then it parse a file witch specifies tags are occur in this XML file.

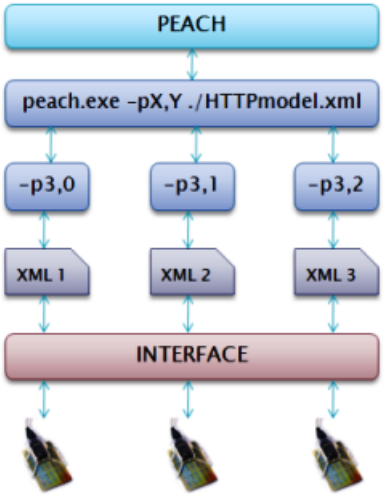


Figure 4. Parallelization

The risk analysis program browses each log file to search a given TAGs. When a TAG found, the request and corresponding response are stored as well as mutate methods and the headers. Finally the file analyzer classifies its results into two different ways :

- Select request/ response by TAGs.
- In a tree shape, associating an TAG to specified methods with corresponding headers. For example, we defines a tag "TAG" witch is only thrown by methods (PUT or DELETE) with headers (etag or expect).

6 Experimental results

Our experimentations are based on products of a major smart card manufacture. The fuzzing results were different depending on the used smart card, this is certainly due to different implementations (security features) between the smart cards.

Using PyHAT we found some cases where the return code does not match like the specification. When the method is not implemented, the embedded HTTP server returns `400` (`Bad Request`)while it should return the status code `501` (`Not Implemented`). We also test our tools on the Java Card 3 platform. The results demonstrate that the implemented HTTP does not support all HTTP methods of version 1.1 as defined in the Java Card 3 specification, However, to clarify that the used Java Card 3 device is a prototype, so only a part of the specification is implemented.

With the smart-fuzz we found some vulnerabilities that we present in the following :

- **Add or delete a resource :** we have noticed some cards accept the PUT and DELETE commands and administrative commands. The PUT command allows creating or modifying a resource and the DELETE command allow deleting a resource. Moreover, the administrative commands, used to add or delete a user, get a list of files in a directory... should be sent in remote by the card Issuer. However, we could send these commands in local and create for example a new user.
- **Invalid if-match :** when we send an invalid if-match header, the card returns a 500 error which means an internal server error while the card should send a 501 error (Not implemented) or a 400 error (Bad request).
- **The host mandatory field** corresponds to the IP address and the port number where the request is sent. This field should be mandatory whereas in the card, **it is ignored**.
- When we send an empty request to the card, a **Java exception was thrown with the 0x6F00 status word**.

7 Conclusion

In this paper we explored the efficiency of the fuzzing method to discover vulnerabilities or implementation errors in the HTTP protocol embedded on smart card. We used the Peach framework to develop our Smart Fuzz tool that generate its own data model and state model to generate fuzzing data. This tool is dedicated to any web servers including embedded web servers based on SCWS or Java Card 3 platform. We use black box approach, for that, PyHAT discovers all implemented functionalities on the target, in order to limit our analysis on only existent properties.

However, the smart cards have a low bandwidth, so, these tests may be parallelized by fuzzing multiples smart cards at the same time. For each smart card a different set of generated fuzzing data is associated. Results are automatically analyzed by comparing the generated response of the smart card (HTTP response and status word) with a predefined patterns associated to some vulnerabilities. If match exists this means a presence of a vulnerabilities, and we can provide the complete commands sequences that leads to this state. However Smart Fuzz should be improved to reduce amount of a false positives wich is the next step.

Acknowledgments

The authors thank Mamadou Balde, Amine Belhocine, Silvère Cainaud, Jérémie Clement, Romain Severin, Nicolas Tarriol and Lylia Tikobaini for their contribution to this work.

Références

1. Pyscard - python for smart cards. `http://pyscard.sourceforge.net/`.
2. D. Aitel. An introduction to spike, the fuzzer creation kit. *immunity inc. white paper*, 2004.
3. Open Mobile Alliance. Smartcard-web-server. Technical report, OMA, 2008.
4. M. Barreaud, J-L. Lanet, and J. Iguchi-Cartigny. Analyse de vulnérabilités sur cartes à puce à serveur web embarqué. *SAR-SSI*, 2011.
5. Gabriel Campana. Fuzzing : injectez des données et trouvez les failles cachées. *MISC*, pages 28–37, SEPT./OCT. 2008.
6. Giesecke & Devrient. Bearer independent protocol (bip). Technical report, G&D, 2006.
7. Giesecke & Devrient. Smart card web server - merging the sim and the world wide web. Technical report, G&D, 2007.
8. Michael Eddington. Peach fuzzing platform. `http://peachfuzzer.com/FrontPage`.
9. ETSI. *ETSI TS 102 223 - Smart Cards : Card Application Toolkit (CAT)*, 2010.
10. R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Rfc 2616 : Hypertext transfer protocol–http/1.1, june 1999. *Status : Standards Track*, 1999.
11. M.Gunes I.Maceno. The Fusil Project. `http://www.labri.fr/perso/fleury/courses/SS08/download/memoirs/adjej-gunes-maceno-memoire.pdf`.
12. J. Lancia. Un framework de fuzzing pour cartes à puce : application aux protocoles emv. *SSTiC*, 2011.
13. C. Miller and Z.N.J. Peterson. Analysis of Mutation and Generation-Based Fuzzing, 2007.
14. A.Portnoy P.Amini. Sulley : Fuzzing framework, 2007. `http://www.fuzzing.org/wp-content/SulleyManual.pdf`.
15. A. Takanen, J. DeMott, and C. Miller. *Fuzzing for software security testing and quality assurance*. Artech House Publishers, 2008.
16. S. Xiao, L. Deng, S. Li, and X. Wang. Integrated tcp/ip protocol software testing for vulnerability detection. 2003.

Smartphones : La sécurité de votre poche...

Ivan Fontarensky

Cassidian, an EADS company

Résumé La protection de la vie privée est dorénavant une préoccupation partagée par tous. Or, le smartphone est rapidement devenu un outil plus personnel que l'ordinateur. Pourtant, il ne semble pas solliciter plus d'attention en terme de sécurité. L'objet de cet article est de présenter de manière pédagogique un aperçu des informations qui peuvent être facilement piratées.

1 Introduction

Les smartphones occupent une part de plus en plus importante dans la vie de tous les jours. Nos cousins québécois les appellent d'ailleurs *ordiphones*. La différence semble subtile mais importante. Toujours en poche, ils accompagnent l'utilisateur dans ses tâches quotidiennes.

Les smartphones sont réellement de petits ordinateurs. Les différents constructeurs font en sorte que le produit offre le maximum de fonctionnalités pour l'utilisateur : qu'il soit agréable à tenir en main, qu'il soit intuitif et ergonomique et qu'il soit fiable.

Pour Winn Schwartau, expert reconnu en sécurité, chaque innovation technologique peut être réutilisée pour devenir une arme. Cela peut paraître improbable pour un simple téléphone. Pourtant l'idée selon laquelle la cyberguerre est un des enjeux majeurs du début de notre 21e siècle devient progressivement une idée de plus en plus partagée.

Cet article présente un panomara des attaques existant qui visent les smartphones.

2 Quelques attaques sur smartphone

La connectivité des smartphones en fait de merveilleux outils pour naviguer simplement et rapidement. Ils comptent de plus en plus d'utilisateurs, avertis ou non. En effet, 59% des Français *surfent* par le biais d'un navigateur mobile de manière quotidienne [10], [9]. En parallèle, de nouvelles attaques apparaissent, spécifiques aux smartphones.

Les enjeux en terme de securité sont donc importants sur ces nouvelles plateformes.

2.1 Tapjacking

Des chercheurs de Stanford ont montré dans un article [11] qu'il était possible d'adapter des attaques connues aux navigateurs de bureau sur les navigateurs de smartphones. Cependant, certaines étapes sont simplifiées. Leurs attaques sont détaillées dans les sections suivantes.

Masquer l'interface Du fait de la taille réduite des écrans, les navigateurs mobiles se concentrent sur l'affichage du contenu. Les informations relatives aux sites habituellement affichées dans les barres d'adresses sont masquées. Les utilisateurs voient donc très peu l'interface et ne sont donc pas forcément familiarisés avec les informations présentes, l'important pour eux étant de visualiser la page demandée.

Grâce à cela, un site malveillant peut effacer l'interface initiale du navigateur pour la remplacer par la sienne. Cela lui permet d'usurper l'identité d'un site web. Il est alors possible de faire croire à l'utilisateur qu'il se trouve sur un site sécurisé, avec une adresse en HTTPS, grâce à une simple image. La petite taille de l'écran rend donc ce type d'attaques plus efficace et plus facile sur les smartphones.

Par exemple, en plaçant ces quelques lignes de code dans une page web, il est possible de cacher l'url du navigateur web.

```
<body onload="setTimeout(function()
        {window.scrollTo(0,1)},100);">
</body>
```

L'interface graphique constituant l'une des principales alertes contre les attaques par MITM (*Man In The Middle*), les protections classiques s'appliquent difficilement.

Les utilisateurs ont de plus en plus tendance à naviguer au moyen de leurs smartphones. L'affichage limité sur ces appareils, et donc la faible quantité d'informations que l'on peut y faire figurer, ne permettent pas de sensibiliser les utilisateurs aux aspects sécurité.

Observons rapidement l'interface du navigateur d'iOS, d'Android, et BlackBerry (Fig. 1). Le premier cas montre un site non sécurisé. Le deuxième est un site sécurisé grâce à un certificat auto-signé. Le dernier est validé par une autorité de certification.

Sur Mobile Safari de iOS, repérer les différents niveaux de sécurité dans la barre de navigation relève de l'exploit. Un simple cadenas et un changement de couleur différencient les niveaux de sécurité. Android, de son côté, ne laisse apparaître aucune différence entre les icônes signalant un site DV (*Domain Validation*

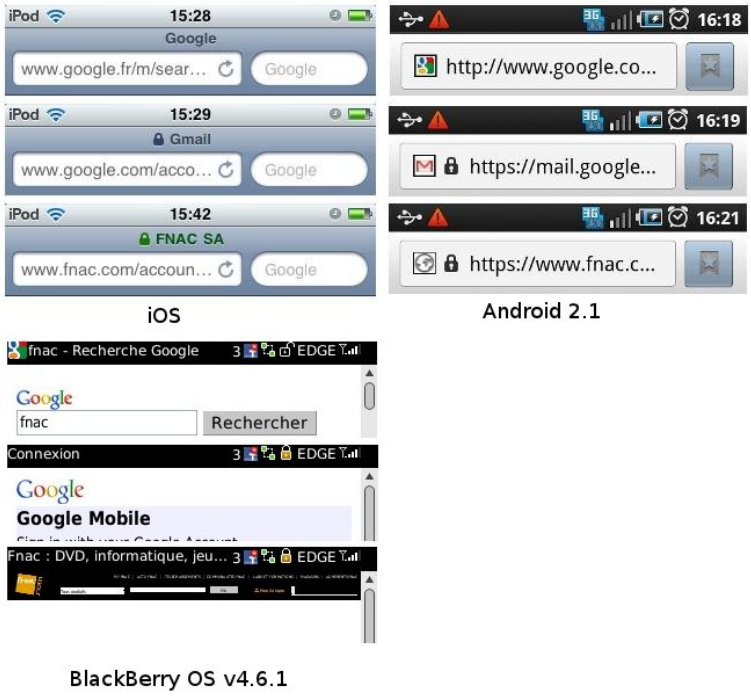


Figure 1. Les barres de navigation selon la sécurité

- certificat signé par une autorité après une faible vérification) et un EV (*Extended Validation* - certificat signé par une autorité après une forte vérification).

S'il n'est pas possible de distinguer facilement les niveaux de sécurité, il serait bon de recommander une utilisation plus systématique des protocoles sécurisés. Le plus connu est HTTPS pour la version sécurisée de HTTP. Il existe aussi HSTS pour *HTTP Strict Transport Security*. Des standards de visualisation sont aussi à recommander afin de sensibiliser l'utilisateur.

2.2 Vol de clic

L'apparence des navigateurs a été améliorée afin d'optimiser l'affichage des pages. Il est maintenant possible de zoomer sur une page grâce à un *pinch* (pincement). Les versions mobiles, centrées sur le contenu, peuvent appeler automatiquement ces fonctionnalités.

Pour rappel, le *click-jacking*, ou "vol de clic", consiste pour un attaquant à forcer sa victime à cliquer sur un bouton ou un lien ayant une action spécifique et invisible pour l'utilisateur.

Les navigateurs mobiles facilitent cette attaque grâce à leur capacité de zoom. Il est maintenant possible d'agrandir la surface de la zone "malveillante" donc d'augmenter la facilité d'un clic piégé. Récemment, il était encore possible de

forcer sur Twitter le clic d'un utilisateur sur un lien invisible pour envoyer un message prédéfini sur son mur.

```
<meta name="viewport" content="width=device-width,
initial-scale=10.0, maximum-scale=10.0, user-scalable=no"/>
```

L'illustration 2 qui suit présente une façon de tromper un utilisateur (ici sur iPhone) en lui présentant un message de réception de SMS. Cependant, cette fausse page est recouverte d'une zone invisible dont le contact tactile (*tap*) exécutera l'attaque sur Twitter précédemment évoquée.

Figure 2. Exemple d'écran d'iPhone pour TapJacking

2.3 Caffe Latte

Aujourd'hui, le smartphone nous accompagne partout. Connectivité et mobilité deviennent liées. L'utilisateur configure donc ses réseaux Wi-Fi favoris. Même hors de la couverture de ces réseaux, le smartphone scanne les réseaux Wi-Fi. S'il est déjà enregistré auprès des bornes diffusées, une demande d'authentification Wi-Fi est envoyée. Même si une grande partie des détenteurs de Smartphones désactivent le Wi-Fi quand ils sont hors couverture, il reste toujours les personnes qui maitrisent mal l'outil technologique, ou tout simplement qui oublient de l'eteindre.

L'attaque *Caffe Latte*[5] consiste à écouter les émissions des différents smartphones à proximité.

Celles-ci contiennent les identifiants des bornes pré-enregistrées.

L'étape suivante de l'attaque consiste à simuler un de ces points d'accès, de forcer une authentification, puis de capturer le trafic. Ce qui permet à un attaquant d'effectuer un man-in-the-middle.

Ce scénario d'attaque peut être réalisé grâce à la suite AirCrack avec la commande suivante.

```
#aireplay-ng -6 -h xx:xx:xx:xx:xx:xx \
                -b xx:xx:xx:xx:xx:xx -D mon0
```

Tout cela ne prend que très peu de temps : celui d'un café au lait, d'où le nom de l'attaque.

2.4 Vol de session

0Une dernière attaque clôturera ce rapide panorama : l'attaque du vol de session. Celle-ci s'inscrit dans un scénario où dans un premier temps l'attaquant aura pu effectuer une attaque Caffe-Latte, et dans un second temps essayer de récuperer tout les secrets du téléphone en commençant par le vol des sessions.

Cette attaque n'est pas récente, elle existe depuis plusieurs années dans l'univers des ordinateurs portables, mais elle prend une plus grande ampleur chez les smartphones car leur connectivité les rend plus vulnérables.

Plusieurs outils existent : FireSheep[7] pour Firefox, et FaceNiff[6] développé pour Android.

3 Analyse des données

Il est assez surprenant de voir que, avec la plupart des smartphones actuels, il est aisé de mener une analyse forensic comme l'expliquent certains livres [12] ou certains sites Internet. Un commerce rentable se développe où l'on voit des sociétés (telles que Cellbrite avec son UFED) spécialisées dans la récupération des données téléphoniques s'intéresser à l'analyse post-mortem.

Les constructeurs proposent eux-même des outils permettant de faire une sauvegarde du téléphone comme le font RIM ou Apple.

Cette partie se focalise particulièrement sur les données stockées sur les mobiles, ce qui implique que les données évoquées ici dépendent du système d'exploitation.

3.1 Géolocalisation

Durant le mois d'avril 2011, une polémique a fait rage dans le monde numérique : *le problème de la géolocalisation des smartphones*. Les articles se sont succédé :

– Apple suit à la trace les utilisateurs d'iPhone [3],
– L'intolérable indiscrétion du mouchard de l'iPhone,
– Android possèderait un mouchard [2].

Les communiqués des fabricants expliquent la présence d'un tel programme sur leur plateforme à des fins de collecte de données relatives aux hotspots Wi-Fi et aux antennes-relais à proximité.

Les iPhone et iPad 3G stockent leurs différentes positions dans un fichier non protégé, facile à récupérer et à analyser. L'outil iPhoneTracker[8] permet d'analyser le fichier *consolidated.db* des *iProduit* qui se trouve à cet emplacement :

```
Library/Caches/locationd/consolidated.db
```

Cette base de données est constituée de deux tables intéressantes : *CellLocation* et *WifiLocation*. Il est possible à partir de ces données de récupérer la position (latitude et longitude) du combiné à un instant donné (timestamp). Quant à Android, ces données de géolocalisation sont stockées dans les fichiers "cache.cell" et "cache.wifi", à l'emplacement suivant :

```
data/data/com.google.android.location/files
```

Ce fichier recense les coordonnées des 50 dernières antennes-relais avec lesquelles le mobile a été en contact et les 200 derniers réseaux Wi-Fi qu'il a repérés. Une fois les données extraites, il est possible de retracer l'itinéraire d'un utilisateur (Fig. 3). Selon la fréquence et l'heure des positions, il est aussi possible d'estimer son domicile, son lieu de travail et ses habitudes.

Il est possible d'automatiser[1] la récupération de ces informations et d'en ressortir la position en quelques secondes pour les obtenir sur un tracé sur Google Maps.

Nous pouvons nous étonner que des applications puissent être en mesure d'exploiter les données de geolocalisation et de stocker ces dernières.

Ainsi *cooliris*, la visionneuse des images sur les plateformes Android, enregistre dans un fichier la position où les images ont été prises.

3.2 Données de la vie privée

Le smartphone, véritable petit assistant personnel, contient un seul profil utilisateur, contrairement aux ordinateurs qui peuvent avoir plusieurs profils.

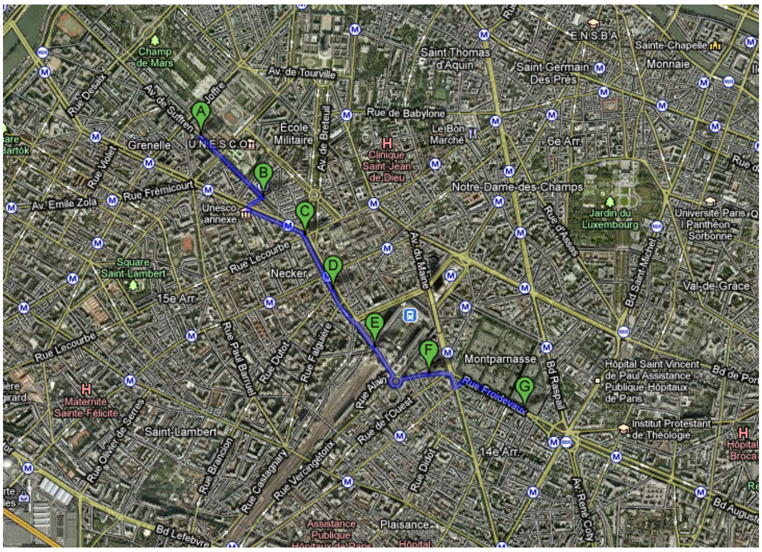


Figure 3. L'itinéraire d'un utilisateur de smartphone

```
/sdcard/Android/data/com.cooliris.media/cache/
geocoder-cache
```

Les informations personnelles du propriétaire, comme les messages, les courriels, l'agenda ou encore les photos, sont donc accessibles très facilement en quelques opérations. La simplicité d'utilisation est l'objectif premier de ce type de matériel.

Les données stockées touchent de très près la vie privée de l'utilisateur. Cela nécessiterait une protection sur des données, tel que du chiffrement. Pourtant ce n'est pas encore le cas.

De nombreuses applications stockent en clair des données aussi sensibles que des identifiants et des mots de passe (Fig. 4). Des applications comme "PayPal" n'ont d'ailleurs pas le même niveau de sécurité sur iPhone et Android.

Applications	Android	iPhone
AIM	Les données sont stockées non sécurisées	
Amazon Mobile	Les données sont stockées non sécurisées	
LinkedIn	Mot de passe en clair	Les données sont stockées non sécurisées
PayPal	OK	Certaines données sensibles ne sont pas protégées

Figure 4. Les niveaux de sécurité par applications et plateformes par viaforensic (Juin 2011) [4]

3.3 Configuration Wi-Fi

À chaque connexion Wi-Fi, Android demande à pouvoir enregistrer le réseau Wi-Fi actuel. Ainsi, l'utilisateur peut se reconnecter de façon automatique et transparente lors d'une session ultérieure. Selon le gestionnaire de connexion, l'engistrement des clés Wi-Fi s'effectue dans des fichiers différents mais par défaut toujours en clair (au moment où nous rédigeons cet article).

Il sera aisé de retrouver les clés Wi-Fi des AP déjà utilisés pour les téléphones de type Samsung Galaxy avec la commande montrée en Fig. 5 ou Fig. 6

```
#cat /data/Wi-Fi/bcm_supp.conf
ctrl_interface=DIR=/data/misc/
Wi-Fi GROUP=Wi-Fi
update_config=1
device_name=Wireless Client
manufacturer=Tutu Electronics
device_type=1-0050F204-1
network={
        ssid="Wi-Fi_example"
        key_mgmt=NONE
        auth_alg=OPEN SHARED
        wep_key0=2105421567043545640456009 6
        priority=1
}
network={
        ssid="Cesar-Wi-Fi"
        psk="m4d_d0g_Wi-Fi_ex4mple"
        priority=9
}
```

Figure 5. Contenu d'un fichier de configuration Wifi sous Android

Sur les HTC avec le système Android on retrouvera plutôt ces informations dans le fichier :

```
#cat /data/misc/wifi/wpa_supplicant.conf
```

Figure 6. Contenu d'un autre fichier de configuration Wifi sous Android

3.4 Compte Mail

Les applications de courrier électronique font partie intégrante du système d'exploitation d'un smartphone. A quelques exceptions, lorsque le mot de passe est enregistré sur le téléphone, celui-ci est stocké en clair.

```
#cd data/data/com.Android.email/databases

#sqlite EmailProvider.db

sqlite> .tables
Account HostAuth Message Message_Updates Attachment
Mailbox Message_Deletes  Android_metadata

sqlite> SELECT * FROM HostAuth;
1|imap|imap.gmail.com|993|5|victim@gmail.com|passw0rd||0
2|smtp|smtp.gmail.com|465|5|victim@gmail.com|passw0rd||0
```

Figure 7. Contenu de la base de donnée : EmailProvider.db

Il est possible de retrouver dans la base de données *EmailProvider.db* d'Android, l'enregistrement des comptes mail (Fig. 9)

Les autres tables ne semblent pas nous apporter d'autres informations intéressantes, exception faite de l'entête des courriels envoyés avec leurs destinataires. Si l'on veut récupérer le corps du message, il faudra regarder la table "Body" dans la base de données "*EmailProvider.db*".

Depuis juillet 2011, les développeurs d'Android ont corrigé cette vulnérabilité. Ceci étant, la politique de Google ne prend pas en compte les smartphones d'ancienne génération.

Il est possible de retrouver dans la base de donnée *accounts.db* (Fig. 8)

```
#cd /data/system

#sqlite accounts.db

sqlite> SELECT  * FROM accounts;
_id|name               |type                      |password
1  |victim@gmail.com   |com.google                |AFcb4K...
2  |victim@hotmail.com|com.facebook.auth.login|
```

Figure 8. Liste des comptes sous Android

Les informations privées ne sont pas bien protégées dans un smartphone et sont facilement accessibles à des tiers, mais progressivement l'evolution et la mise à jour des applications montrent qu'un (petit) effort est fait dans ce sens.

3.5 Réseaux sociaux

Véritable phénomène actuel, les réseaux sociaux s'invitent dans les smartphones. Prenons le cas de Facebook. Même si le mot de passe n'est pas disponible avec l'application Facebook sur Android et iPhone, il est tout de même possible

d'accéder à une partie du carnet d'adresses, ainsi qu'à plusieurs liens vers les photos des profils ou des albums qui ne nécessitent aucunement d'être connectés à Facebook pour être consultés.

```
#sqlite /data/data/com.facebook.katana/databases/fb.db
sqlite> .tables
albums                mailbox_messages search_results
Android_metadata      mailbox_threads  stream_photos
default_user_images   mailbox_users    user_statuses
friends               notifications    user_values
info_contacts         photos
```

En revanche pour autoriser l'application mobile à se connecter au réseau social, il faut que l'utilisateur ait préalablement donné les permissions au programme.

L'authentification se fait par l'API que Facebook met à disposition [1]. Il faut pour cela que l'application ait accès à certains secrets nécessaires qui sont soit stockés dans la table '*user_values*', soit qui peuvent être codés en dur comme le "*CLIENT_ID*" et le "*SECRET_ID*". C'est entre autre le cas pour l'application *com.facebook.katana*

```
CODE:000A9A88  Method 4978 (0x1372):
CODE:000A9A88  private void
CODE:000A9A88  com.facebook.katana.service.method.
                   GraphApiExchangeSession.<init>(
                   java.lang.String p0)
               ...
CODE:000A9A88  const-string  v0, aPost # "POST"
CODE:000A9A8C  const-string  v1, aOauthExchange_
               # "oauth/exchange_sessions"
CODE:000A9A90  const-string  v2, aHttpsGraph_fac
               # "https://graph.facebook.com/"
<void GraphApiMethod.<init>(ref, ref, ref)>
CODE:000A9A94  invoke-direct {this, v0, v1, v2},
               ...
CODE:000A9AAC  const-string  v1, aClient_id
               # "client_id"
CODE:000A9AB0  const-wide    v2:v3, 0x31A17C8050
               ...
CODE:000A9ACC  const-string  v1, aClient_secret
               # "client_secret"
CODE:000A9AD0  const-string  v2, a62f8ce9f74b12f
               # "62f8ce9f74b12f84c122cc23437a4a32"
```

Figure 9. Desassemblage d'un fichier DEX

Un attaquant ayant un accès physique au smartphone de sa victime pourra ainsi usurper l'application qui a été autorisée et naviguer sur le réseau social afin

1. `https://developers.facebook.com/docs/authentication/`

de consulter les contacts, les messages, et toutes autres informations personnelles du detenteur du mobile [2].

4 Conclusion

La tendance actuelle est de rester *connecté* en permanence. Les smartphones répondent à ce besoin de mobilité, de connectivité, et de simplicité. Bon nombre de communications entre collègues, amis ou entreprises passent désormais par ce biais. La sécurité semble avoir été mise de côté. Les attaques présentées dans cet article montrent clairement les faiblesses des smartphones. Toutes nos informations personnelles y sont centralisées.

L'idée qu'un ordinateur doit être immédiatement équipé d'un antivirus et d'un pare-feu est devenue un réflexe partagé par tous. Cet automatisme pour les ordinateurs l'avons-nous pour les téléphones mobiles ? L'utilisateur est persuadé qu'il est urgent de protéger son ordinateur fixe ou portable mais son simple téléphone qui lui semble inoffensif, il en négligera la sécurité.

Certaines tentatives de développement d'antivirus, et parefeu commencent à apparaître, mais l'utilisateur a pris l'habitude de ne dépenser que quelques euros pour des applications sur son téléphone. Sera - t- il prêt à investir davantage dans un antivirus pour un smartphone ?

Il paraît important d'inciter les entreprises à développer des H-IDS pour téléphone qui soient compatibles avec les fortes contraintes des utilisateurs qui souhaitent des mobiles réactifs, autonomes et fluides.

Références

1. Android-locdump.
`https://github.com/packetlss/android-locdump?locale=fr#readme`.
2. Android possèderait un mouchard.
`http://www.01net.com/editorial/531977/comme-liphone-android-possederait-un-mouchard/`.
3. Apple suit à la trace les utilisateurs d'iphone.
`http://www.ladepeche.fr/article/2011/04/20/1064188-Apple-suit-a-la-trace-les-utilisateurs-d/iPhone-et-iPad-selon-des-chercheurs.html`.
4. appWatchdog.
`http://viaforensics.com/appwatchdog/`.
5. Caffe Latte attack.
`http://www.airtightnetworks.com/home/resources/knowledge-center/caffe-latte.html`.
6. Faceniff.
`http://faceniff.ponury.net/`.
7. Firesheep Fallacies and Practical Advice.
`http://revolutionwifi.blogspot.com/2010/11/firesheep-fallacies-and-practical.html`.

2. L'emploi du mot 'consulter', sous entend également l'envoi des messages à la place de la victime.

8. iPhone Tracker.
http://petewarden.github.com/iPhoneTracker/.
9. Smartphone, Internet Mobile : la France dans le peloton de tête mondial selon Google.
http://www.businessmobile.fr/actualites/smartphone-internet-mobile-la-france/dans-le-peloton-de-tete-mondial-selon-google-39761718.htm.
10. Smartphone user study shows mobile movement under way.
http://googlemobileads.blogspot.com/2011/04/smartphone-user-study-shows-mobile.html.
11. Elie Bursztein Gustav Rydstedt, Baptiste Gourdin and Dan Boneh. Framing Attacks on Smart Phones and Dumb Routers : Tap-jacking and Geo-localization Attacks.
12. Andrew Hoog. Android forensics : Forensic acquisition and analysis, 2011.

État de l'art de la prise d'empreinte 802.11

Olivier Heen, Christoph Neumann, Stéphane Onno

Technicolor, Security & Content Protection Labs, Cesson-Sévigné, France
prenom.nom(@)technicolor.com

Résumé La prise d'empreinte d'un appareil réseau consiste en la mesure de son trafic afin d'identifier l'appareil lui-même ou certaines de ses caractéristiques. C'est une technique bien connue lorsqu'elle est appliquée à la reconnaissance des systèmes d'exploitation, avec des outils populaires comme NMAP, SinFP ou p0f (et des outils moins populaires comme Cron-OS ;). C'est une technique moins connue lorsqu'elle est appliquée en couche 2 pour la reconnaissance d'appareils 802.11. Pourtant, depuis 2005, environ trente papiers sont consacrés à ce sujet et plusieurs outils ont vu le jour comme BAFFLE [1] ou WiFinger [6].

Nous évoquons divers usages de la prise d'empreinte 802.11 et nous décrivons brièvement les principales techniques utilisées.

Usages

Le premier usage de la prise d'empreinte 802.11 est la reconnaissance de point d'accès. Il s'agit essentiellement d'éviter les points d'accès illégitimes (cf. *rogue access points*, *evil twin attack*). En effet un attaquant peut forger un faux point d'accès, semblable à un vrai, puis attendre que des stations légitimes se connectent et contrôler alors leur trafic réseau. Avec une méthode de prise d'empreinte, une station vérifie l'empreinte du point d'accès et agit en conséquence. Par exemple elle se déconnecte lorsque l'empreinte mesurée diffère trop de l'empreinte habituelle. Certaines techniques permettent même de vérifier l'empreinte d'un point d'accès avant de s'y connecter, évitant toute connexion malencontreuse.

La prise d'empreinte s'applique également aux stations. C'est alors le point d'accès qui vérifie que les empreintes des stations correspondent bien à des empreintes attendues. C'est utile sur des réseaux effectuant un contrôle d'accès par adresse MAC. En effet, un attaquant peut facilement forger une adresse MAC pour passer outre le contrôle d'accès, mais il lui sera plus difficile de forger une empreinte 802.11 valide.

Plus récemment, on trouve d'autres utilisations de la prise d'empreinte liées à la localisation des machines. Un ordinateur portable, par exemple, peut mesurer les empreintes 802.11 environnantes et en déduire sa localisation. Localisé dans le domicile de l'utilisateur, l'ordinateur portable omet de demander un mot de passe et devient donc plus facilement utilisable. Partout ailleurs, il demande systématiquement un mot de passe.

Lorsque des empreintes sont suffisamment stables, différenciées et nombreuses, elles peuvent même caractériser un utilisateur précis. Par exemple la combinaison de trois empreintes, un téléphone professionnel (Wi-Fi activé), un téléphone personnel (Wi-Fi activé) et un ordinateur portable, sert de signature pour un utilisateur nomade. Il est peu probable qu'un autre utilisateur ait systématiquement la même combinaison d'empreintes. On peut alors détecter cette combinaison et marquer quelques points de présence de l'utilisateur sur une carte, dans un but louable ou non. Ce cas deviendra préoccupant si le nombre d'appareils 802.11 mobile continue d'augmenter.

Enfin, imaginons deux appareils mobiles ayant pour un temps des déplacements similaires : deux appareils mobiles dans une même poche, la station 802.11p d'un véhicule et le téléphone mobile du conducteur, etc. Ces deux appareils utilisent les empreintes 802.11 environnantes et leurs variations au cours des déplacements comme une source d'information corrélée. Sur cette base ils décident d'un secret partagé, difficile à forger pour un attaquant n'ayant pas suivi le même trajet au même moment.

Test actif d'implémentation

Dans les méthodes dites actives, le détecteur envoie diverses trames 802.11 à la cible et analyse les réponses ou l'absence de réponse. Sur certains paramètres, les réponses varient significativement d'un matériel à un autre. Ces variations révèlent des libertés dans l'implémentation des spécifications, des différences dans les choix de valeurs par défaut, des extensions présentes ou non, etc.

Dès 2006, Gopinath et al. [5] observent selon les constructeurs des variations de paramètres tels que : les temps d'attente avant l'envoi de trames (*random back-off timers*), les valeurs du champ *duration*, l'utilisation de champs réservés, etc.

La même année, J. Cache [3] met en évidence des différences de comportement entre stations vis-à-vis du mécanisme de redirection d'association. Ce mécanisme permet à un point d'accès de rediriger dynamiquement des stations vers un autre point d'accès. Certains *chipset* obéissent à l'ordre de redirection, d'autres non. De plus, un ordre de redirection contient plusieurs paramètres comme l'adresse source et le BSSID. En faisant varier ces paramètres et en itérant les tests, J. Cache différencie plusieurs catégories de stations. À l'usage, cette méthode reste relativement "artisanale" avec beaucoup de faux-positifs et une précision limitée.

Plus récemment, Bratus et al. [2] présentent une méthode qui permet la prise d'empreinte active des points d'accès (et à moindre titre des stations). Il s'agit d'envoyer au point d'accès des requêtes utilisant des combinaisons de paramètres atypiques, inutiles ou non-définies. Les variations de plusieurs paramètres sont utilisées, parmi lesquels :

- Les bits `FromDS` e `ToDS`, sensés valoir zéro dans un message `Probe Request`.
- Les mêmes bits dans un message `Authorisation Request`
- Des bits de contrôle de trame, notamment
- L'absence d'élément attendus dans un message `Probe Request`, tels que le ESSID et l'information sur les taux de transfert.
- Des bits normalement à zéro dans une trame `Null Data` utilisée pour la gestion de consommation.

Le nombre de paramètres permet une combinatoire importante qui mène à une méthode de prise d'empreinte assez précise. Une implémentation est disponible sous le nom "Baffle" [1]. C'est actuellement la méthode active la plus précise pour les points d'accès. L'utilisation sur des stations est moins efficace.

Test passif d'implémentation

Dans les méthodes passives, le détecteur se contente d'analyser le trafic de la cible. Le détecteur n'envoyant aucune trame, les méthodes passives sont beaucoup plus discrètes que les méthodes actives. En revanche elles peuvent être plus lentes et imprévisibles puisqu'il faut attendre d'analyser des trames caractéristiques.

Franklin et al. [4] remarquent d'importantes variations dans l'algorithme d'envoi de trames `Probe Request`. Cet algorithme est souvent utilisé par les stations pour se connecter (en plus de l'algorithme par `beacon`). Sur les centaines de pages de la spécification 802.11 une seule page est consacrée à la description de cet algorithme pourtant non trivial. Les constantes critiques sont nommées mais aucune valeur de référence n'est indiquée. On retrouve donc naturellement d'importantes variations d'un constructeur à l'autre, comme le montre la figure 1. Cette méthode est particulièrement discriminante et facile à mettre en œuvre. En revanche elle est assez facile à contrer : Windows Seven notamment permet de désactiver en quelques clics l'envoi de trames `Probe Request`.

Mesure de dérive d'horloge

Il s'agit de méthodes passives un peu particulières dont la précision théorique va jusqu'à la machine elle même (et pas seulement le constructeur). En théorie on peut reconnaitre l'empreinte d'une machine précise parmi toutes les autres, sur différents réseaux.

L'idée de base est due à Kohno et al. [8] dans le contexte réseau (couche 3). Il remarque que toute machine possède une horloge basée sur un quartz. Chaque quartz étant physiquement différent, les horloges ont toutes une dérive différente. Ainsi, deux cartes 802.11 produites par le même constructeurs sur la même chaîne de montage auront quand même deux quartz différents : l'une dérivera par exemple de $+17\mu sec/sec$ l'autre dérivera de $-42\mu sec/sec$. Si les

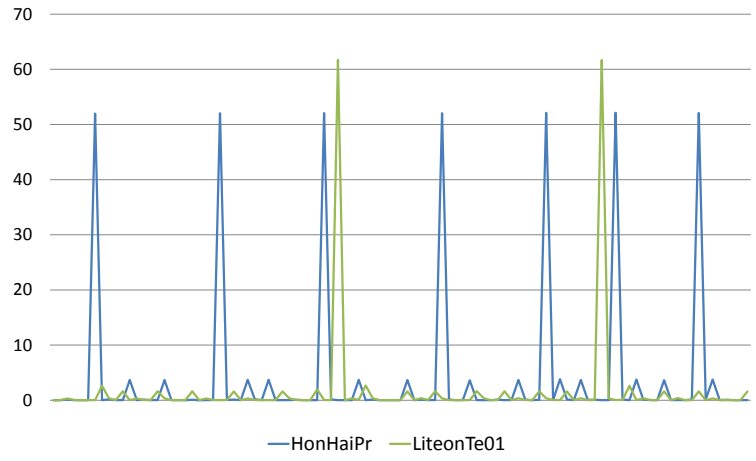


Figure 1. Délai en seconde entre deux `Probe Request` en fonction numéro de `Probe Request` de 1 à 100.

conditions de mesure sont correctes et si la durée de la mesure est suffisante, alors la dérive caractéristique peut-être calculée avec précision.

Jana et al. [7] mesurent la dérive d'horloge des points d'accès. La mesure est facilitée par la présence du champ *timestamp* dans chaque trame `Beacon` envoyée par le point d'accès. Il suffit de comparer la valeur de ce champ et l'heure de réception locale du détecteur. Quelques centaines de `Beacon` suffisent pour calculer une dérive stable. Les trames `Beacon` sont envoyées toutes les 0,1 seconde, cette valeur étant fixée par la norme. En pratique une mesure fiable est obtenue en quelques dizaines de seconde.

La mesure des dérives sur les stations est plus difficile car il n'existe pas d'équivalent du champ *timestamp*. Loh [9] propose une méthode alternative. Dans un premier temps il observer les trames `Probe Request` sur une longue période, une heure et plus. Ces trames sont regroupées par "rafales", c'est-à-dire par ensemble de trames normalement envoyées à intervalles fixes. En fait, ces intervalles ne sont pas tout à fait fixes : ils dérivent légèrement, en fonction directe de la dérive de l'horloge. La durée de la mesure restreint l'utilisation pratique de cette méthode à des cas particuliers : voyage, présence dans une salle de conférence, dans une salle de cours, etc.

Conclusion

La prise d'empreinte sur 802.11 est un domaine de recherche actif. Des papiers significatifs ont été publiés en 2010 et 2011. Les recherches actuelles portent sur trois axes :

- L'amélioration des méthodes de prise d'empreinte, en particulier leur précision et la diminution des faux positifs.
- La diversification des applications, notamment dans les réseaux domestiques.
- L'évasion et la déception pour, respectivement, ne pas laisser d'empreinte et forger de fausses empreintes.

Références

1. Sergey Bratus. *Baffle*. `http://baffle.cs.dartmouth.edu/`
2. Sergey Bratus, Cory Cornelius, David Kotz, and Daniel Peebles. Active behavioral fingerprinting of wireless devices. In *Proceedings of ACM WiSec'08*, March 2008.
3. Johnny Cache. *Fingerprinting 802.11 Devices*. Master Thesis, 2006.
4. Jason Franklin, Damon McCoy, Parisa Tabriz, Vicentiu Neagoe, Jamie Van Randwyk, and Douglas Sicker. Passive data link layer 802.11 wireless device driver fingerprinting. In *Proceedings Usenix Security 06*, August 2006.
5. K.N. Gaopinath, Pravin Bhagwat, and K. Gopinath. An empirical analysis of heterogeneity in ieee 802.11 mac protocol implementations and its implications. In *Proceedings of ACM WiNTECH'06*, September 2006.
6. Craig Heffner. *WiFinger*. `http://www.sourcesec.com/2009/05/09/wifinger-passive-wireless-fingerprinting-tool/`
7. Suman Jana and Sneha K. Kasera. On fast and accurate detection of unauthorized wireless access points using clock skews. In *Proceedings of ACM MobiCom 08*, September 2008.
8. Tadayoshi Kohno, Andre Broido, and K. C. Claffy. Remote physical device fingerprinting. *IEEE Trans. Dependable Secur. Comput.*, 2 :93–108, April 2005.
9. Desmond C. C. Loh, Chia Yuan Cho, Chung Pheng Tan, and Ri Seng Lee. Identifying unique devices through wireless fingerprinting. In *Proceedings of ACM WiSec'08*, March 2008.

De la radio matérielle à la radio logicielle : impact sur l'étude de la sécurité des réseaux sans fil

Chaouki Kasmi, Arnaud Ebalard et Pierre-Michel Ricordel

Agence Nationale de la Sécurité des Systèmes d'Information
51, Boulevard de la Tour-Maubourg
75700 Paris 07 SP
prenom.nom(@)ssi.gouv.fr

Résumé Pour des besoins de mobilité ou de coût, les systèmes d'information utilisent des moyens de communications "sans-fil" comme le Wi-Fi, le Zigbee, le Bluetooth, ou les réseaux de téléphonie cellulaires (DECT, 2G/3G). Ces réseaux sont aujourd'hui largement déployés pour l'échange de données privées ou professionnelles.

La transition progressive d'équipements radiofréquences dits matériels vers des technologies de radio logicielle ont permis de profiter du haut niveau d'intégration et de la flexibilité de ces technologies.

Ces équipements commercialisés à des prix parfois très faibles peuvent être détournés – logiciellement ou matériellement – de leurs fonctions premières, pour être ensuite utilisés comme des outils d'analyse. Ces derniers ont ainsi permis de réaliser des études protocolaires de nombreux réseaux "sans-fil" jusqu'alors difficiles ou très coûteuses à mettre en œuvre.

Mots-clés: Radio logicielle, SDR, USRP.

1 Introduction

La multiplication en nombre et en type des réseaux sans-fil a fortement fait évoluer les technologies et les méthodes utilisées pour leur développement. L'apparition de la radio logicielle a ainsi entrainé la réduction des temps et des coûts de développement des produits radios.

Ces évolutions ont également permis à la communauté scientifique de développer plus simplement des outils d'analyse efficaces, notamment pour évaluer leur sécurité.

Après une introduction aux réseaux sans-fil, le document s'attarde dans un second temps sur le concept de radio logicielle pour ensuite présenter son utilisation dans l'analyse de protocoles radio. Enfin, les évolutions envisagées dans le domaine font l'objet d'une section spécifique.

2 Introduction aux réseaux sans-fil

Il existe différents types de réseaux radios qui se distinguent par leurs caractéristiques physiques et protocolaires. Les premières dépendent grandement des

fonctionnalités attendues du réseau radio (débit et portée par exemple), des capacités physiques des émetteurs et terminaux (puissance, consommation) mais également de la date de conception de la technologie. Les caractéristiques protocolaires sont fortement liées à la complexité des services fournis par le réseau radio.

2.1 Aspects radios

Les aspects radios dimensionnants au niveau des couches physiques (PHY) et de contrôle d'accès au media (MAC) sont :

- ***la fréquence de travail***, qui impacte la portée, les antennes et les premiers étages analogiques ;
- ***les sauts de fréquence*** éventuels d'un canal à un autre. Certains standards, comme le Bluetooth, nécessitent une très grande agilité en fréquence, qui impacte les modules de traitement du signal et notamment les fonctions de conversion analogique/numérique.
- ***la bande passante*** de travail, qui impacte les performances du système. Plus la bande passante est élevée, plus la capacité de traitement de la radio doit être élevée, et plus le débit binaire utile est élevé ;
- ***le parallèlisme des communications*** : simplex (un seul sens de communication), alternat (half-duplex), bidirectionnel simultané (full-duplex), voire MIMO [1], qui impacte l'architecture de la radio ;
- ***la modulation et le codage employés***, qui peuvent être simples (OOK [2], FSK [3]) ou très complexes à mettre en oeuvre (OFDM [4], DSSS [5], modulation d'amplitude en quadrature haute densité, codage en treillis) ;
- ***les codes détecteurs et correcteurs d'erreurs***, qui peuvent aller du simple CRC [6] aux codes les plus complexes (Reed-Solomon, Turbo Codes) ;
- ***la méthode d'accès au media (MAC)*** peut nécessiter des temps de réponse très courts (synchronisation avec les autres émetteurs pour éviter les collisions, acquittements des paquets reçus) ou des adaptations spécifiques des étages de décodage (détection de collision). Ceci accroît les besoins de traitement temps réel ;

En pratique, la complexité à interagir au niveau physique avec un type de réseau radio donné dépend de la capacité du matériel à supporter ces différentes

1. Multiple Input and Multiple Output, pour réception et émission multiple
2. On-Off Keying
3. Frequency Shift Keying
4. Orthogonal Frequency-Division Multiplexing
5. Direct-Sequence Spread Sprectrum
6. Cyclic Redundancy Check

caractéristiques. Mais celle-ci dépend également de la capacité à reprogrammer ou contrôler le matériel à une vitesse compatible avec la fréquence de fonctionnement du réseau.

2.2 Aspects protocolaire

Certains réseaux radios simples transportent des données brutes, parfois dans un seul sens de communication. D'autres, plus complexes, utilisent des mécanismes de signalisation pour supporter diverses fonctionnalités : partage du spectre radio entre différentes ressources, négociation de fonctionnalités ou de paramètres de sécurité comme l'authentification ou le chiffrement du canal de communication.

Les réseaux de téléphonie mobile intègrent ainsi un grand nombre de fonctionnalités nécessitant le maintien d'états au niveau des récepteurs et des émetteurs. Ils supportent au final le transport d'information de voix, de données et de signalisation.

La maîtrise des aspects protocolaires des réseaux radios étudiés et la capacité à contrôler des éléments logiciels et matériels de manière à interagir avec ceux-ci sont des éléments clés dans leur étude. La complexité des protocoles et l'accès aux documents de spécification sont bien évidemment des aspects dimensionnants dans leur étude.

2.3 Analyse

Les interactions avec un réseau radio dépendent donc initialement de la complexité à reproduire sa couche physique soit par un développement de matériel spécifique soit en détournant/réutilisant un matériel dédié. La difficulté est ici inhérente aux éléments physiques cités précédemment : fréquence, largeur de bande, modulation et utilisation potentielle de saut de fréquence ...

De plus, en fonction du scénario envisagé, la capacité à émuler les couches physiques et donc à réaliser les premières interactions avec le réseau radio n'est pas l'unique obstacle à considérer. Ainsi, la compréhension de la couche protocolaire transportée est généralement nécessaire dans de nombreux scénarios ; une trame donnée n'ayant par exemple une signification qu'à un instant donné, pour une ressource radio donnée et en fonction des échanges précédents. Pour illustrer cet exemple, l'émission d'une trame de désauthentification sur un réseau Wi-Fi nécessite une authentification préalable d'un client au point d'accès et la présence des identifiants spécifiques à ce client dans la trame de désauthentification construite.

3 Présentation de la radio logicielle

Issue de la recherche militaire américaine à la fin des années 70 sur les radios multimodes opérant en bandes VHF et UHF, la radio logicielle aboutit au début des années 90 par la famille de projets SPEAKeasy Phase I et Phase II aux premières implémentations de radios logicielles émulant des radios tactiques. Elle rejoint en 1991 [1] la communauté scientifique dans le cadre des applications de télécommunications civiles. En 1995, un RFI [7] sur la radio logicielle pour des applications de téléphonie mobile marque le point de départ d'une activité accrue dans ce domaine. En 1996 apparait le Modular Multifunctional Information Transfer System Forum qui devient en 2009 le Wireless Innovation Forum [8]. Il s'agit d'une coopération entre chercheurs et industriels internationaux dont l'objectif est de promouvoir et de développer la technologie radio logicielle.

3.1 Le principe de radio logicielle

Une radio logicielle – en anglais Software Defined Radio ou SDR – est un système de radiocommunication configurable utilisant des techniques de traitement numérique du signal sur des circuits numériques programmables. Sa flexibilité lui permet de s'adapter à différents protocoles de radiocommunication, et de répondre au besoin croissant de performance et d'interopérabilité entre systèmes. L'objectif de la radio logicielle consiste en une dématérialisation complète de l'interface radio. Elle participe à la tendance globale des circuits électroniques à devenir des circuits à haute densité d'intégration [36].

L'évolution ultime de la radio logicielle est la radio intelligente ou radio cognitive [1,2]. Une radio intelligente est une radio logicielle dans laquelle les éléments de communication évoluent en fonction des conditions de propagation et de son état interne, ce qui se traduit par une modification de sa couche physique : utilisation de différentes modulations, différents types de codes correcteurs pour répondre aux stress du canal de propagation.

Les radios logicielles permettent l'utilisation de multiples formes d'ondes, éventuellement dans différentes bandes spectrales, pour différents usages, voire même de façon simultanée. Dans une radio logicielle, les propriétés de la fréquence porteuse, de la bande passante du signal, de la modulation et de l'accès au réseau sont définies par logiciel. Celles-ci ont donc vocation à être portables sur tout plate-forme ; ceci explique la nécessité des recherches actuelles dans le développement de standards ouverts [22].

Les radios logicielles modernes mettent également en œuvre des codes de correction d'erreurs, des modules d'encodage de la voix, de la vidéo ou des données

7. Request For Information
8. `http://www.wirelessinnovation.org/`

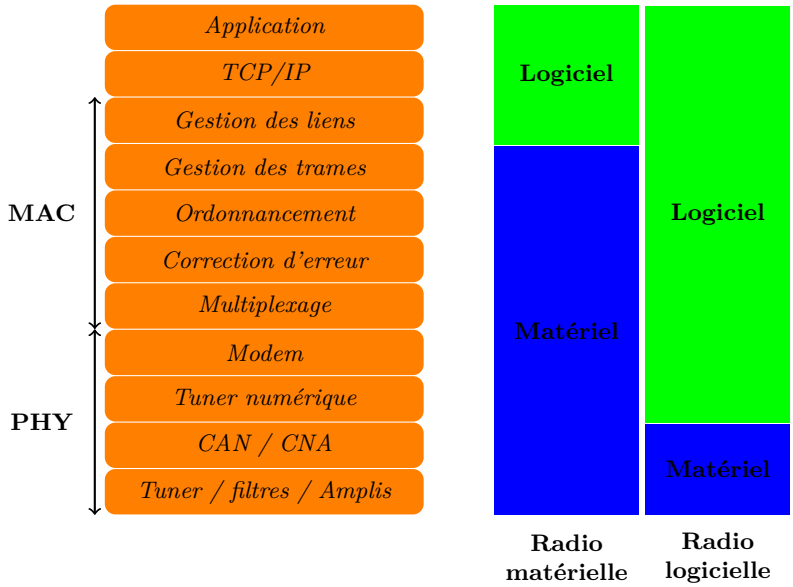


Figure 1. Évolution de la radio matérielle à la radio logicielle

et des fonctions cryptographiques. Ces équipements sont aujourd'hui éprouvés [8] et font partie de notre quotidien, que ce soit pour la conception de modem 3G+ [34,35] dans les terminaux mobiles ou les équipements des réseaux radios comme les stations de base.

3.2 Architectures de la radio logicielle

Il existe différentes architectures intermédiaires de la radio dite matérielle à la radio intelligente. Le niveau d'intégration des fonctions numériques résulte de l'augmentation des performances des composants utilisés pour ces traitements [44]. Du DSP au FPGA [27] ou par l'utilisation de cartes multi-processeurs, la capacité de traitement temps réel a fortement progressé. Il est ainsi possible d'ajouter à l'équipement les capacités liées à la prise de décisions sur des modifications des paramètres de la couche physique en fonction de capteurs externes (bruit électromagnétique, surcharge de la bande de fréquence utilisée...).

Ces différentes évolutions [37] ainsi que la multiplication des capacités des terminaux se traduisent par une classification de ces systèmes :

Catégorie	Dénomination	Degré de reconfiguration
0	Radio matérielle	Radio qui ne peut pas être modifiée par logiciel ; reconfiguration par échange de composant.
I	Radio contrôlée par logiciel	Reconfiguration limitée à un jeu prédéfini de paramètres (niveau de puissance, interconnexions, ...)
II	Radio définie par logiciel	Contrôle logiciel et reconfiguration des formes d'ondes, fréquence, bande passante, (dé)modulation, détection du signal, paramètres de sécurité, etc.
III	Radio logicielle idéale	Conversion analogique au niveau de l'antenne, du haut-parleur et du microphone ; tout le reste est logiciel.
IV	Radio logicielle ultime	Comprend tout type de trafic et d'informations de contrôle, et supporte la plupart des applications et circuits d'antenne.

Figure 2. Classification des systèmes radio logicielle selon le Wireless Innovation Forum

Les plates-formes et bancs d'essai de radios logicielles "libres" offrent aux chercheurs et développeurs la possibilité de concevoir leurs propres applications radios logicielles. Au vu de la complexité croissante des systèmes à concevoir, une plateforme de prototypage est choisie en fonction de critères multiples : flexibilité, rapidité de calcul, communications entre composants et présence d'interfaces externes. Bien que la radio logicielle restreinte offre de multiples avantages aux concepteurs de systèmes radio, il reste de nombreuses questions ouvertes sur la façon de mettre en œuvre et de gérer la flexibilité dans un système de transmission sans fil.

Dans le cadre de projets de recherche, plusieurs plates-formes radios logicielles expérimentales ont pu voir le jour. La liste des radios logicielles réalisées, en développement ou à l'état de prototype [19] est très étendue.

3.3 Exemples de plates-formes spécifiques

On distingue différentes catégories de dispositifs à base de SDR. Les dispositifs spécifiques, conçus et développés pour répondre à un besoin particulier, relativement peu flexibles, sont souvent coûteux ; cependant ces systèmes sont optimaux pour l'application visée. On retrouve notamment dans cette catégorie les équipements de test et de métrologie ainsi que de nombreuses plates-formes de recherche, mais aussi certaines fabrications artisanales dédiées à une norme de radiocommunication ou un protocole particulier. Les applications visées par ces plates-formes sont très diverses :

- BEE2, Berkeley Emulation Engine 2 est une plate-forme de radio logicielle utilisée pour des applications nécessitant des capacités de calcul importantes, notamment en radioastronomie [7,9,10].
- CalRadio est un programme de recherche et de développement d'émetteur/-récepteur sans-fil mettant en œuvre des plates-formes de test [11].
- Chameleonic Radio est un prototype expérimental de radio multibande et multimode, élaboré par Virginia Tech [12].
- VT-CORNET, Virginia Tech Cognitive Radio Network Testbed est une plate-forme de test de radio intelligente [13].
- FPGA4U, FPGA4U est une carte développée par l'école polytechnique fédérale de Lausanne (EPFL) à des fins pédagogiques [14].
- HPSDR, HPSDR (High Performance Software Defined Radio) est un projet matériel et logiciel "libre" de récepteur SDR [15].
- KNOWS, Kognitiv Networking Over White Spaces issu de l'alliance de Microsoft et de Dell est une radio intelligente détectant et exploitant de façon adaptative les bandes libres TV [16].
- KUAR, la plate-forme Kansas University Agile Radio est une plate-forme expérimentale "low-cost" [17].
- MARS, le système Maynooth Adaptable Radio System, finalisé en 2007, est une SDR où toutes les opérations de traitement du signal sont implémentées sur des processeurs à usage généraux [18].
- Le Japanese National Institute of Information and Communications Technology (NICT) a réalisé une plate-forme SDR pour tester des réseaux mobiles de dernière génération [19].
- SDR4ALL, Software Defined Radio For All, projet de recherche né de la collaboration entre le CEA et l'école d'ingénieurs SUPELEC, consiste en la mise en œuvre d'outils pour tester, en conditions réelles, des algorithmes et schémas de transmission radio [20].
- WARP, Wireless Open Access Research Platform est une plate-forme de radio logicielle communautaire évolutive, extensible et programmable, à vocation académique et de recherche, développée par l'université RICE de Houston (Texas, Etats-Unis) [45].

En ce qui concerne les modules logiciels, plusieurs standards ont été spécifiés : Ham Radio Control Libraries [23], Software Communications Architecture [21,22], Object Management Group Model-Driven Architecture [24] IEEE Standards Coordinating Committee et la GNU Radio [39].

3.4 Sécurité et sûreté de fonctionnement

Des travaux portant sur la fiabilité et la sécurité de ces équipements ont été réalisés. Le Wireless Innovation Forum a élaboré un ensemble d'exigences de sécurité [25] visant à prévenir un comportement indésirable d'une radio logicielle

dans le cas d'une action malveillante. La cause de ces comportements anormaux peut être un virus, un vers ou tout autre code malveillant, qui provoquerait la génération d'interférences, des risques sur la santé des personnes (rayonnement électromagnétique), la divulgation (de secrets industriels ou informations classifiées) ou des fraudes en tout genre sur des réseaux à accès payant.

3.5 Des approches matérielles pures et SDR

La réalisation de plates-formes à faible coût est maintenant possible et nombreuses sont les réalisations de SDR radioamateur. Ces équipements offrent la possibilité au plus grand nombre d'étudier le spectre radioélectrique, et notamment les protocoles de transmission radiofréquence. Divers projets universitaires emploient ces radios logicielles à des fins d'étude et de recherche dans des bandes de fréquences autorisées (bandes ISM [9] par exemple), mais aussi l'utilisation de ces équipements sur d'autres parties du spectre, pour par exemple étudier des réseaux sans fil (GSM, GPS, DECT ...).

L'émergence des radios logicielles au sein des équipements de radiocommunication apporte de la souplesse de fonctionnement, une grande évolutivité et permet de réaliser des économies d'échelle aux industriels du secteur. Cependant, la migration logicielle des fonctions de traitement du signal radio a aussi des inconvénients. En effet, de nombreuses normes de radiocommunication sont soumises à licence et/ou restrictions d'emploi. Les équipements conventionnels sont censés respecter ces contraintes de fonctionnement.

4 Analyse de protocoles radios

4.1 Capacité d'analyse d'un modem reconfigurable : le dongle

Les limitations décrites précédemment [10], associées à l'utilisation d'un équipement matériel dédié, notamment en terme de complexité de développement, ont poussé de nombreux chercheurs à utiliser certains matériels dédiés mettant en oeuvre un logiciel reconfigurable.

Ces matériels prennent souvent la forme d'un *dongle* utilisant différents types d'interface de communication avec la machine hôte (USB, PCI, PCIe, ...).

La généralisation de l'utilisation de ce type d'équipement remonte aux débuts de l'étude pratique des réseaux Wi-Fi à la fin des années 90. A cette époque, certaines cartes PCMCIA mettant en oeuvre des chipsets particuliers permettaient le scan, l'écoute voire l'injection de trames sur les réseaux wifi. Celles-ci ont permis la mise en oeuvre des premières preuves de concepts (PoC) visant à

9. *Industrial, Scientific and Medical*, bandes radio fréquences réservées pour utilisation par les applications industrielles, médicales et scientifiques.

10. protocole à saut de fréquence, bande passante importante, ...

démontrer les nombreuses attaques sur la première version du protocole 802.11 (désauthentification, attaque sur le WEP ...).

Ces études ont été rendues possibles car les couches basses des matériels n'étaient pas figées mais reconfigurables logiciellement. Par la suite, de nombreux autres protocoles ont été étudiés par l'intermédiaire de matériels dédiés utilisant un logiciel reconfigurable :

- ***Bluetooth***
  L'explosion des possibilités offertes par le protocole Bluetooth pour les communications à faible distance a engendré la production en masse de chipsets intégrés directement aux équipements terminaux (PC, téléphones, terminaux de paiement, ...) mais également de *dongles* (généralement USB).
  La grande majorité de ces équipements fournissent le support matériel des couches basses du protocole, nécessité par l'utilisation d'une technique de saut de fréquence assez poussée (jusqu'à 1600 sauts par seconde) et une largeur de spectre importante (79 canaux de 1 Mhz répartis de 2.402 Ghz à 2.480 Ghz). En pratique, ces spécificités du Bluetooth rendent difficile le développement d'un support sur une plate-forme de radio logicielle comparativement aux gains obtenus par rapport à l'utilisation d'un *dongle*.

- ***DECT***
  Créé en 1992, le protocole DECT est aujourd'hui utilisé par plusieurs centaines de millions de périphériques, parmi lesquels principalement des téléphones.
  L'analyse du protocole par la communauté a débuté vers 2006 et a rapidement mené à plusieurs résultats, mettant notamment au jour des faiblesses dans les primitives cryptographiques et les implémentations [3,4].
  Même si les travaux initiaux d'analyse cryptographique du protocole ont nécessité une rétroingéniérie matérielle et logicielle, la démonstration des possibilités d'écoute et d'émission sur des réseaux DECT a été simplifiée par l'utilisation d'une simple carte PCMCIA DECT de type ComOnAir.

- ***Zigbee***
  Basé sur le protocole IEEE 802.15.4, les avantages en matière de simplicité protocolaire, de consommation électrique et de coût de Zigbee en ont fait un protocole de choix pour le contrôle sans-fil dans le milieu industriel et la gestion de bâtiment : climatisation, vannes, alarmes, serrures.
  Des *dongles* voire des kits de développement complets sont disponibles pour des sommes dérisoires. Des projets comme KillerBee [11] permettent de transformer certains de ces *dongles* pour capturer et émettre des trames à

11. `http://code.google.com/p/killerbee/`

bas niveau.

- ***NFC***
  L'apparition de la technologie NFC, notamment sur les téléphones portables, pour des utilisations comme le paiement sans contact ou la récupération de contenu (URL, coupons de réductions, ...) a poussé le développement de projets [6,5] visant à supporter les principaux matériels. Cette technologie ouvre des portes vers l'émulation native de cartes sans contact, ce qui était auparavant difficile à mettre au point matériellement.

4.2 Capacité d'analyse d'une radio logicielle type USRP

Les performances des équipements de type radio logicielle et l'ouverture de ces plates formes dans des projets dits "libres" ont permis aux chercheurs du domaine d'analyser les signaux électromagnétiques qui nécessitaient auparavant des équipements coûteux. Les problèmes d'ordre matériel sont devenus des problèmes logiciels bien moins difficiles à résoudre car les processus de conception, de mesure et de test sont associés à une mise à jour logicielle au lieu d'une conception matérielle.

Les projets "libres" USRP [46] et GNURadio [39] sont des exemples de l'essor de l'utilisation de la radio logicielle. Ces projets ont permis aux spécialistes de la sécurité de s'intéresser aux protocoles radios.

- ***Systèmes champs proche : RFID et NFC***

  En 2006, Henryk Plötz a utilisé un USRP [46], le projet GNU Radio [39] et un outil de visualisation pour capturer et analyser le signal d'identification d'une carte RFID pour déclencher l'ouverture d'une porte dotée d'un système de contrôle d'accès 125 kHz. Ces travaux ont été suivis par l'analyse des technologies à 13.56 MHz ayant abouti aux attaques sur Mifare Classic [47].

- ***Réseaux de données : Bluetooth, ZigBee et Wi-Fi***

  Les protocoles Bluetooth, ZigBee et Wi-Fi ont fait l'objet d'une analyse de leurs mécanismes de sécurité. L'acquistion d'un transfert de données Bluetooth à l'aide d'une radio logicielle de type USRP n'est pas chose aisée du fait de certaines particularités de ce protocole. L'évasion en fréquence impose à l'équipement de mesure une agilité en fréquence ou une bande d'acquisition importante. Ces deux approches consistent au choix :

- à acquérir tous les canaux Bluetooth en parallèle ce qui nécessite un équipement à bande-passante importante et capable de numériser un grand nombre de données ;
- à reconfigurer la fréquence de l'USRP à chaque saut de fréquence.

Celles-ci ne sont en pratique ni satisfaisantes ni suffisantes.

- ***Réseaux cellulaires : DECT et GSM***

  Les réseaux cellulaires ont également fait l'objet d'études ayant permis l'analyse de trames de réseau de téléphonie mobile existants [42]. L'implémentation de la pile protocolaire est d'ailleurs disponible sur Internet depuis 2004. L'analyse du GSM à l'aide d'une radio logicielle permet l'étude et l'analyse des mécanismes cryptographiques présents dans les couches physiques. Sans implémentation des couches bas niveaux, il est impossible d'étudier en pratique la sécurité de ces réseaux.
  En ce qui concerne le DECT un projet d'implémentation a échoué en raison des difficultés rencontrés [41]. Les ressources matérielles pour le traitement des données des couches 3 et 4 du protocole étaient trop importantes pour être gérées pas l'USRP. Les chercheurs se sont alors intéressés à l'utilisation d'un *dongle* dédié [12].
  L'implémentation SDR de ces différents protocoles n'a donc pas toujours abouti. Les difficultés rencontrés ainsi que l'attrait du protocole pour la communauté de recherche sont des éléments déterminants pour la réussite du projet.
  Des travaux sont en cours dans la communauté sur les réseau PMR-TETRA [43] et les réseaux de téléphonie satellite [42].

4.3 Comparaison des approches SDR et *dongle*

L'analyse de réseaux radio est utile afin de vérifier la conformité de l'implémentation vis-à-vis des spécifications techniques. Les travaux réalisés par différents chercheurs en sécurité ont, avec le temps, démontré le besoin d'auditer les différentes couches de ces protocoles. L'implémentation des mécanismes de sécurité dans certains protocoles comme le GSM ont poussé les chercheurs à s'intéresser à de nouvelles méthodes d'analyse. La radio logicielle est rapidement devenue le moyen efficace pour la compréhension du protocole et des échanges de données entre les différents équipements. Cependant l'audit de ces couches de sécurité nécessite la réimplémentation du protocole dans son intégralité.

12. carte PCMCIA ComOnAir citées précédemment

La radio logicielle offre les possibilités suivantes lors de l'étude de réseaux sans-fil :

- analyse en disponibilité du canal radio ;
- analyse protocolaire ;
- analyse des données sur l'interface air.

Néanmoins, ces possibilités sont à modérer en fonction :

- de la capacité et les performances des équipements utilisés ;
- de la robustesse du protocole ;
- de la disponibilité des spécifications techniques ;
- des mécanismes de sécurité mis en œuvre.

A contrario l'utilisation de *dongles* ayant une architecture optimisée pour un protocole donné permet une étude simplifiée, notamment dans les cas suivants :

- le saut en fréquence est utilisé par le protocole étudié (Bluetooth) ;
- la largeur spectrale est importante (Wi-Fi) ;
- la spécification du protocole est indisponible (DECT) ;
- les mécanismes de sécurité sont mis en oeuvre dans les couches hautes du protocole (Wi-Fi).

5 Evolutions envisagées

L'intégration des nouvelles technologies de transmission radiofréquence dans une plate-forme SDR nécessite une augmentation des performances des interfaces de communication, de la puissance des processeurs et des capacités de traitement du signal par les composants embarqués. L'USRP2 (FPGA de nouvelle génération, interface Gigabit Ethernet) est un premier pas vers des SDR de nouvelle génération. Voici quelques axes d'amélioration possibles des SDR actuelles :

- ***Le défi de la numérisation haut débit***
  Les défis de la numérisation sont aujourd'hui en passe d'être relevés. Cependant, le traitement de ces données est encore lié à l'utilisation de processeurs multi-coeur. Pour contourner cette contrainte, des travaux de conception portant sur un composant analogique ont été réalisé ; le SASP [13] est inséré entre l'amplificateur faible bruit et le CAN [14]. Il a pour

13. Sampled Analog Signal Processor
14. Convertisseur Analogique Numérique

rôle d'effectuer un prétraitement analogique du signal RF afin d'abaisser la fréquence de travail du CAN à 10 MHz.

- ***Amélioration des performances des interfaces entre la plate-forme et l'ordinateur hôte***
  L'utilisation de cartes multi-processeur impose au contrôleur principal de pouvoir traiter un grand nombre de données. Afin de transmettre ces données du module radio vers le module principal (utilisation d'un PC hôte pour le module protocolaire) la liaison de donnée doit être la plus efficace possible :
  - Une liaison PCI Express version 2.0 est optimisée pour le transfert de données en streaming et offre un débit utile de 1 Go/s par sens de transmission. La combinaison de plusieurs voies PCI Express pourrait augmenter significativement le débit global. La plupart des ordinateurs disposent d'au moins deux voies par port d'extension. Un accès au bus graphique offre une disponibilité de 32 voies, soit 16 Go/s de données bidirectionnelles en PCIe 32x 2.0. Les spécifications de la version 3.0 de PCI Express prévue pour une commercialisation en courant 2011 prévoient de doubler ces débits ;
  - Une liaison Gigabit Ethernet dernière génération offre jusqu'à 100 Gigabit/s de débit, ce qui est également suffisant pour la plupart des applications. On notera l'évolution de la radio logicielle USRP sur ces aspects permettant le passage d'une bande passante de 8 MHz à 25 MHz ;
  - Une liaison USB 3.0 permettrait par ailleurs d'augmenter les débits de communications ; ce standard n'est cependant pas encore apparu sur des systèmes temps-réel embarqués.

- ***Augmentation des capacités de traitement embarqués***
  Le concept d'intégration de la majorité des traitements in situ est motivé par le fait que les schémas de communication nécessitent des temps de réponse de plus en plus courts. La réalisation d'opérations de traitement du signal par des circuits embarqués spécialisés type DSP et FPGA de plus en plus performants, permet d'envisager la mise en œuvre d'applications temps réel pour des réseaux de communications haut-débits.

- ***Divers types de processeurs au sein d'une même plate-forme***
  Une implémentation typique de SDR contient un processeur général, un DSP ou un FPGA, voire un FPGA embarquant des cœurs DSP. Les systèmes à base de FPGA offrent des performances intéressantes mais au prix d'une augmentation de la complexité de conception. Les processeurs à usage généraux dits GPP sont moins efficaces pour le traitement de la couche phy-

sique, mais excellents pour les couches supérieures et paraissent plus accessibles aux développeurs. Aucune solution n'est optimale, c'est pourquoi les processeurs dédiés aux applications de radio logicielle possèdent généralement une architecture hétérogène multicœur composée de FPGA(s), DSP(s) ou GPP(s). L'exploration des capacités des puces graphiques GPU pour des traitements parallèles est en cours ; ces processeurs ayant démontré leurs capacités pour des calculs distribués.

L'apparition de nouveaux standards de communication pour la gestion de transfert de données entre le contrôleur général et les modules de traitement ainsi que les hausses en performance de composants dédiés permettra une amélioration non négligeable des équipements radio logicielle.

En ce qui concerne les projets "libres", plusieurs évolutions prouvent l'impact signicatif sur les techniques d'acquistion et de traitements des échantillons. La numérisation large bande ne peut à elle seule améliorer les débits d'acquisition car le traitement des données est bridé par les capacités des processeurs dédiés ainsi que par le contrôleur principal devant absorber des masses importantes de données.

6 Conclusion

L'utilisation de radio logicielle dans les systèmes de radiocommunication est en plein essor. Cette technologie possède des avantages économiques significatifs : elle permet de réaliser rapidement et à moindre coût les développements nécessaires au support des nombreux protocoles existants ainsi que de ceux en cours de spécification.

L'amélioration des architectures matérielles dédiées aux applications de radio logicielle ainsi que la mise en place de standards internationaux promet à cette technologie un vaste champ d'utilisation (outils de métrologie, équipement de télécommunications, systèmes tactiques ...).

Les mécansimes de sécurité spécifiés aux niveaux des différentes couches des modèles protocolaires imposent aux chercheurs en sécurité d'adapter les moyens d'analyse aux protocoles testés. L'apparition de périphériques reprogrammables ainsi que la radio logicielle deviennent ainsi des solutions complémentaires pour l'analyse de ces modèles.

La pertinence de l'un ou de l'autre de ces outils d'analyse dépend en pratique des spécificités du protocole considéré, aussi bien au niveau des couches physiques que protocolaires. Une des contraintes résiduelle forte sur l'utilisation de la radio logicielle pour évaluer la robustesse de certains protocoles réside ainsi dans la disponibilité de spécifications techniques.

La disponibilité pour la communauté scientifique et le grand public de ces outils d'analyse a permis de mettre en évidence les limites de certains protocoles

et d'améliorer la sécurité de leurs successeurs. Malgré tout, le détournement à des fins malveillante [15] de ces outils d'analyse permet d'envisager la récupération par un attaquant de données personnelles.

Les évolutions à venir de la radio logicielle prenant en compte les limitations des plates-formes actuelles en feront certainement un outil de plus en plus incontournable dans l'analyse des réseaux sans-fil.

Références

1. J. Mitola. **"Software Radio Architecture : Object-Oriented Approaches to Wireless Systems Engineering"**. Editions Wiley, 2000. 543 pages. ISBN 0471384925.
2. J. Mitola. **"Cognitive Radio : An Integrated Agent Architecture for Software Defined Radio"**. Ph.D. dissertation, Royal Institute of Technology (KTH), Sweden, 2000, Disponible à `http://web.it.kth.se/~maguire/jmitola/Mitola_Dissertation8_Integrated.pdf`
3. Andreas Schuler, Erik Tews, Ralf-Philipp Weinmann, **"deDECTed.org"**, 29 Décembre 2008, Disponible à `https://dedected.org/trac/raw-attachment/wiki/25C3/talk-25c3.pdf`
4. Karsten Nohl, Andreas Schuler, Erik Tews, Ralph-Philipp Weinmann, Mathias Wenzel **"DECT (part II)"**, 29 Décembre 2009. Disponible à `https://dedected.org/trac/raw-attachment/blog/slides-for-the-26c3-talk/DECT\%20\%28Part\%20II\%29.pdf`
5. Projet de boite à outils NFC basé sur libnfc, Disponible à `http://code.google.com/p/nfc-tools/`
6. Biliothèque fournissant le support pour l'interaction avec des périphériques NFC. Disponible à `http://code.google.com/p/nfc-tools/`
7. **"Berkeley Emulation Engine 2"**, Disponible à `http://bee2.eecs.berkeley.edu/`
8. Dr James E. Gunn, **SDR Market studies Overview**
9. C. Chang, J. Wawrzynek, et R. Brodersen. **"BEE2 : a high-end reconfigurable computing system. IEEE Design & Test of Computers**, IEEE, vol. 22, no. 2, pages 114 à 125, mars-avril 2005. Disponible à `http://bee2.eecs.berkeley.edu/papers/BEE2_chang_ieee.pdf`
10. S. Mishra, D. Cabric, C. Chang, D. Willkomm, B. Van Schewick, S. Wolisz et B. Brodersen. **"A real time cognitive radio testbed for physical and link layer experiments"**. IEEE International Symposium Dynamic Spectrum Access Networks (DySPAN), novembre 2005, [en ligne]. Disponible à `http://www.tkn.tu-berlin.de/publications/papers/dyspan05_cr-testbed2.pdf`
11. **"Calit2 Wireless Communications Research and Development Platforms"** Disponible à `http://calradio.calit2.net/`
12. **Gumstix packs**. Disponible à `http://www.gumstix.com/store/catalog/packs.php`
13. **"Cognitive Radios and Networks"**. Bradley Department of electrical & Computer Engineering. Virginia Tech, Disponible à `http://wireless.vt.edu/research/Cognitive_Radios_Networks/`
14. **USB-powered FPGA-based development board**. School of Computer and Communication Sciences of EPF. Disponible à `http://fpga4u.epfl.ch/wiki/Main_Page`
15. **HPSDR Wiki : Community Portal**. Disponible à `http://openhpsdr.org/wiki/index.php?title=HPSDRwiki:Community_Portal`
16. P. Bahl, R. Chandra, T. Moscibroda, R. Murty, et M. Welsh. **White space networking with Wi-Fi like connectivity**. SIGCOMM '09, New York, États-Unis 2009, pages 27 à 38. Disponible à `http://www.eecs.harvard.edu/~mdw/papers/whitefi-sigcomm09.pdf`

15. En France, ces systèmes sont soumis à règlementation dès lors qu'ils deviennent des outils dits d'interception [48]

17. G. J. Minden, J. B. Evans, L. Searl, D. Depardo, V. R. Petty, R. Rajbanshi, T. Newman, Q. Chen, F. Weidling, J. Guffey, D. Datla, B. Barker, M. Peck, B. Cordill, A. M. Wyglinski et A. Agah. **"KUAR : A Flexible Software-Defined Radio Development Platform"**. Information Technology and Telecommunications Center. Université du Kansas. 2007. Disponible à `http://www.ittc.ku.edu/publications/documents/minden2007_dyspan07.pdf`
18. R. Farrell, M. Sanchez et G. Corley. **"Software-Defined Radio Demonstrators : An Example and Future Trends"**. 30 septembre 2008. Disponible à `http://www.hindawi.com/journals/ijdmb/2009/547650.html`
19. H. Harada. **"Software defined radio prototype for multi-mode and multi-service radio communication systems"**. National Institute of Information and Communications Technology (NICT). Disponible à `http://www.sdrforum.org/pages/sdr05/4.6\%20Special\%20Applications\%202/4.6-04\%20Harada.pdf`
20. **Software Defined Radio For All**. Disponible à `http://sdr4all.org/index.html`
21. **SCA : Support for "Three Category" Approach for Software Communications Architecture (SCA) Standards**
22. **SCA : SCA Implementation Opensource FM3TR Reference Waveform – Developed by Mercury Computer Systems under contract to the Wireless Innovation Forum, this reference implementation provides an open source implementation of an SCA enabled test waveform Link opens Mercury website**
23. **Ham Radio Control Libraries**. Disponible à `http://hamlib.sourceforge.net/`
24. **Object Management Group Model-Driven Architecture**. Disponible à `http://www.omg.org/mda`
25. **Security of SDR study**. Disponible à `http://srg.cs.uiuc.edu/swradio/`
26. D. Spill et A. Bittau. BlueSniff : Eve meets Alice and Bluetooth. First USENIX Conference on Offensive Technologies (WOOT'07), **USENIX, 6 aout 2007**
27. M. Cummings and S. Haruyama., **"FPGA in software radio"**, IEEE Comms. Mag., pp. 108–112, 2 1999.
28. S. Gultchev, K. Moessner, and R. Tafazoli, **"Management and control of reconfiguration procedures in software radio terminals"** in Proc. 2nd Workshop on Software Radios, Karlsruhe, Germany, March 2002, pp. 125–129.
29. R. Hoshyar, S. Gultchev, K. Seo, and R. Tafazolli, **"Software reconfigurability - algorithm level approach"**, in Proc. 4th International Conference on 3G Mobile Communication Technologies, London, UK, June 2003.
30. A. Kountouris and C. Moy, **"Reconfiguration in software radio systems"** in Proc. 2nd Workshop on Software Radios, Karlsruhe, Germany, March 2002, pp. 119–124
31. M. Mehta, N. Drew, and C. Niedermeier, **"Reconfigurable terminals : an overview of architectural solutions", IEEE Comms. Mag., pp. 82–88, 8 2001.**
32. S. Srikanteswara, J. Reed, P. Athanas, and R. Boyle, **"A soft radio architecture for reconfigurable platforms"** IEEE Comms. Mag., vol. 38, pp. 140–147, 2 2000.
33. W. Bennet, J. R. MacLeod, J. P. McGeehan, **"Broadband High Dynamic-Range RF Transmitter Technology for Flexible Multi-Standard radios"**, ACTS, Mobile Communications Summit, Rhodes, June 8-11, 1998.
34. C. Bonnet, G. Caire, A. Enout, P. Humblet, G. Montalbano, A. Nordio, D. Nussbaum, T. Höhne, R. Knopp, B. Rimoldi, **"An open software-radio architecture supporting adavanced 3G+ systems"**, Annales des télécommunications, Tome 56, n^o5-6, mai-juin 2001.
35. A. Ciochocki, R. Unbehauen, **"Neural Networks for Optimization and Signal Processing"**, Wiley and sons, New York, 1993.
36. N. J. Drew, P. Tottle, **"IC Technologies and Architectures to Support the Implementation of Software Define Radio Terminals"**, ACTS, Mobile Communications Summit, Rhodes, June 8-11, 1998.

37. D. Efstathiou, J. Fridman Z. Zvonar, **"Recent Developments in Enabling Technologies for Software Defined Radio"**, IEEE Communications Magazine, August 99, pp. 112-117.
38. H. Harada, M. Fujise, **"Multimode Software Radio System by Parameter Controlled and Telecommunication Toolbox Embedded Digital Signal Processing Chipset"**, ACTS, Mobile Communications Summit, Rhodes, June 8-11, 1998.
39. **"GNU Radio : Introduction and Computational Capabilities of the Open Source GNU Radio Project"**, Wireless Innovation Forum Tom Rondeau (Center for Communications Research, USA), Decembre 2010
40. **Universal Software Radio Peripheral Architectures and Products Lists,** `http://www.ettus.com`
41. **Research and experimentation with the DECT,** `http://dect.osmocom.org/trac/dect`
42. **Research and experimentation with the GSM network,** `http://misterhac.appspot.com/airprobe.org`
43. **Research and experimentation with the TETRA trunked radio system,** `http://tetra.osmocom.org/trac/`
44. **"The Future is Not the Past ; Watch the Trends"**, Paul Kolodzy (Kolodzy Consulting) , Decembre 2010
45. `http://warp.rice.edu/trac/wiki/Projects/UCI_NIJ-SDR`
46. `http://www.ettus.com/`
47. Gerhard de Koning Gans, Jaap-Henk Hoepman, and Flavio D. Garcia **A Practical Attack on the MIFARE Classic**, Disponible à `http://www.proxmark.org/documents/mifare_weakness.pdf`
48. `http://www.ssi.gouv.fr/fr/anssi/publications/communiques-de-presse/legislation-en-matiere-d-outils-d-espionnage.html`

Deuxième partie

29 novembre 2011

Mise en œuvre de politiques de protection des flux d'information dans l'environnement Android

Valérie Viet Triem Tong, Radoniaina Andriatsimandefitra, Stéphane Geller, Simon Boche, Frédéric Tronel, Christophe Hauser

SUPELEC
Équipe Sécurité des Systèmes d'Information (EA4039)
prénom.nom(@)supelec.fr

Résumé Les téléphones portables sont des espaces de stockage de données d'horizons très différents pour lesquelles il n'existe pas encore de moyen de protection *grand public* efficace. Néanmoins depuis plusieurs années, des travaux de recherche s'intéressent au suivi dynamique de flux d'information dans un système. Il a été montré que ces travaux étaient bien adaptés à la protection de l'information. Nous pensons que le développement de ce type d'approche par un public large se heurte principalement à la définition de la politique qui régit le moniteur de flux d'information. Ce papier s'intéresse donc à la faisabilité d'une telle approche pour le système Android. Dans ce travail, nous détaillons pas à pas la construction d'une politique de flux d'information pour Android, puis nous mettons en œuvre cette politique. Notre objectif principal est de répondre aux questions suivantes : *en pratique, sur un système réel, combien d'informations sensibles peut-on surveiller ? Sur un téléphone portable quelles sont les informations qu'il est souhaitable de surveiller ? Pouvons nous proposer une/des politiques de flux d'information par défaut pour de tels systèmes ?*

1 Introduction

Aujourd'hui un téléphone mobile est devenu un important centre de données. En effet sur un téléphone mobile, nous trouvons aussi bien des informations personnelles de l'utilisateur comme des photos, des vidéos, des SMS/MMS que des données concernant le téléphone lui même (caractéristiques techniques, numéro d'identification) ou encore des données de géolocalisation. Enfin il n'est pas rare de trouver des données liées à l'entreprise employant l'utilisateur, un téléphone mobile étant synchronisable avec un ordinateur de bureau. Toutes ces données ont des propriétaires distincts, aussi bien l'utilisateur final que son employeur, l'encarteur ou l'opérateur et se retrouvent sur un environnement contrôlé par un utilisateur parfois peu sensible aux questions de sécurité hors de portée de la surveillance de leur propriétaire. Les problèmes surviennent lorsque cet utilisateur final installe des applications sur son téléphone. Sous Android, ces applications vont demander l'accès aux services existants sur le téléphone afin de pouvoir fonctionner. Ces applications peuvent alors accéder aux données présentes sur le téléphone et parfois en avoir une utilisation malveillante. Par exemple, fin

2010 les utilisateurs téléchargeant des applications sur une plateforme du type *AndroidMarket* [8] étaient susceptibles d'installer une application infectée par le virus Geinimi[14]. Ce virus s'installait sur le téléphone en bénéficiant des droits accordés à l'application infectée, collectait des informations comme les coordonnées géographiques de l'utilisateur, les identifiants du téléphone (numéro IMSI et IMEI) et la liste des applications installées, avant de les envoyer à un serveur distant. L'utilisation qu'a pu faire le serveur des ces informations est mal connue. Néanmoins, la fuite conjointe des informations de géolocalisation du téléphone et des caractéristique de ce téléphone permettent de connaître sans équivoque la localisation du téléphone et l'actualisation régulière de ces données permettent même de suivre les déplacements de ce téléphone (et de son possesseur). D'autres virus existent aujourd'hui comme *DroidDream*[11], *HongTouTou*[10] ...et leurs actions malveillantes est souvent d'accéder à des informations stockées sur le téléphone pour les modifier, les effacer ou les faire fuir hors du téléphone.

Nous pensons qu'une bonne manière de se protéger de ce type d'actions malveillantes est d'être capable de suivre dynamiquement l'évolution des informations et les flux d'information ayant lieu sur le téléphone afin d'interdire les flux d'information illégaux. Le suivi dynamique de flux d'information fait l'objet de nombreux travaux et de nombreuses publications. Sous l'environnement Android, Enck *et al* dans [4] ont, les premiers, utilisé ces techniques pour mettre en évidence la réalité de la fuite d'information. Dans leur travail, les informations suivies sont peu nombreuses (géolocalisation, données liées à l'utilisateur et son opérateur et les identifiants de l'appareil) et la politique de sécurité régissant le moniteur de flux d'information n'est pas formellement définie. Nous pensons cependant que la définition de la politique de sécurité reste un obstacle à l'utilisation de ce type d'approche par un public plus large. La politique de sécurité définit les informations qui doivent être surveillées, elle détermine aussi quels sont les flux d'information autorisés/illégaux.

Dans ce papier, nous nous intéressons précisément à la définition de la politique de sécurité. Nous détaillons pas à pas la construction d'une politique de flux pour Android (section 5) suivant un modèle théorique présenté dans [7], [6] et rappelé en section 3. Le but de notre travail est d'être capable de répondre aux questions suivantes : quelles sont les informations sensibles sur un environnement Android habituel ? Pouvons nous proposer une ou plusieurs politiques de protection par défaut de ces informations ? Comment dimensionner cette politique afin d'avoir un coût raisonnable à l'exécution ? Nous nous efforçons d'apporter des éléments de réponses à ces questions en section 6.

2 Positionnement théorique et pratique

Il existe un grand nombre de travaux s'intéressant au suivi de flux d'information. Ces approches utilisent toutes une marque (appelé également label, tag, teinte ...) qui permet de caractériser un contenu. La marque attachée à un contenu change à chaque fois que ce contenu est modifié. Ces approches ont en commun le fait qu'elles reposent sur la définition d'une politique de flux d'information qui détermine les contenus qui doivent être marqués et comment ces contenus peuvent évoluer au sein du système.

Dans [13], Suh *et al* présentent un moniteur de flux d'information capable de surveiller l'évolution des contenus sur une machine en observant les actions des applications. Après avoir observé que les actions malveillantes effectuées sur une machine résultent bien souvent d'information provenant de canaux d'entrées non sûrs, les auteurs proposent d'utiliser une politique de flux distinguant les informations provenant de ces canaux et spécifiant où ces informations peuvent se propager. Dans ce travail la politique de sécurité est donc très simple, les informations sont uniquement caractérisées par leur provenance (sûre/non sûre) et la politique détermine les actions autorisées sur les données non sûres, (resp. sur les données sûres).

Dans [16], Zeldovich *et al* présentent un système d'exploitation nommé Histar, permettant un contrôle strict des flux d'information entre les processus en utilisant les labels nommés *Asbestos labels* [3]. Dans ce système, chaque processus se voit associer deux labels. Le premier représente la *contamination courante* du processus tandis que le second représente le maximum de contamination autorisé pour ce processus. La granularité de cette approche est plus fine que la précédente car les labels peuvent prendre cinq valeurs possibles au lieu de deux, la politique exprimée est donc un peu plus fine. D'autre part, dans ce travail la définition de la politique est décentralisée : chaque application détermine la valeur de son label et donc le degré de contamination des informations qu'elle va faire circuler dans le système. Dans [2] P. Efstathopoulos *et al* explique que la définition d'une politique de flux pour Histar est un travail difficile et propose un langage dédié pour la spécification de telles politiques.

À la fin des années 90, A. Myers et B. Liskov ont proposé dans [12] un moniteur de flux intra-applicatif à grain fin. Dans ce travail, les données manipulées par des programmes Java sont caractérisées par leur propriétaire et par les lecteurs autorisés. Cette approche est un pas important vers la définition de politique de flux fine et expressive. Cette approche étant dédiée au suivi de flux d'information à l'intérieur même d'un programme, la définition d'une politique pour un système d'exploitation tout entier reste un problème très difficile.

Enfin, dans [9,15,7,6] nous avons proposé un modèle de politique de flux d'information à grain très fin. Dans notre modèle, chaque information considérée

comme sensible est identifiée et la politique détermine ensuite où les informations sensibles ont le droit de se trouver. Dans ce papier, nous poursuivons nos efforts de recherche dans ce domaine et nous nous concentrons sur l'étude pratique de l'utilisation de telles politiques dans le système Android. Nous avons préféré le système Android à d'autres systèmes tels que *iOS, Windows Phone ...* car celui ci est un système plus ouvert ce qui facilite la compréhension de son architecture et de son fonctionnement ainsi que son adaptation aux besoins de chacun. Sur plusieurs aspects, son fonctionnement le place d'ailleurs comme un candidat de choix pour notre modèle et implémentation. Du point de vue du modèle théorique de la politique de flux, Blare détermine en effet la politique d'un processus en fonction de l'utilisateur qui l'a créé et de la politique liée à l'application que lance ce processus. Le système Android utilise un UID et un processus différent pour chaque application, ce qui nous permet d'appliquer précisément notre modèle. Au contraire, *Windows Phone et iOS* ne font pas une distinction aussi fine, ce qui implique que sur ces systèmes nous ne pourrions pas utiliser de politique de flux aussi précise que celle présentée ici.

Enfin d'un point de vue pratique, Blare est un module de sécurité Linux tirant profit du framework LSM. Ce framework LSM est présent dans les systèmes de type *Linux* mais n'existe pas pour les systèmes type *Windows Phone et iOS*. Le portage de Blare pour *Windows Phone et/ou iOS* nécessitera donc en phase amont de développer un framework de sécurité assurant les mêmes services que LSM.

3 Modèle de politique de flux

Notre modèle de politique s'attache à distinguer les contenus, c'est à dire les données (exécutable ou non), les informations de leur conteneur. Dans notre terminologie, un conteneur est une entité logique ou physique capable de détenir de l'information. Au niveau d'un langage de programmation, un conteneur d'information est par exemple une variable. Au niveau d'un système d'exploitation, un conteneur est un fichier, un processus, une plage mémoire, un utilisateur ... Nous avons proposé dans [7,6] de définir par le biais d'une politique de flux d'information comment les informations peuvent se mélanger et circuler dans les différents conteneurs d'un système d'exploitation. Dans ce papier, nous rappelons brièvement le modèle théorique d'une politique de flux d'information, nous renvoyons notre lecteur vers les publications antérieures [15,7,6] sur ce travail pour plus de détails.

3.1 Les données

Une étape importante de la mise au point d'une politique de flux d'information est de sélectionner les données qui sont sensibles sur le système et qu'il est sou-

haitable de surveiller. La sélection de ces données est un travail important pour la pertinence de la politique : choisir trop de données à surveiller risque fortement de conduire à une politique de sécurité inutilisable car trop coûteuse (en temps de calcul, ou en espace de stockage) à faire respecter. Aujourd'hui, il n'existe, à notre connaissance, aucune étude permettant de quantifier le nombre de données surveillables par un moniteur de flux d'information pour un coût raisonnable sur un système d'exploitation réel. C'est un des objectifs du travail présenté ici. La définition d'une politique de flux selon notre modèle suppose qu'un administrateur averti est capable d'identifier l'ensemble des données sensibles existantes sur le système. Plus précisément, cet administrateur devra associer un identifiant numérique positif à une donnée et un identifiant numérique négatif à un code exécutable [1]. Chaque code sensible est vu à la fois comme une donnée sensible et comme un code. Nous associons un identifiant numérique i à chaque code sensible. L'information vue comme une donnée non exécutée (par exemple la suite des instructions d'un programme) est attachée à $|i|$ et la même donnée vue comme du code exécuté qui donnera naissance à un processus est attaché à $-|i|$. Prenons un exemple : si notre administrateur considère qu'un programme `monprg.exe` est une donnée sensible, alors il devra lui attacher un nombre unique par exemple 3 et pour toute la suite $+3$ désignera le code de ce programme vu comme un texte contenant une suite d'instruction alors que -3 désignera l'information générée par l'exécution de ce code [2]. La première étape de la construction d'une politique de flux sera donc de déterminer avec précision et sans excès deux ensembles $\mathcal{I}$ et $\mathcal{X}$ désignant respectivement l'ensemble des identifiants associés aux informations non-exécutées sensibles (les identifiants positifs) et l'ensemble des identifiants associés aux programmes sensibles (les identifiants négatifs). L'identifiant 0 exclu car ces deux ensembles doivent être nécessairement disjoints. Dans la section 5.1 nous détaillons quelles données exécutables et non-exécutables nous semblent sensibles pour l'environnement Android.

3.2 Les conteneurs d'information persistants ou volatiles

Sur un système d'exploitation, un conteneur d'information est dit persistant lorsque son contenu persiste après l'arrêt du système. Il s'agit typiquement un fichier. L'ensemble des conteneurs persistants sera noté $\mathcal{C}$. Au contraire, un conteneur d'information sera dit volatile lorsqu'il est créé temporairement pour les besoins de fonctionnement d'une ou plusieurs applications. Un conteneur d'information volatile est typiquement une plage mémoire, une *socket* .... Au niveau de la définition de la politique de flux d'information, il faut retenir que la politique liée à un conteneur persistant est définie au démarrage du système et éventuellement modifiable par la suite, alors que la politique liée à un conteneur volatile

1. L'identifiant nul est donc exclu
2. ou encore le *programme* lui même

dépendra de la politique du processus qui créera ce conteneur. La politique d'un conteneur persistant est la liste des combinaisons d'informations exécutables ou non que l'on autorise à se trouver dans ce conteneur. Plus précisément, la politique attachée à un conteneur persistent c sera constituée d'un ou plusieurs ensembles d'éléments de $\mathcal{I} \cup \mathcal{X}$ signifiant que n'importe quelle information calculée à partir d'un de ces ensembles a le droit de se trouver dans c. Par exemple, si la politique d'un fichier f est égale à $\{\{1, 2, 3, -8\}, \{3, 4, -5\}\}$ alors cela signifie qu'un processus résultant de l'exécution du code marqué -8 a le droit d'écrire de l'information générée par lui même dans le fichier f et que ce programme a le droit d'injecter dans f des données provenant des informations marquées par $1, 2, 3$, un autre processus provenant de l'exécution du code marqué -5 a le droit d'injecter dans f une donnée provenant des données marquées par $3, 4$.

Les utilisateurs

La politique de flux d'information liée aux utilisateurs détermine à quels mélanges d'information marquée comme sensible un utilisateur a le droit d'accéder. L'ensemble des utilisateurs sera noté $\mathcal{U}$ et, exactement comme pour les conteneurs persistants, la politique d'un utilisateur $u \in \mathcal{U}$ est donc un élément de $\mathcal{P}(\mathcal{I} \cup \mathcal{X})$ soit un ensemble d'éléments de $\mathcal{I} \cup \mathcal{X}$.

Les programmes, les processus

Nous distinguons ici les programmes des processus. Dans ce travail, un programme ou encore une application est essentiellement caractérisé par une donnée, une information que nous savons être exécutable. Au début de ce travail, nous avons proposé d'identifier certaines de ces données exécutables comme sensibles, ce qui a permis de construire l'ensemble $\mathcal{X}$. L'ensemble des programmes sera noté $\mathcal{P}rog$, nous avons $\mathcal{X} \subseteq \mathcal{P}rog$. Un processus résulte de l'exécution d'un programme et donc d'une donnée exécutable. Enfin, la politique de flux d'information liée à un programme détermine à quels mélanges d'information ce programme a le droit d'accéder. La politique d'un programme $p \in \mathcal{P}rog$ est donc là encore un élément de $\mathcal{P}(\mathcal{I} \cup \mathcal{X})$. Lorsqu'un processus exécute un programme, la politique liée à ce processus dépendra à la fois de la politique liée au programme exécuté et de la politique attachée à l'utilisateur propriétaire de ce processus. Plus formellement, lorsqu'un utilisateur u ayant une politique $p_1 \in \mathcal{P}(\mathcal{I} \cup \mathcal{X})$ exécute une donnée exécutable x ayant une politique $p_2 \in \mathcal{P}(\mathcal{I} \cup \mathcal{X})$ la politique attachée au processus résultant est $p_1 \cap p_2$.

En résumé, la définition d'une politique de flux d'information pour notre approche est simplement la définition d'une fonction $\mathbb{P} : \mathcal{C} \cup \mathcal{U} \cup \mathcal{P}rog \to \mathcal{P}(\mathcal{P}(\mathcal{I} \cup \mathcal{X}))$ qui à chaque conteneur persistent, utilisateur, ou programme associe l'ensemble des ensembles d'information qu'il peut contenir. La nature de la politique de flux

d'information liée à un processus est similaire mais elle sera calculée en fonction du code exécuté et de l'utilisateur responsable du processus.

Mise en œuvre de la politique
Le modèle de politique de flux d'information que nous venons de décrire peut être mise en œuvre via l'utilisation d'un moniteur de flux d'information capable de suivre l'évolution des contenus dans le système. Ce moniteur de flux peut être implémenté au niveau inter-applicatif : il surveille les flux d'information effectué par l'ensemble des applications sur un système d'exploitation. Ce moniteur peut aussi s'implémenter au niveau intra-applicatif et donc surveiller les flux d'information ayant lieu au sein même d'un processus. Nous nous intéressons ici au premier type de moniteur car nous souhaitons surveiller le système tout entier et l'ensemble des données se trouvant sur un téléphone, une tablette. Le moniteur que nous développons pour Android est une extension et un portage de celui présenté dans [9]. Son fonctionnement interne ne sera pas développé ici.

Rapidement et intuitivement, ce moniteur utilise des méta-données appelées *tag* sur les conteneurs d'information (fichiers et processus). Un premier tag appelé *tag information* liste les informations sensibles à l'origine de son contenu. Un second tag rappelle la politique liée à cet objet. Par exemple un fichier f dont la politique a été fixée à $\mathbb{P}(f) = \{\{-1, -3, 1, 2\}, \{-4, 5, 6\}\}$ et contenant les données sensibles 1 et 2 aura un tag information égal à $\{1, 2\}$ et son tag politique sera directement égal à sa politique (donc vaudra $\mathbb{P}(f) = \{\{-1, -3, 1, 2\}, \{-4, 5, 6\}\}$).

Le moniteur surveille ensuite les appels systèmes responsables des flux d'information entre les applications (`read, write, execve, fork`). À chaque observation d'un appel système engendrant un flux d'information, les tags des conteneurs modifiés (processus, fichier) sont mis à jour de sorte que le tag information du fichier modifié reflète toujours l'origine du contenu réel de l'objet. Si, à la suite d'un flux le tag information montre que l'objet contient une donnée non-autorisée par le tag politique de cet objet, une alerte est levée. Plus de détails formels sur le fonctionnement de ces tags sont données dans [7] et [6].

4 Flux d'informations dans Android

Android est un système d'exploitation *Open Source* proche du système Linux. L'observation des flux d'information peut se faire à un premier niveau de granularité en observant les appels systèmes en `read,write, fork, execve` mais il devra aussi se faire au niveau applicatif car Android met en œuvre un mécanisme de communication interprocessus particulier que nous détaillons rapidement ici. Plus précisément, nous présentons dans ce début de section les flux d'information possibles sur un système Android à travers le fonctionnement des applications livrées à l'utilisateur. Nous renvoyons notre lecteur vers [5] pour plus de détails.

Les applications sont écrites en Java et utilisent une version du SDK qui diffère de *Java SE*. Sous Android, chaque application Java est exécutée dans une machine virtuelle (Dalvik) particulière et est généralement sous un utilisateur particulier. Chaque application est décrite par un fichier appelé *AndroidManifest.xml* qui contient des données relatives à l'application et à ses composants. Les composants formant une application sont de quatre types :

- **Activity** fournit une interface utilisateur.
- **Service** est un composant tournant en tâche de fond. Il n'y a pas d'interaction directe avec l'utilisateur. Il peut être utilisé pour traiter des données.
- **Content Provider** est un fournisseur de contenu comme son nom l'indique. Il est utilisé pour partager des données avec d'autres applications. Un exemple est le fournisseur de contacts.
- **Broadcast Receiver** est destiné à définir les réactions suite aux messages envoyés en *broadcast* dans le système.

Sous Android, les applications fonctionnent de manière collaborative et sont conçues dans cette optique. Intuitivement, les fonctionnalités implémentées ou les données utilisées par une application sont partageables avec les autres applications. Par exemple, l'application appareil photo peut faire appel aux applications *gmail, mail, facebook ...* pour partager une photo prise. Les mécanismes permettant cette collaboration sont des mécanismes qui engendrent des flux d'information dans le système.

Communications entre les applications

Intent et intent filter

Un *intent* est utilisé pour transmettre des messages vers un ou plusieurs composants. Le destinataire d'un intent peut ne pas faire partie de la même application. Son contenu peut être une action à réaliser, des informations liées à un évènement apparu (niveau de batterie faible par exemple)...Une description plus complète est disponible sur [1].

Un intent est qualifié d'*explicit* lorsque le composant destinataire est précisé par l'émetteur. Seul lui reçoit le message. Il est qualifié d'*implicit* lorsque l'émetteur décrit un ensemble de critères permettant au système de définir le(s) destinataire(s). L'intent *filter* est un filtre qui définit les *implicit intents* attendus par un composant. Tout message dont les critères correspondent à un filtre sera transmis au composant ayant déclaré ce filtre. Par exemple, un utilisateur clique sur un lien envoyé dans un message. S'il y a plusieurs navigateurs sur son système, un *intent* demandant l'ouverture de la page sera envoyé à chacun.

Remote method calls

Une autre manière de transmettre de l'information est d'exécuter des méthodes

à distance sous Android. Cette fonctionnalité est similaire au *Java RMI (Remote Method Invocation)* et est proposée pour accéder à plusieurs services offerts par le système ainsi qu'à des informations liées à ce dernier ou à l'appareil. Le tableau 1 présente quelques classes instanciées par le système et dont les méthodes peuvent être appelées directement à l'intérieur des applications.

Service	Nombre d'applications clientes
LocationManager (géolocalisation)	3
TelephonyManager (géolocalisation, appareil, carte SIM)	9
SensorManager (capteurs)	3
SmsManager (messages)	2

Table 1. Quelques services disponibles à distance et le nombre d'applications les utilisant

ContentProvider

Enfin, le composant *ContentProvider* est également à l'origine de certains flux dans le système. Ce composant est utilisé pour partager des données. Le tableau 2 liste des providers fournis sous Android.

L'accès à certaines données est protégé par des permissions. Nous nous en sommes ainsi servis pour avoir une approximation du nombre d'applications interagissant avec *les providers*. À la lecture du tableau, il est important de garder en mémoire que 0 ne signifie pas forcément qu'il n'y aura jamais de flux mais qu'aucun contrôle n'est effectué en terme de permissions.

Fichiers

Au niveau applicatif, les flux notables liés aux fichiers sont ceux entre un content provider et le(s) fichier(s) stockant les données qu'il propose.

5 Comment mettre au point une politique de flux pour Android

La mise au point d'une politique de flux pour Android nécessite une observation précise de la nature des données présentes sur l'équipement. Ce travail requiert beaucoup d'expertise sur le système, à titre d'exemple, la mise au point de la politique que nous détaillons a pris plusieurs semaines. Nous pensons que la politique que nous détaillons ici est à la fois suffisamment précise et générale pour servir de base à l'établissement de politiques dérivées spécialisées pour des environnements particuliers. Cette politique est générale puisque nous l'avons

Providers	Permissions	Applications
ContactProvider	READ_CONTACTS, WRITE_CONTACTS	17
CalendarProvider	READ_CALENDAR, WRITE_CALENDAR	2
DownloadProvider	ACCESS_ALL_DOWNLOADS	0
DrmProvider		0
MediaProvider		0
TelephonyProvider	READ_SMS, WRITE_SMS	4
UserDictionary Provider	READ_USER_DICTIONARY, WRITE_USER_DICTIONARY	1

Table 2. Providers présents sous Android, les permissions demandées et le minimum d'applications déclarant ces permissions

établie pour un téléphone portable sous Android avec ses composants installés *par défaut* . En effet nous nous sommes basés sur l'environnement par défaut tel qu'il est disponible sur `http://developer.android.com/index.html`. Cette politique est précise puisque nous avons détaillé chaque information qui nous a semblé pertinente. Pour mettre au point notre politique, nous commençons par mettre en évidence les informations sensibles ce qui constitue l'ensemble $\mathcal{I}$, puis les données exécutables sensibles, ce qui constitue l'ensemble $\mathcal{X}$. Nous détaillons ensuite quels mélanges d'information sont autorisés et dans quels fichiers et à quels mélanges les utilisateurs sont autorisés à accéder. Dans cette partie nous détaillons la construction d'une politique qui nous semble pertinente pour Android. Dans la section suivante nous construisons des variations autour de cette politique afin de savoir si la mise en œuvre de cette politique est raisonnable en terme de surcoût engendré à l'exécution, nous cherchons également à savoir si il est possible d'en utiliser de plus précises encore.

5.1 Les données sensibles sous Android

Nous avons distingué 75 informations sensibles. Environ la moitié d'entre elles correspondent à des données non-exécutables. Comme précisé dans la section 3, nous leur avons associé un identifiant numérique non nul. Les informations non exécutées (qui forme l'ensemble $\mathcal{I}$) les plus pertinentes sont détaillées dans la table 3.

L'autre moité des informations sensibles est formée par le code des applications. Chaque code sensible est vu à la fois comme une donnée sensible (un élément de $\mathcal{I}$) et comme un code (un élément de $\mathcal{X}$). Nous associons donc un identifiant numérique i non nul à chaque code sensible. L'information vue comme *une donnée non exécutée* est attachée à $|i|$ et la même donnée vue comme du *code exécuté* est attachée à $-|i|$. Les plus pertinentes d'entre elles sont listées dans la table 4.

Information	Identifiant numérique
Données de géolocalisation	1
SMS reçus/envoyés	3
Coordonnées email du propriétaire	5
Historique des appels	6
Nom de l'opérateur téléphonique	7
International Mobile Equipment Identity (IMEI)	9
Contacts et données associées (téléphone, adresse . . .)	10
Agenda	11
Configuration VPN	19
Liste des applications installées	34
⋮	⋮

Table 3. Une partie des données sensibles – non exécutables

Origine de l'information	Identifiant vu comme un élément de $\mathcal{I}$	Identifiant vu comme un élément de $\mathcal{X}$
TelephonyProvider	37	-37
CalendarProvider	38	-38
Mms app	59	-59
Browser	72	-72
DrmProvider	73	-73
⋮	⋮	

Table 4. Extrait des données atomiques exécutables

5.2 Les utilisateurs

Sous Android, un utilisateur unique est généralement associé à chaque application. Nous pensons que cette particularité du système est intéressante car elle permet un premier cloisonnement des informations par le contrôle d'accès. Néanmoins, la pratique a montré que ce contrôle d'accès n'est pas suffisant car il ne protège pas l'accès à l'information : des fuites d'informations restent possibles.

5.3 Les conteneurs persistants

Le système d'exploitation Android utilise plus de 30000 fichiers. Au démarrage, nous avons une cinquantaine de processus qui tourne et 42 applications installées par défaut. La politique attachée à un fichier f est vide si ce fichier n'est pas autorisé à contenir des informations sensibles. Nous n'avons donc à détailler que les fichiers autorisés à contenir au moins une information sensible, ce qui réduit considérablement le temps de construction de la politique. Dans ce travail, nous avons établi une politique qui autorise 96 fichiers à contenir au moins une information sensible.

5.4 La politique

Comme décrit dans la section 3, la politique associe à chaque fichier, chaque programme et chaque utilisateur donnés des éléments de $\mathcal{P}(\mathcal{I} \cup \mathcal{X})$, ce qu'il est autorisé à contenir. Nous avons représenté cette politique sous la forme d'une matrice. Notre matrice liste en colonnes les informations et en lignes les fichiers, programmes puis utilisateurs. Notre matrice possède 117 colonnes : 75 données non exécutées et 42 données exécutées. Notre matrice possède 260 lignes qui détaillent les conteneurs (fichiers, programmes, utilisateurs) autorisés à accéder ou à contenir au moins une information identifiée par une colonne (et donc vu comme sensible). Une croix à l'intersection de la ligne l et de la colonne c signifie que l'information listée en colonne c est autorisée à se trouver dans le conteneur listé en ligne l. L'ensemble des croix dans une colonne c détermine l'ensemble des conteneurs possibles pour une information. L'ensemble des croix dans une ligne l détermine quels mélanges d'information sont autorisés pour le conteneur de la ligne l. Une partie de cette matrice est détaillée dans la table 5.

5.5 Modification de la politique

La politique que nous présentons peut être modifiée : cette politique peut donc servir de base pour la construction d'autres politiques. Plus simplement, lorsque l'utilisateur installe une nouvelle application sur son téléphone, il est nécessaire d'étendre la spécification de la politique courante pour prendre en compte cette nouvelle application. Une telle modification est simple à mettre en œuvre dans notre modèle, nous distinguons essentiellement deux cas :

– L'utilisateur, l'administrateur décide de considérer une nouvelle information sensible. Il faut alors utiliser un nouvel identifiant pour cette information, ajouter une colonne dans la matrice et déterminer quels conteneurs (fichier, programme, utilisateur) peuvent accéder à cette donnée.
– Un nouveau conteneur d'information est créé (par exemple lorsque le propriétaire du téléphone installe une nouvelle application). Si ce conteneur est autorisé à contenir au moins une information sensible alors une nouvelle ligne correspondant au conteneur est ajoutée à la matrice.

	Identifiant des informations									
	11	-44	10	6	-55	-39	3	9	37	-37
Fichiers										
calendar.db	×									
contacts2.db			×	×						
mms.db							×			
telephony.db								×		
TelephonyProvider.apk									×	
Programmes										
com.android.calendar	×	×								
com.android.providers.contact			×	×	×					
com.android.proviers.telephony							×	×		×
Utilisateurs										
10024	×	×								
10004			×	×	×	×				
1001							×	×		×

Table 5. Un extrait de la politique de flux d'information

6 Expériences

Le but de notre travail est d'évaluer comment dimensionner une politique d'information pour un équipement fonctionnant sous l'environnement Android. Dans la section précédente, nous avons détaillé cet environnement afin de caractériser les informations potentiellement sensibles et leur politique associée. Dans cette partie, nous implémentons plusieurs politiques de complexité croissante, nous mesurons ensuite la surcharge induite en temps de calcul par ces politiques. Le moniteur de flux que nous utilisons est une extension de celui présenté dans [15]. Cette extension comprend en particulier la gestion fine des processus.

Pour ce travail, les tests ont été menés sur un émulateur de téléphone Android. Cet émulateur est contenu dans les sources du projet Android disponible librement sur `android.git.kernel.org`. La machine hôte que nous utilisons tourne avec le système d'exploitation Ubuntu 10.10, dispose d'une mémoire vive de $3Go$ et d'un processeur Core 2 duo de $2GHz$. L'émulateur utilise lui un noyau qui lui est

spécifique. Il s'agit du noyau *goldfish*. Sa version la plus récente disponible dans les sources d'Android est la 2.6.29.

Notre premier test a consisté à déterminer un seuil à partir duquel l'initialisation de la politique de flux d'information sur le système pouvait induire une gêne pour un utilisateur. Pour cela, nous avons utilisé un script d'initialisation de la politique de flux d'information qui attribue des tags aux fichiers de notre système. Nous avons mesuré le temps passé à marquer ces fichiers et nous avons fait varier le nombre de fichiers à marquer ce qui correspond au nombre d'information à surveiller. Le tableau 6 synthétise les résultats que nous avons obtenus. En particulier, il est important de noter que la politique que nous avons dimensionnée dans les sections précédentes de ce papier est initialisée sur le système Android en 2 secondes ce qui n'occasionne pas de gêne pour un utilisateur. En revanche une petite gêne se remarquera certainement dès le marquage de 400 informations sensibles. L'initialisation de la politique ne sera faite qu'une fois : à l'installation du moniteur de flux, le temps nécessaire à cette initialisation est donc très raisonnable puisque même l'opération de marquage de 1600 conteneurs ne dépasse pas la minute.

	Quantité d'information atomique surveillée						
	50	75 (politique réelle)	100	200	400	800	1600
Durée d'initialisation	1,61	2	3,21	7,31	14,85	28,46	52,63

Table 6. Durée d'initialisation (en secondes)

Nous avons ensuite estimé l'espace disque supplémentaire utilisé par le moniteur de flux d'information Blare. Le moniteur que nous utilisons actuellement est développé sous forme de module de sécurité LSM. La surcharge engendrée en espace disque peut donc se mesurer en comparant la taille d'un noyau avec et sans Blare. Les résultats détaillés dans le tableau 7 montrent que l'espace utilisé en plus par le moniteur de flux d'information est minime, même pour un téléphone.

	Sans Blare	Avec Blare
Taille du noyau (en Octet)	1672160	1680152
Surcharge	0%	0,47%

Table 7. Taille du noyau Android sans et avec le moniteur Blare

Nous avons enfin estimé la surcharge en temps induite par le moniteur lors de l'utilisation courante du téléphone. Pour cela, nous avons développé une application effectuant un ensemble d'opérations susceptibles d'être réalisées par un utilisateur : notre application insère 240 contacts dans le carnet d'adresse, envoie un message tous les 20 contacts ajoutés puis récupère les informations liées à

ces 240 contacts, ce qui fait 472 opérations en tout. Nous avons exécuté cette application afin de connaître son temps d'exécution dans un environnement habituel puis nous avons répété l'expérience dans un environnement intégrant cette fois-ci notre moniteur de flux d'information. Nous avons aussi comparé les temps moyens d'affichage de l'interface. Le tableau 8 détaille ces résultats. Nous pouvons constater que la surcharge engendrée par le moniteur Blare est négligeable et aux yeux de l'utilisateur sa présence pourrait même être imperceptible. En effet, notre application effectue 472 opérations en tout. Chaque opération prend donc en moyenne 0.372 secondes. La surcharge est de 8% ce qui représente 0.03s de temps moyen supplémentaire. Nous pensons que cette surcharge ne sera pas ressentie.

	Durée moyenne d'exécution de l'opération	Surcharge	Temps moyen d'affichage	Surcharge
Sans Blare	168535 ms	0%	170978 ms	0%
Avec Blare	182429 ms	8%	178905 ms	4%

Table 8. Surcharge engendrée par Blare. L'unité de temps est ici milliseconde

7 Conclusion

Le travail présenté dans ce document pose les bases vers une utilisation plus générale et systématique des techniques de suivi de flux d'information pour la sécurisation des données sensibles. Il existe aujourd'hui beaucoup de travaux visant à observer les échanges d'informations sur un système et les prochains challenges de ce domaine visent à montrer comment ces moniteurs peuvent être utilisés par le grand public sur des systèmes courants. En particulier, il faut maintenant étudier comment configurer, utiliser et administrer ces moniteurs. Nous avons ici implémenté un moniteur de flux d'information qui surveille les échanges d'information entre des objets en surveillant les appels systèmes effectués par les processus. Ce moniteur a été porté sur Android qui est aujourd'hui le système d'exploitation le plus utilisé par les *ordiphones*. En introduction de cet article, nous nous demandions "*Sur un téléphone portable quelles sont les informations qu'il est souhaitable de surveiller ?*" Notre travail a montré que l'ensemble de données permettant de caractériser de manière unique un téléphone et/ou son utilisateur sont des données sensibles. Ces données sont au nombre de 75 dont la moitié représente des données exécutables. Nous nous demandions aussi "*en pratique, sur un système réel, combien d'informations sensibles peut-on surveiller ?* " Nos expérimentations ont montré que la surcharge induite par le moniteur pour la surveillance de ces informations est négligeable. Enfin nous avons créé une véritable politique de flux d'information pour ces données sensibles. Cette politique est disponible sur

`http://www.rennes.supelec.fr/blare/`, nous espérons qu'elle puisse servir à tous.

Références

1. Dev guide - intents and intent filters.
2. Petros Efstathopoulos and Eddie Kohler. Manageable fine-grained information flow. In Joseph S. Sventek and Steven Hand, editors, *EuroSys*, pages 301–313. ACM, 2008.
3. Petros Efstathopoulos, Maxwell Krohn, Steve Vandebogart, Cliff Frey, David Ziegler, Eddie Kohler, David Mazières, Frans Kaashoek, and Robert Morris. Labels and event processes in the asbestos operating system. In *In Proc. 20th ACM Symp. on Operating System Principles (SOSP*, pages 17–30, 2005.
4. William Enck, Peter Gilbert, Byung-gon Chun, Landon P. Cox, Jaeyeon Jung, Patrick McDaniel, and Anmol N. Sheth. Taintdroid : An information-flow tracking system for realtime privacy monitoring on smartphones. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2010.
5. Adrienne Porter Felt, Erika Chin, Steve Hanna, Dawn Song, and David Wagner. Android permissions demystified. Technical Report UCB/EECS-2011-48, EECS Department, University of California, Berkeley, May 2011.
6. Stéphane Geller. Politique de flux d'information et d'exécution pour un modèle de détection fiable. In *Proceedings of the 6th Conference on Network Architectures and Information Systems Security (SAR-SSI)*, 2011.
7. Stéphane Geller, Christophe Hauser, Frédéric Tronel, and Valérie Viet Triem Tong. Information flow control for intrusion detection derived from mac policy. In *IEEE International Conference on Communications (ICC'11)*, 2011.
8. Google. Android market.
9. Guillaume Hiet, Valérie Viet Triem Tong, Ludovic Me, and Benjamin Morin. Policy-based intrusion detection in web applications by monitoring java information flows. *Int. J. Inf. Comput. Secur.*, 3(3/4) :265–279, 2009.
10. Lookout. Security alert : Hongtoutou, new android trojan, found in china.
11. Lookout. Update : Security alert : Droiddream malware found in official android market.
12. Andrew C. Myers and Barabara Liskov. Complete, safe information flow with decentralized labels. In *IEEE Symposium on Security and Privacy*, 1998.
13. G. Edward Suh, Jae W. Lee, David Zhang, and Srinivas Devadas. Secure program execution via dynamic information flow tracking. *SIGARCH Comput. Archit. News*, 32(5) :85–96, 2004.
14. Timothy Wyatt Timothy Strazzere. Geinimi trojan technical teardown. In *Lookout Mobile Security*, 2011.
15. Valérie Viet Triem Tong, Andrew Clark, and Ludovic Mé. Specifying and enforcing a fined-grained information flow policy : Model and experiments. In *Journal of Wireless Mobile Networks, Ubiquitous Computing and Dependable Applications (JOWUA)*, 2010.
16. Nickolai Zeldovich, Silas Boyd-Wickizer, Eddie Kohler, and David Mazières. Making information flow explicit in histar. In *OSDI*, pages 263–278, 2006.

Etablissement de Clé de Session en Environnement M2M entre Nœuds à Ressources Fortement Hétérogènes

Yosra Ben Saied, Alexis Olivereau

CEA, LIST, Laboratoire des systèmes communicants 91191 Gif-sur-Yvette CEDEX, France
{yosra.ben-saied, alexis.olivereau}(@)cea.fr

Résumé Dans les réseaux M2M qui émergent actuellement, le besoin de faire communiquer différentes entités conduit à s'intéresser aux exigences de sécurité que cette interconnexion demande. Le caractère hétérogène des nœuds M2M impose de nouveaux défis que les solutions de sécurité existantes ne peuvent pas satisfaire. Pour pouvoir communiquer de façon sécurisée, deux entités commencent généralement par échanger une clé de session. Or deux entités M2M peuvent ne pas être en mesure de mener cette étape d'établissement de clé de session, parce qu'elles ne peuvent s'accorder sur des primitives cryptographiques communes. Dans ce papier, nous proposons donc une nouvelle approche pour l'établissement de clé de session entre un capteur à très faibles capacités de calcul, de mémoire et d'énergie, incapable de supporter la cryptographie asymétrique et un serveur distant. Le système proposé exploite la collaboration entre les nœuds hétérogènes en déléguant les opérations cryptographiques asymétriques couteuses à un ensemble de nœuds voisins. Une analyse de sécurité est effectuée pour vérifier que la solution proposée atteint son objectif avec sécurité et efficacité.

Mots-clés: M2M ; établissement de clé de session ; contraintes énergétiques ; efficacité énergétique ; coopération ; confiance.

1 introduction

Les communications machine à machine (M2M) sont considérées comme la prochaine étape dans les communications sans fil. Les réseaux M2M se libèrent des contraintes traditionnelles qui demandent une intervention humaine pour passer à des communications autonomes entre les entités du réseau [1]. Ces communications représentent une évolution par rapport aux réseaux de capteurs traditionnels, qu'elles étendent en les interconnectant directement ou au travers de l'Internet avec des nœuds M2M mobiles et/ou distants.

Considérant ce grand nombre de nœuds M2M communiquant suivant un modèle pair à pair distribué, des mesures de sécurité doivent être prises afin d'assurer une transmission sécurisée de leurs informations sensibles [2]. Les principales exigences relatives à la sécurité concernent la confidentialité, l'authentification et l'intégrité des données. Ces services de sécurité reposent sur l'utilisation des primitives cryptographiques comprenant des schémas de chiffrement / déchiffrement et de signature / vérification. A leur tour, ces primitives requièrent un processus d'établissement de clé de session qui doit s'adapter aux faibles capacités en

ressources des composants M2M, dont certains ne peuvent pas s'appuyer sur ces systèmes de sécurité complexes.

Dans ce contexte d'établissement de clé de session, plusieurs solutions efficaces en termes de ressources ont été proposées pour les réseaux de capteurs. Les solutions qui utilisent des clés symétriques ont suscité le plus d'intérêt en raison de leur faible coût énergétique. Cependant ces solutions sont caractérisées par une forte complexité en gestion des clés. Elles introduisent également d'importantes vulnérabilités dans le cas où des nœuds du réseau M2M seraient compromis. Les solutions utilisant la cryptographie à clé publique ont tout d'abord été jugées inapplicables aux capteurs à cause de leurs ressources limitées. Mais, certaines recherches ont récemment montré qu'il peut néanmoins être envisageable d'utiliser la cryptographie à clé publique, lorsqu'elle met en œuvre des algorithmes utilisant des courbes elliptiques. Ces solutions restent toutefois trop exigeantes au regard des capacités limitées d'un capteur.

Dans le scénario M2M envisagé dans ce papier, les capteurs interagissent directement avec des nœuds M2M mobiles qui n'appartiennent pas au même réseau capteur. Cependant, les nœuds à ressources limitées, incapables de supporter la cryptographie à clé publique, ne sont pas en mesure de communiquer avec un serveur distant puissant qui exigerait ce niveau de sécurité. Les solutions existantes d'établissement de clé de session n'adaptent pas l'utilisation des primitives cryptographies asymétriques comme requis pour assurer un haut niveau de sécurité à des nœuds contraints en ressources. Ainsi, de nouvelles solutions d'établissement de clés sont nécessaires pour établir des communications M2M de manière sécurisée.

Nous proposons dans ce papier une nouvelle approche qui permet à un capteur fortement contraint en termes de ressources d'établir une clé de session avec un serveur M2M externe puissant, sans connaissance mutuelle préalable et en s'appuyant sur des primitives cryptographiques asymétriques. Notre système exploite la collaboration entre des nœuds hétérogènes pour déléguer les opérations cryptographiques asymétriques à un ensemble de nœuds au voisinage du capteur et moins contraints en termes de ressources de ce dernier. Ces nœuds supportent d'une manière distribuée et coopérative les opérations de cryptographie asymétriques requises pour envoyer un secret du capteur au serveur distant. Bien qu'impliquant les nœuds assistant le capteur dans des opérations de cryptographie asymétrique, notre solution s'appuie sur des mécanismes à l'efficacité énergétique démontrée pour réduire et maitriser la charge sur ces nœuds.

Nous démontrons à travers une analyse de sécurité que la confidentialité du secret transmis depuis le capteur jusqu'au serveur distant est garantie même en environnement fortement hostile. D'autre part, la solution en elle-même n'introduit pas de nouvelles vulnérabilités de type attaque par déni de service. La section 2 de ce papier présente l'état de l'art des solutions existantes pour l'établissement

de clé de session. La section 3 décrit notre système coopératif d'établissement de clé proposé pour les réseaux M2M. La section 4 analyse l'efficacité et la sécurité du système proposé. La section 5 conclut par un aperçu sur la nécessité d'un modèle cognitif pour la sélection des nœuds assistant le capteur fortement contraint.

2 Etat de l'Art

Afin de sécuriser une communication entre deux entités au moyen d'un algorithme à cryptographie symétrique, une clé de session secrète doit être partagée entre eux. Dans certains cas, une entité peut générer ou obtenir une clé de session et ensuite la transmettre de manière sécurisée à l'autre entité. Dans d'autres cas, les deux parties peuvent ensemble construire cette clé de session, ou la récupérer à partir d'un tiers de confiance. Dans les réseaux de capteurs sans fil, ce processus d'établissement de clé de session est considéré comme crucial en raison des contraintes imposées par les capacités limitées des capteurs : faible puissance de calcul, capacité d'énergie et de stockage limitée. Dans ce contexte, la littérature propose plusieurs systèmes d'établissement de clés de session mettant l'accent sur l'efficacité énergétique. Dans ce papier, nous nous concentrons sur l'étude des systèmes d'établissement de clés par paire (le scénario des communications pair à pair est le plus fréquent dans l'environnement M2M). Les solutions existantes sont classées en trois catégories principales, décrites ci-dessous.

2.1 Systèmes Basés sur la Cryptographie Symétrique

La plupart des approches existantes pour l'établissement de clé de session dans les réseaux de capteurs reposent sur des primitives cryptographiques symétriques du fait de leur consommation raisonnable de ressources. Ces solutions sont considérées plus efficaces pour les capteurs. Les recherches plus pertinentes dans ce domaine sont décrites dans ce qui suit.

La solution la plus simple consiste à munir chaque capteur de N-1 clés secrètes, chacune de ces clés étant connue uniquement par ce capteur et un des N-1 autres capteurs (N étant le nombre total des capteurs dans le réseau). Cependant, ce schéma n'est pas applicable concrètement pour des capteurs avec une capacité de stockage très limitée, car N pourrait être important. D'autre part l'ajout de nouveaux capteurs est difficile parce que les capteurs présents ne partagent pas de clé avec ces nouveaux capteurs. Eshenauer et Gligor proposent dans [3] un schéma de pré-distribution de clés purement probabiliste, dans lequel chaque nœud est muni avant le déploiement d'un sous-ensemble aléatoire de clés. L'idée de ce schéma est que chaque paire de nœuds partage au moins une clé commune qui appartient aux deux sous-ensembles de ces deux nœuds. Chan et al. proposent dans [4] l'approche q-composite pour améliorer la résilience du schéma Eschenauer-Gligor

contre les nœuds compromis. La différence est que q clés communes (au lieu d'une seule) sont nécessaires pour établir un lien de communication sécurisé dans une paire de nœuds.

D'autres approches [5] et [6] proposent des schémas de pré-distribution de clés s'appuyant sur une connaissance du déploiement de chaque nœud, en vue d'accroitre la probabilité de partage de clés. Les clés sont attribuées aux nœuds en fonction de leur position géographique. Ces solutions ne sont pas applicables dans les réseaux de capteurs déployés de manière aléatoire.

Le travail dans [7] propose un schéma de pré-distribution de clés par paire, dans lequel un part de polynôme est distribuée à chaque nœud. En l'utilisant, deux nœuds sont capables d'établir une clé symétrique.

Perrig et al. présentent dans [8] un protocole gestion de clés (SPINS) qui s'appuie sur une station de base de confiance pour distribuer les clés. Deux capteurs utilisent la station de base comme un tiers de confiance pour mettre en place leur clé secrète. SPINS comprend deux parties : SNEP (Secure Network Encryption Protocol) qui sécurise les communications entre un nœud et la station de base ou entre deux nœuds, et $\mu/TESLA$ (μTime Efficient Streaming Loss-tolerant Authentication) qui authentifie les paquets envoyés par la station de base.

Les systèmes d'établissement de clé de session basés sur la cryptographie symétrique sont très abordés puisqu'ils sont les plus efficaces en termes de consommation de ressources, et donc les plus appropriés pour les réseaux de capteurs contraints en ressources. Toutefois, le passage à l'échelle et la complexité de la gestion de clés restent des problèmes considérables que l'on ne peut pas négliger. On considérera, par exemple, un large réseau de capteurs dont la mise en œuvre nécessiterait la génération d'un grand nombre de clés partagées et leur installation dans les capteurs avant déploiement – parfois en tenant compte des futures positions respectives de ces capteurs.

En outre, ces schémas n'offrent pas un bon niveau de sécurité et sont particulièrement vulnérables à la compromission des nœuds. De plus, les algorithmes symétriques ne permettent pas d'offrir les services d'authentification et d'intégrité des données puisque les codes d'authentification de message (MAC, Message Authentication Code) ne sont pas publiquement vérifiables. Afin de vérifier l'intégrité d'un message, le récepteur a besoin d'avoir une clé partagée avec l'émetteur. D'autre part et dans le contexte des scénarios M2M, ces schémas basés sur la cryptographie symétrique ne sont pas applicable si un capteur doit communiquer avec des entités externes.

2.2 Systèmes Basés sur la Cryptographie Asymétrique

La cryptographie à clé publique est elle aussi utilisée pour établir des clés de session symétriques. Les services de sécurité (par exemple la non-répudiation) et

le niveau de protection (par exemple la résistance à la compromission de nœuds) qu'elle offre sont plus évolués que ceux offerts par la cryptographie symétrique. Pour cette raison, plusieurs travaux ont considéré l'application de la cryptographie asymétrique dans les réseaux de capteurs afin de fournir le meilleur compromis entre les services de sécurité et les exigences en termes de puissance de calcul et de mémoire. Les études [9], [10] et [11] ont montré que l'utilisation de la cryptographie asymétrique par des capteurs à ressources limitées est viable.

Lopez dans [12] met en évidence les limites de l'utilisation des solutions basées sur la cryptographie symétrique dans les réseaux de capteurs et favorise celles basées sur la cryptographie à clé publique pour améliorer la sécurité de l'ensemble du système, tout en mettant en garde contre le coût élevé de ces algorithmes asymétriques. L'auteur souligne également le rôle important que la cryptographie sur les courbes elliptiques (Elliptic Curve Cryptography, ECC) peut jouer pour surmonter la complexité de calcul des algorithmes à clé publique. Ces algorithmes utilisant des courbes elliptiques ont été privilégiés en raison de leur faible cout énergétique, du temps de traitement rapide, des signatures compactes générées, et de la taille réduite des clés. Par exemple, une clé ECC de 163 bits garantit un niveau de protection équivalent à une clé RSA de 1024 bits, avec un coût énergétique réduit de moitié [13]. Les auteurs de [14] et [15] proposent des systèmes légers d'établissement de clés de session reposant sur les courbes elliptiques ; ils affirment que l'utilisation de l'ECC est le meilleur compromis entre la consommation énergétique et le niveau de sécurité.

Parallèlement à ces travaux basés sur l'ECC, [16] et [17] se concentrent sur l'algorithme asymétrique RSA [18] afin de l'adapter aux nœuds contraints en termes de ressources en réduisant l'exposant (e) et la taille de la clé. Ainsi, Watro et al. dans [17] développent le système TinyPK qui permet l'implémentation de la cryptographie asymétrique dans les réseaux de capteurs. Le concept requiert l'utilisation des paramètres réduits de l'algorithme RSA (exposant, taille de la clé) et l'utilisation des seules opérations publiques (chiffrement et vérification de la signature) au niveau du capteur. Cependant, un tel avantage s'acquiert aux dépens du niveau de sécurité [19].

D'autres travaux [19] [20] [21] ont proposé des solutions matérielles qui étendent les capacités de calcul d'un nœud standard au moyen de modules de calcul dédiés. Les résultats montrent que ces systèmes matériels peuvent offrir un bon niveau de sécurité tout en diminuant la consommation énergétique.

L'utilisation de la cryptographie à clé publique (Public Key Cryptography, PKC) résout les problèmes du passage à l'échelle et de la complexité de la gestion de clés. Elle augmente au même temps la sécurité de l'ensemble du système, puisque la compromission d'un nœud ne révèle pas les clés des nœuds non compromis. Elle permet également la communication avec un nœud distant avec lequel aucun contexte de sécurité prédéfini n'existe. Cependant, les avantages de l'uti-

lisation de la cryptographie asymétrique s'effectuent au détriment d'une grande consommation de ressources, puisque les opérations à clé publique sont beaucoup plus couteuses que celles basées sur des clés symétriques. Pour rendre possible l'utilisation de la PKC, des versions légères des algorithmes asymétriques ont été proposés dans les réseaux de capteurs tels que décrits ci-dessus. En pratique, la consommation énergétique de ces solutions demeure toujours non négligeable [20] ce qui s'avère entravant pour les capteurs fortement contraints en ressources.

2.3 Systèmes Hybrides

D'autres schémas d'établissement de clés de session qui combinent à la fois les opérations cryptographiques symétriques et asymétriques ont été proposés. Ce concept hybride a pour but de réduire le coût élevé de la cryptographie asymétrique en remplaçant certaines opérations par d'autres, basées sur des clés symétriques, pour ainsi associer les avantages des deux approches.

Huang et al. [22] et Kotzanikolaou et al. [23] proposent des protocoles hybrides, qui combinent le standard Elliptic Curve Diffie Hellman (ECDH) et des certificats implicites avec des techniques symétriques pour réduire le cout élevé des multiplications scalaires des points des courbes elliptiques. Le travail [22] exploite le modèle hiérarchique de certains réseaux de capteurs et propose que certains nœuds puissants prennent en charge le maximum de calcul cryptographique. Kotzanikolaou et al. [24] étend le protocole hybride d'établissement de clé proposé dans [22] pour supporter le déploiement multi-phases dans les réseaux de capteurs et maintenir le rôle des nœuds puissants. Le travail [25] corrige une faille de sécurité dans [24] qui permet à un adversaire de prendre connaissance des communications d'un nœud compromis ayant eu lieu dans le passé, tout en assurant des coûts de calcul et de communication similaires.

Mache et al. développent dans [26] un framework de sécurité pour les capteurs contraints en terme de ressources qui exploite l'hétérogénéité dans les réseaux de capteurs et se base sur une combinaison à la fois d'opérations symétriques et asymétriques. L'idée est d'utiliser de la cryptographie symétrique sur la première partie du chemin du capteur vers le nœud destinataire jusqu'à ce qu'une passerelle riche en ressources soit atteinte. A partir de cette passerelle, il sera ensuite possible de passer à la cryptographie asymétrique.

Riaz et al. proposent dans [27] trois schémas d'établissement de clés : SACK qui se base sur la cryptographie symétrique, SACK-P qui se base sur la cryptographie asymétrique et SACK-H, qui s'appuie sur une approche hybride utilisant la cryptographie asymétrique pour les communications intra-cluster et la cryptographie symétrique pour les communications inter-cluster. L'auteur mène ensuite une comparaison entre les trois schémas proposés et montre que SACK consomme le moins de ressources, mais fournit un niveau de sécurité plus faible,

vu que la compromission d'un seul nœud rend vulnérable l'ensemble du réseau. En revanche, SACK P consomme trop de ressources mais offre le niveau de sécurité le plus haut, avec une résilience maximale à la compromission des nœuds. Le schéma hybride SACK-H associe les avantages de ces deux approches en offrant une consommation de ressources et un niveau de sécurité moyens.

En résumé, les solutions hybrides telles qu'elles sont proposées ci-dessus sont soit basées sur une passerelle intermédiaire qui assure l'interopérabilité entre les deux domaines symétriques et asymétriques, soit construits sur l'idée de remplacer certaines opérations dans un algorithme asymétrique par des opérations symétriques. Dans le premier cas, la sécurité est assurée en hop-by-hop. La confidentialité et la disponibilité du système se trouvent de ce fait compromises, vu que la passerelle intermédiaire présente potentiellement à la fois un point unique de défaillance et une faille de sécurité. Dans le second cas, les communications avec des nœuds externes requises par les scénarios M2M deviennent moins réalisables, puisque les deux entités voulant établir une connexion sécurisée sont censées partager une clé symétrique.

3 Solution Proposée

Considérant les insuffisances des trois approches décrites ci-dessus, nous proposons un nouveau système d'établissement de clé de session adapté aux scénarios M2M en permettant d'établir de manière sécurisée une communication de bout en bout entre des nœuds M2M de capacités différentes en termes de ressources.

3.1 Modèle du Réseau

Nous considérons une infrastructure M2M hétérogène se composant de nœuds de capacités distinctes en termes à la fois de puissance de calcul et de ressources énergétiques. Nous distinguons en particulier trois catégories différentes de nœuds :

- Des capteurs fortement contraints en ressources, incapables de supporter la cryptographie à clé publique.
- D'autres capteurs, moins contraints, en mesure d'utiliser uniquement les opérations publiques de la cryptographie asymétrique. Ces opérations sont celles qui sont effectuées avec la clé publique, à savoir le chiffrement ou la vérification de signature. Elles ont été évaluées dans [28] comme étant beaucoup moins exigeantes en termes de ressources que celles utilisant la clé privée, dans le cryptosystème RSA.
- Des nœuds M2M avec de grandes capacités en termes de ressources (par exemple, des serveurs distants reliés à forte puissance de calcul et alimentés en énergie par une source extérieure).

Contrairement aux réseaux de capteurs classiques mais en accord avec les scénarios M2M émergeants, nous considérons dans ce papier que des communications directes entre un capteur et un serveur distant peuvent avoir lieu, sur une base de bout en bout. Ainsi, un protocole d'établissement de clé de session entre ces deux entités est nécessaire puisqu'elles vont échanger des données sensibles sans connaissance préalable l'une de l'autre.

3.2 Hypothèses

La solution développée dans cette section s'appuie sur les hypothèses suivantes :

- Après la phase d'initialisation, chaque capteur partage une clé pair à pair avec un sous-ensemble de ses voisins. Ces clés peuvent être générées durant une phase de bootstrapping à l'aide d'une entité de confiance pour la gestion des clés, ou au moyen de mécanismes plus subtils tels que le transitive imprinting.
- Les capteurs fortement contraints en ressources sont capables d'identifier un ensemble de capteurs moins contraints qui seront disponibles pour supporter des opérations cryptographiques couteuses à leur place.
- Il existe une entité de confiance au sein du réseau de capteurs qui détient un secret partagé avec tous les nœuds dans le réseau de capteurs, ainsi qu'une paire de clés publique / privée.
- Les nœuds M2M puissants supportent la cryptographie à clé publique et reposent sur des primitives cryptographiques asymétriques (algorithme RSA) pour établir une clé de session avec leurs correspondants.
- Les nœuds M2M puissants ne peuvent pas communiquer avec l'entité de confiance située dans le réseau de capteurs mais sont statiquement configurés avec ou en mesure de valider sa clé publique.
- Les nœuds M2M puissants font confiance à l'entité de confiance pour s'assurer qu'un capteur a le droit de signer au nom de l'autre.

3.3 Notations

Nous utilisons les notations suivantes pour décrire les échanges du protocole et les opérations cryptographiques dans ce document :

3.4 Aperçu de la Solution Proposée

Notre solution est conçue pour fournir un mécanisme d'établissement de clés de session efficace en ressources afin de rendre possible, pour un nœud fortement contraint en ressources, l'établissement d'une clé de session avec une entité externe de manière sécurisée et ce, à l'aide de primitives cryptographiques asymétriques.

Notation	Description
A	Capteur
B	Serveur extérieur
P_i	Nœud assistant
T	Entité de confiance locale dans un réseau de capteurs
N_X	Valeur aléatoire (nonce) généré par un nœud X
ID_i	Identifiant de P_i
K_{XY}	Clé paire à paire entre X et Y
${K_X}^{-1}$	Clé privée du nœud X
K_X	Clé publique du nœud X
${LK_i}^{-1}$	Clé privée de type Lamport de P_i
LK_i	Clé publique de type Lamport de P_i
$\underline{sk_i}$	Germe aléatoire utilisé pour générer des paires de clés privés/ publiques de type Lamport pour P_i
MT_{Root}	Racine de l'arbre de Merkle
MT_{Pathi}	Chemin d'authentification de la clé publique de P_i dans l'arbre de Merkle
M_i	Partie du modèle initial de clé (premaster key) M traitée par Pi

Table 1. Notations.

Comme le capteur est supposé être incapable de supporter la cryptographie à clé publique, nous proposons que toutes les opérations cryptographiques soient déléguées à des nœuds voisins moins contraints en ressources. Ces nœuds prennent en charge les parties couteuses du calcul requis pour envoyer la clé de session, d'une manière distribuée et coopérative. Chacun de ces nœuds assistants (ou proxies, P_i) est chargé de chiffrer une partie du secret partagé envoyé du capteur fortement contraint en ressources A au serveur distant B en utilisant la clé publique de B. Il est également nécessaire que chaque nœud assistant prouve à B qu'il est réellement un nœud légitime, autorisé par A à agir en son nom.

Une fois que le capteur A a décidé d'établir un secret partagé avec un serveur distant B, le processus de notre solution met en œuvre les étapes suivantes :

- Sélection par A des nœuds assistants P_i qui seront utilisés pour effectuer l'établissement de clé avec B.
- Récupération par les nœuds assistants des clés qui leur seront nécessaires pour signer au nom de A.
- Etablissement d'une connexion sécurisée entre les nœuds assistants et B.
- Préparation et segmentation du secret partagé dans A, répartition des différents segments de la clé aux différents proxies et transport sécurisé des fragments de chaque proxy vers B.
- Validation par B des différents messages reçus et réassemblage, afin de récupérer le modèle initial du secret (pre master key) et pouvoir calculer la clé maîtresse.
- Vérification de la clé maîtresse.

3.5 Outils Techniques

Signatures à usage unique (one-time signatures).

Une signature à usage unique est une signature digitale basée sur des fonctions de hachage à sens unique. Des valeurs privées aléatoires sont générées pour correspondre aux différentes parties d'un message à signer. Pour chaque valeur privée, une valeur publique est calculée en se basant sur les résultats de fonctions de hachage.

Ce système de signature est particulièrement léger et efficace en termes de ressources comparé à d'autres alternatives [29]. Deux inconvénients pourraient éventuellement atténuer son applicabilité : d'une part, une paire de clés publique / privée ne peut être utilisé qu'une seule fois, puisque la clé privée est nécessairement divulguée avec la signature. D'autre part, une longue clé sera nécessaire pour signer un long message puisque la clé privée (resp. publique) est la concaténation de toutes les valeurs privées (resp. publiques), qui sont aussi nombreuses que les blocs d'un message. Néanmoins, aucun de ces défauts n'affecte notre solution, qui considère des échanges de messages courts effectués en une seule fois.

Dans notre solution, nous avons retenu l'algorithme de one-time signature HORS [30] de Reyzin & Reyzin. Cet algorithme convient pour être utilisé par les nœuds assistants parce qu'il est considéré comme le plus rapide et le moins couteux en termes de ressources [29] parmi les systèmes de signature à usage unique.

Arbre de Merkle.

Un arbre de Merkle est une structure dont l'intérêt est de permettre d'authentifier un grand nombre de valeurs sans avoir besoin de les signer individuellement, mais plutôt de les authentifier globalement. Le principe en est le suivant : les valeurs à authentifier sont placées dans les feuilles d'un arbre binaire. La valeur correspondant à un nœud parent est calculée à partir des valeurs de ses deux enfants, au moyen d'une fonction de hachage à sens unique. De la sorte, toutes les valeurs interviennent dans le calcul de la valeur du nœud racine. Ainsi, il est suffisant d'authentifier cette seule valeur racine pour pouvoir authentifier les valeurs de toutes les feuilles. Une feuille peut ensuite être vérifiée individuellement en connaissant de la valeur de la racine et le chemin d'authentification de la feuille, ce dernier étant défini comme les valeurs successives nécessaires pour calculer la valeur de la racine à partir de la feuille considérée.

L'utilisation traditionnelle de l'arbre de Merkle en rapport avec les signatures à usage unique consiste à authentifier un grand nombre de clés publiques, correspondant à plusieurs signatures à usage unique provenant du même nœud (puisque chaque message doit être signé avec une clé différente). Contrairement à cet usage, nous proposons dans notre solution l'utilisation de l'arbre de Merkle

pour authentifier un grand nombre de clés publiques correspondant à plusieurs signatures effectuées en parallèle et provenant de différents nœuds.

Système de correction d'erreurs.

Le principe de la correction d'erreurs consiste à ajouter de la redondance au message original, divisé en plusieurs paquets, afin que ce dernier puisse être récupéré par le récepteur, même si certains paquets ont été modifiés ou perdus au cours de la transmission. Soit u le nombre total de blocs envoyés, v ($v < u$) est le nombre minimum de blocs nécessaires pour reconstituer le message original.

Dans notre solution, le schéma de correction d'erreurs sera utilisé par l'émetteur A pendant la phase de partitionnement de la clé de session entre les différents nœuds assistants P_i. Ainsi, le récepteur B sera en mesure de reconstituer la clé de session à condition qu'un nombre suffisant de paquets soient reçus de la part des nœuds assistants, sans requérir la réception de l'ensemble des messages provenant de la totalité des nœuds assistants. Ce système protège notre solution contre la compromission ou le refus de collaborer des nœuds assistants. En outre, il permet de ne pas imposer une livraison fiable dans chaque connexion au serveur proxy. Dans notre solution, nous avons choisi de nous appuyer sur le code de Reed-Solomon, qui est un code correcteur d'erreur largement utilisé basé sur les corps de Galois et qui consiste à générer un polynôme formel à partir des symboles à transmettre et à le suréchantillonner. La redondance introduite par ce suréchantillonnage permet au récepteur de reconstituer le message original même s'il y a eu des erreurs ou des pertes pendant la transmission.

Mobile IP.

Le protocole Mobile IP est un standard proposé par l'IETF qui permet à un nœud de changer son point d'attachement dans l'Internet sans avoir besoin de changer son adresse IP. Mobile IP permet une connectivité continue au niveau application alors que le nœud mobile se déplace de point en point.

La solution développée par l'IETF implique une extension du protocole IP selon laquelle les paquets destinés à un hôte mobile sont envoyés à son réseau mère et interceptés par un nœud statique (non mobile) appelé agent mère de l'hôte mobile. L'hôte mobile enregistre son emplacement réel au niveau de l'agent mère, qui est chargé de lui transmettre les paquets. Ces paquets doivent être tunnelés à travers l'Internet jusqu'au point d'attachement réel de l'hôte dans le nouveau réseau.

Ce processus de tunneling n'est nécessaire que dans un seul sens. Les paquets envoyés par l'hôte mobile peuvent, quant à eux, être acheminé par le réseau IP en utilisant les procédures standard en indiquant la nouvelle adresse de l'hôte comme adresse source. Toutefois, il est également possible que l'agent mère réceptionne les paquets sortants de l'hôte mobile et les délivre au correspondant (reverse tunneling). Dans notre solution, Mobile IP peut être utilisé pour permettre la

mobilité des nœuds P_i tout en gardant une connectivité ininterrompue avec ceux-ci lors du processus d'établissement de clé de session.

3.6 Description des Echanges par Phases

En phase d'initialisation, A sélectionne soigneusement les proxies $P_1...P_n$ qui supporteront son processus d'établissement de clé de session en se basant sur leurs réputations et leurs capacités en termes de ressources. Dans le cas où A n'est pas statiquement muni d'une liste de proxies utilisable (et, en particulier, dans le cas où A est mobile), A doit ajouter comme pré-requis au processus de sélection la découverte à son voisinage de nœuds avec lesquels il peut disposer d'un secret partagé. Cette découverte peut s'effectuer dynamiquement, ou peut être facilitée par le recours à une entité connaissant l'intégralité de la topologie M2M dans laquelle A évolue.

Un autre élément important pour A consiste à s'assurer que les proxies identifiés restent joignables tout au long de la phase d'établissement de clé. Certes, le mécanisme de correction d'erreur utilisé permet de supporter la défaillance d'un faible nombre de proxies, mais il convient néanmoins de prendre en compte leur mobilité éventuelle dans le processus de sélection. A cette fin, A peut être renseigné par les proxies sur le type de mobilité attendu (s'il leur est connu), ou sur leur mécanisme éventuel de support de la mobilité, tel que Mobile IP. A prend ensuite en considération ces informations dans son choix des proxies P_i auxquels il recourra par la suite.

Les phases ultérieures qui composent notre proposition sont illustrées dans le tableau ci-dessous, et expliquées plus bas dans cette section.

Phase 1 : échange initial des informations relatives à la sécurité supportée par A et B.

Cette phase est initiée par le capteur A qui envoie le message A-hello (1) pour informer B des algorithmes cryptographiques et de la technique coopérative d'établissement de clé qu'il supporte. Si B est d'accord avec la technique coopérative proposée, il répond avec un message B-hello (2) en choisissant l'algorithme cryptographique parmi ceux qui sont proposés par A. Les deux messages comprennent également des valeurs aléatoires utilisées comme nonces pour prévenir les attaques par rejeu et calculer la clé de session.

Phase 2 : Préparation des entités impliquées dans le processus d'établissement de clé de session.

Après avoir réussi la connexion initiale avec B, A commence cette deuxième phase par l'envoi d'un message commun (3) à tous les proxies afin de leur apprendre l'identité du serveur distant B. A noter qu'un proxy peut refuser explicitement de participer à l'établissement de clé entre A et B, au moyen d'un NAK (non indiqué dans le tableau ci-dessus). En particulier, un tel message NAK peut

Phase 1	1. $A \xrightarrow{A-Hello\ (param\ êtres\ cryptographiques\ ,N_A)} B$ 2. $B \xrightarrow{B-Hello\ (param\ êtres\ cryptographiques\ ,N_B)} A$
Phase 2	3. $\forall i \in [1;n]$ $A \xrightarrow{B,N_A,N_B} P_i$ 4. $A \xrightarrow{Hello\ (N_A)} T$ 5. $T \xrightarrow{Hello\ (N_T)} A$ 6. $A \xrightarrow{\{s,P_1,...,P_n,N_T\}_{k_{AT}}} T$
Phase 3	7. $\forall i \in [1;q]$ (q étant le nombre de proxies participant) $P_i \xrightarrow{Hello\ (N_{P_i})} T$ 8. $T \xrightarrow{\left\{\underline{sk_i},MT_{Path\ i},N_{P_i},[R-P,A,MT_{Root}]_{K_T^{-1}}\right\}_{k_{P_iT}}} P_i$ 9. $T \xrightarrow{\{(P_1,ID_1,N_{P_1}),...,(P_n,ID_n,N_{P_n}),N_A\}_{k_{AT}}} A$
Phase 4	10. $P_i \xrightarrow{Demande\ de\ certificat\ ,N_B,pk_i,MT_{Path\ i},[ID_1,...,ID_n,A,MT_{Root}]_{K_T^{-1}}} B$ 11. $B \xrightarrow{cert_B,[N_A]_{K_B^{-1}}} P_i$ 11'. $P_i \xrightarrow{cert_B} A$
Phase 5	12. $A \xrightarrow{\{N_{P_i},M_i\}_{K_{AP_i}}} P_i$ 13. $\forall i \in [1;m]$ (m étant le nombre de messages 11' reçus) $P_i \xrightarrow{[M_i]_{K_B},[[M_i]_{K_B},N_B]_{LK_i^{-1}}} B$
Phase 6	14. $B \xrightarrow{\{ID_i\}_{MS}} A$

Table 2. Phases de la solution proposée pour l'établissement de clé de session.

être envoyé si le proxy identifié détermine qu'il ne sera pas à même de maintenir la connexion avec A pendant un temps suffisamment long (par exemple, du fait d'une mobilité attendue et de l'absence d'un mécanisme de support de la mobilité).

Comme mentionné plus haut, en raison de leurs contraintes en termes de ressources, les proxies supportent seulement les opérations publiques du RSA, c'est-à-dire le chiffrement et la vérification des signatures avec la clé publique de B. Cependant, notre solution suppose que ces nœuds assistants doivent contacter B et lui envoyer des messages au nom de A. Ainsi des problèmes d'authentification et d'autorisation se posent du côté des proxies, puisque d'une part ces nœuds doivent non seulement prouver l'intégrité des messages transmis mais aussi leur

représentativité de A. Pour cela, nous proposons d'attribuer une paire de clés privée/ publique éphémères à chaque nœud assistant, en se basant sur les schémas de signature à usage unique expliqué dans la section 1) afin que le proxy puisse signer des messages au nom de A. La charge de calcul due à la génération des paires de clés est déplacée de ces proxies à une entité de confiance T, qui est aussi la seule entité capable d'affirmer qu'un proxy est autorisé à signer au nom de A. Pour fournir les paires de clés éphémères à tous les proxies, A commence par établir une connexion avec l'entité de confiance T à travers les messages (4), (5) et lui apprend dans (6) les identités des nœuds assistants ainsi que la taille des données qui seront transmises à B (puisque, dans le schéma de signature à usage unique, les longueurs des clés publiques et privées dépendent de la taille du message à signer).

A noter que T représente une entité "logique", qui peut être physiquement instanciée sous la forme de différentes entités physiques mettant en œuvre un système de redondance de manière à assurer une continuité de service en cas de crash ou de mobilité d'une de ces entités. Bien sûr, le caractère "stateful" du protocole proposé du côté de T doit être pris en compte dans ce système de redondance.

Phase 3 : Remise des clés éphémères.

Après avoir reçu le message (6), T génère une paire de clés privée / publique $({LK_i}^{-1}, LK_i)$ et un identifiant ID_i pour chaque proxy P_i. L'identifiant ID_i est calculé comme suit : $ID_i = H(LK_i)$ avec $H()$ étant une fonction de hachage à sens unique, et ce afin d'avoir une taille pratique à échanger entre les nœuds. Après avoir reçu le message Hello (7) envoyé par le proxy P_i en réponse à un message (3), T remet à P_i les clés nécessaires (une clé privée à usage unique associée à une clé publique) pour signer au nom de A à travers le message (8). En fait, les clés publiques envoyées par T à tous les proxies ne sont pas individuellement signées étant donné que la vérification de chaque signature serait coûteuse pour B. Les clés publiques sont toutes regroupées dans un arbre de Merkle (comme expliqué ci-dessus), dont seule la racine est signée par T. Une analyse détaillée de la structure des messages (8) montre les paramètres suivants :

- $\underline{sk_i}$ est la graine aléatoire utilisé par T pour générer la paire de clés privée/-publique $({LK_i}^{-1}, LK_i)$ du proxy P_i. La transmission de $\underline{sk_i}$ seulement est plus efficace en termes de bande passante que la transmission de l'ensemble des valeurs privées et publiques.
- MT_{path_i} est le chemin d'authentification de la clé publique du proxy P_i dans l'arbre de Merkle tree à n clés publiques.

T envoie également dans le message (8) une portion signée à transmettre ensuite à B, contenant les paramètres suivants :

- MT_{root} est la racine de l'arbre de Merkle à n clés publiques.

- R-P (Paramètres de reconstitution) indique le nombre de proxies qui sont envisagés pour participer au processus d'établissement de clé de session, ainsi que le nombre minimal de proxies assistants nécessaires pour retrouver le message original.
- A est l'identité du capteur, qui informe ainsi B de l'identité du nœud obtenant l'assistance du proxy P_i.
- N_{P_i} est le nonce correspondant au proxy P_i Il est utilisé afin de fournir une protection contre les attaques par rejeu durant la communication de T à P_i.

A ce stade, il faut noter que certains proxies peuvent ne pas envoyer le message Hello (7) à T.

Ces nœuds peuvent ne pas avoir suffisamment de ressources ou peuvent être réticents à participer au processus de collaboration pour d'autres raisons. Par conséquent, nous considérons que seuls p nœuds légitimes ($p < n$) ont envoyé le message Hello (7) à T, exprimant ainsi leur volonté de participer au processus de collaboration.

- Pour chacun de ces p proxies, T envoie le triplet (P_i, ID_i, N_{P_i}) à A dans le message (9). A la fin du processus d'établissement de la clé de session, A recevra de B la liste des identifiants correspondant aux proxies qui ont transmis leur fragment de clé, conformément au processus collaboratif. A utilisera alors l'information contenue dans ces triplets ainsi que la liste des identifiants envoyée par B pour constituer une liste finale des proxies qui lui auront effectivement porté assistance.

Phase 4 : Etablissement des connexions sécurisées entre B et les proxies.

Après avoir reçu les clés nécessaires, chaque proxy P_i contacte B via le message (10) pour lui demander son certificat et lui fournir sa propre clé publique à usage unique LK_i. En réponse à ce message, B vérifie que le nœud assistant est autorisé à prendre part au processus d'établissement de clé de session au nom du A et qu'il fournit une clé publique valide. B répond alors à la demande de certificat dans le message (11). Chaque proxy vérifie la validité du certificat de B. Optionnellement, à l'issue de cette phase, tous les q proxies ($q < p$) qui sont parvenus à contacter B peuvent transmettre le certificat de B à A, dans un message (11 ') afin d'informer A de leur connexion réussie avec B

Phase 5 : Préparation et remise du modèle initial de secret.

Cette phase suit la connexion réussie entre B et les nœuds assistants. Elle assure la livraison effective du modèle initial de secret (premaster secret) M utilisé plus tard par A et B afin de calculer la clé maitresse partagée. A applique le schéma de correction d'erreurs sur le message original M, le décompose en q fragments $M_i, ..., M_i$ et envoie ensuite chaque fragment M_i au proxy P_i qui lui

correspond, associé au nonce N_{P_i} (afin d'empêcher les attaques par rejeu) dans le message (12).

À la réception du message M_i, le proxy P_i chiffre le fragment en utilisant la clé publique de B et signe le résultat en utilisant sa clé privée à usage unique dans le message (13). À son tour, B vérifie l'intégrité du message reçu en utilisant la clé publique de P_i puis déchiffre le fragment M_i.

Après avoir reçu un nombre suffisant de fragments M_i, B peut retrouver le message original et obtenir le modèle initial de secret M. Enfin, le secret maître est calculé par les deux entités A et B comme la fonction de hachage du modèle initial de secret et des deux nonces N_A et N_B échangés au début de la procédure.

Phase 6 : Vérification du secret maître.

Cette phase termine la procédure d'établissement de clé de session. Le message final (14) authentifié par le secret maître, assure à la fois que B a obtenu le bon secret maître et fournit la liste des identifiants des nœuds qui ont participé à l'échange. En utilisant cette liste ainsi que les identifiants des proxies envoyés par T dans le message (9), A peut savoir si un proxy initialement choisi a participé ou non à la transmission de son fragment de clé. Cette information peut être utilisée pour affiner le modèle de confiance, pour une meilleure sélection future des nœuds assistants.

4 Analyse de Sécurité

4.1 Compromission de la Clé

La solution est basée sur de multiples envois de fragments de la clé secrète en "hop-by-hop", chaque fragment étant envoyé à travers un proxy qui peut donc y avoir accès "en clair". Ainsi, un premier point sur lequel il convient de porter notre attention dans cette partie consiste à étudier l'impact des proxies malveillants sur la confidentialité de la clé de session à transmettre, afin d'assurer un bon niveau de sécurité même dans le cas de leur présence. La sélection de plusieurs noeuds assistants au niveau de A représente la principale protection contre la compromission de la clé, car un seul proxy aura accès à seulement une partie du secret - plus le nombre de proxies est grand, plus le fragment divulgué à chaque proxy est petit. Néanmoins, il faut également faire l'hypothèse que les proxies malveillants peuvent collaborer en partageant leurs différents fragments secrets reçus de A, afin de reconstituer la clé de session. Pour contrer cette menace potentielle d'attaque par collusion, on propose de se baser la sélection des nœuds qui supporteront la transmission de la clé secrète au nom de A sur un modèle de confiance. Ce modèle repose sur un système qui permet de suivre le comportement des nœuds dans le réseau et prend en compte les expériences passées, afin de pouvoir sélectionner des nœuds se comportant bien et d'exclure les nœuds malveillants du processus d'établissement de la clé de session.

Toutefois, une mauvaise identification des proxies est toujours possible et, de ce fait, l'utilisation d'un modèle de confiance ne fait que réduire la menace de collusion. Un second mécanisme permettant de se prémunir contre la collaboration de nœuds malveillants au cours du mécanisme coopératif de transmission de clé secrète consiste à ce que A conserve un petit fragment du modèle initial de secret et le transmette plus tard à B, chiffré par la clé publique de B. La charge de cette opération de chiffrement reste réduite compte tenu de la petite taille du fragment gardé. En même temps, cela permet d'augmenter considérablement la difficulté pour un attaquant de récupérer la clé, à moins bien sûr que la plupart des nœuds assistants ne soit malveillants. Si A s'appuie sur 6 proxies, dont chacun prend en charge la transmission d'un fragment de 96 bits pour un modèle initial de secret de 512 bits (donc un proxy peut échouer sans conséquences) et que A conserve les 32 bits restants, une collusion comptant jusqu'à 4 proxies malveillants ne gagnerait aucune information sur une clé de session de 128 bits obtenue comme résultat d'une fonction de hachage appliquée au secret et aux nonces.

Les attaques Sybil, où un unique nœud malicieux est capable de revendiquer l'identité de plusieurs nœuds, peuvent affecter notre solution dans le cas ou un proxy unique se fait passer pour plusieurs proxies, afin de récupérer plus facilement de multiples fragments envoyés par A. Cette attaque est à gérer par l'entité de confiance, qui ne devrait pas établir différents contextes sécurisés avec des instanciations différentes d'un même nœud physique.

Un deuxième point d'intérêt pertinent pour cette analyse de sécurité consiste à valider la sécurité de chaque remise d'un fragment de la clé de A à B à travers un proxy P_i. La connexion $A \Leftrightarrow P_i$ est garantie par un contexte préétabli et ne peut donc pas souffrir de vulnérabilités spécifiques à notre solution. En revanche, la connexion de P_i à B est créée dynamiquement sans contexte préétabli entre les nœuds y prenant part. De ce fait, notre solution est conçue pour protéger les échanges entre P_i et B contre l'usurpation et les attaques man-in-the-middle : P_i reçoit en toute sécurité l'identité de B depuis A dans le message (3), et valide la clé publique certifiée appartenant à B dans le message (11). B identifie P_i dans le message (13) grâce à la preuve de la possession de la clé privé à usage unique, dont la clé publique correspondante est certifiée par T (auquel B fait confiance).

4.2 Attaques par Déni de Service

Les attaques par Déni de Service (DoS) contre notre solution peuvent être basées sur des attaques par rejeu. Une attaque par rejeu se produit quand un attaquant intercepte une séquence de messages valides entre deux entités et tente de la rejouer à des fins malveillantes. Dans notre cas, plusieurs messages échangés entre les différentes parties pourraient être rejoués. Prenons le message (6) dans lequel A envoie à T la liste des nœuds assistants sélectionnés pour appuyer son

mécanisme d'établissement de clé. Il est clair que ce message peut être rejoué ; un attaquant pourrait ainsi personnifier A et envoyer à T une liste obsolète de proxies. Le serveur T accepterait alors le message rejoué et considérerait son contenu comme authentique. Cela signifie que T commencerait à consommer des ressources pour générer les paires de clés à usage unique pour la liste rejouée des proxies. Le moyen de protéger ce message et d'autres échanges de notre protocole contre cette attaque par rejeu implique l'utilisation des nonces. Un nonce est d'abord envoyé par le destinataire avant que ne débute l'échange de données et est ensuite inclus par l'expéditeur avec les données qu'il envoie, comme on peut le voir dans les paires de messages (5) & (6) et (7) & (8). Ce nonce est transmis de façon sécurisée, en utilisant la clé privée de l'expéditeur ou la clé partagée avec le récepteur. Cela garantit l'authenticité et l'intégrité du nonce transporté et empêche toute tentative d'attaque par rejeu.

Un autre type d'attaques DoS contre notre solution consisterait pour un proxy malveillant à tenter de perturber le fonctionnement du protocole d'établissement de clé en envoyant de faux messages à B. Sans un système de protection adapté, un proxy égoïste pourrait paralyser tout le système et faire échouer l'établissement de clé entre A et B. Ce genre de comportement de la part des proxies a été soigneusement pris en compte dans la conception de notre solution. Prévenir ce type de mauvaise conduite de survenir est évidemment la clé pour s'en prémunir. Mais, on l'a vu, il est impossible d'exclure qu'un ou plusieurs proxies malicieux aient été choisis par A. Aussi, une deuxième ligne de défense consiste à pouvoir récupérer des conséquences de la mauvaise conduite d'un proxy. Pour cette raison, nous nous sommes concentrés sur l'élaboration à la fois de techniques de prévention et de réaction pour surmonter cette attaque. La technique de prévention commence au niveau de B. Ce dernier détecte les proxies non coopératifs lors de la reconstitution du modèle initial du secret et renvoie un rapport à A contenant la liste des identifiants des participants. Ainsi, A apprend les identités des proxies qui se sont mal conduits et empêche leur sélection dans l'avenir.

La technique de récupération est quant à elle assurée par l'usage du mécanisme de correction d'erreurs. L'ajout de redondance à chaque fragment transmis par un nœud assistant rend la reconstruction de la clé principale possible, et ce même si certains proxies malveillants refusent de coopérer au cours du processus. Ceci assure la résilience de notre solution à ce type d'attaque.

Un dernier type d'attaques DoS serait possible si un attaquant utilisait le mécanisme proposé dans ce papier pour cibler des nœuds ou des systèmes qui participent à la mise en œuvre de la solution proposée. Plus précisément, des attaques par épuisement de ressources pourrait se produire si un client malveillant A tentait d'épuiser la batterie et / ou d'augmenter la consommation des ressources de calcul des proxies, en faisant croire qu'il a besoin de leur aide. Ainsi, un

modèle cognitif de confiance permettant d'évaluer les entités qui demandent de l'assistance est également nécessaire au niveau des proxies.

5 Conclusion

Ce papier présente un nouveau protocole collaboratif pour l'établissement de clé de session dans l'environnement hétérogène M2M. Ce nouveau système permet à un nœud fortement contraint en ressources d'utiliser des primitives de cryptographie asymétrique pour établir un secret partagé avec un serveur distant. Cela est possible grâce à des échanges simples avec des nœuds voisins, échanges qui consomment considérablement moins d'énergie qu'une utilisation réelle des primitives cryptographiques asymétriques. Les prochaines étapes de la validation de ce protocole consisteront à effectuer des simulations, afin de quantifier précisément les économies en termes d'énergie en fonction du type d'environnement M2M (densité, types de nœuds). Le modèle de confiance utilisé pour sélectionner les nœuds assistants sera également spécifié, de sorte qu'il soit efficace en énergie. Dans ce cas également, l'hétérogénéité de l'environnement M2M sera exploitée afin d'optimiser les opérations cognitives visant à produire, de manière distribuée, un modèle de confiance global.

Références

1. W. Geng, S.Talwar, K. Johnsson, N. Himayat, K.D. Johnson, "M2M : From mobile to embedded internet," Communications Magazine, IEEE, vol.49, no.4, pp.36-43, April 2011.

2. I. Cha, Y. Shah, A. U. Schmidt , A. Leicher , M. Meyerstein," Security and Trust for M2M Communications", in Proceedings of the Wireless World Research Forum Meeing 22, Paris, France, May 5-7, 2009

3. L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in Proceedings of the 9th ACM conference on Computer and communications security, Washington, DC, USA, November 18-22 2002, pp. 41–47.

4. H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in IEEE Symposium on Security and Privacy, Berkeley, California, May 11-14 2003, pp. 197–213.

5. D. Liu and P. Ning, "Location-based pairwise key establishments for static sensor networks," in Proceedings of the 1st ACM Workshop on Security of Ad Hoc and Sensor Networks (CCS '03), pp. 72–82, October 2003.

6. W. Du, J. Deng, Y. S. Han, S. Chen, and P. K. Varshney, "A key management scheme for wireless sensor networks using deployment knowledge," in Proceedings of IEEE INFOCOM'04. 2004.

7. S. Schmidt, H. Krahn, S. Fischer, D. Watjen, "A security architecture for mobile wireless sensor networks," in Proceedings of the 1st European Workshop on Security in Ad-Hoc and Sensor Networks (ESAS'04) (Heidelberg, Germany), Vol. 3313, August 2004.

8. A. Perrig, R. Szewczyk, V. Wen, D. Cullar, and J. D. Tygar, "Spins : Security protocols for sensor networks," in Proceedings of the 7th Annual ACM/IEEE Internation Conference on Mobile Computing and Networking (MobiCom), Rome, Italy, July 2001, pp. 189–199.

9. D.J. Malan, M. Welsh, M.D. Smith, "Implementing public-key infrastructure for sensor networks," ACM Transactions on Sensor Networks 4 (4), 1–23, 2008.

10. N. Gura, A. Patel, and A.Wander, "Comparing elliptic curve cryptography and RSA on 8-bit CPUs," in Proceedings of the 2004 Workshop on Cryptographic Hardware and Embedded Systems (CHES 2004), pages 119–132, August 2004.

11. H. Wang and Q. Li, "Efficient implementation of public key cryptosystems on mote sensors," in Proceedings of International Conference on Information and Communication Security (ICICS), pages 519–528, Dec. 2006.

12. J. Lopez, "Unleashing public-key cryptography in wireless sensor networks," Journal of Computer Security, vol. 14, no. 5, pp. 469–482, 2006.

13. N. R. Potlapally, S. Ravi, A. Raghunathan, N.K. Jha, "A study of the energy consumption characteristics of cryptographic algorithms and security protocols," IEEE Transactions on Mobile Computing 5 (2) (2006) 128–143.

14. D. J. Malan, M. Welsh, M. D. Smith, "A Public-Key Infrastructure for Key Distribution in TinyOS Based on Elliptic Curve Cryptography," First IEEE International Conference on Sensor and Ad Hoc Communications and Networks, 2004.

15. E.-O. Blass and M. Zitterbart, "Towards acceptable public-key encryption in sensor networks," in The 2nd International Workshop on Ubiquitous Computing (ACM SIGMIS), May 2005.

16. P. K. Dutta, J. W. Hui, D. C. Chu, and D. E. Culler, "Securing the deluge network programming system," in Proceedings of the 5th International Conference on Information Processing in Sensor Networks (IPSN). ACM, New York, NY, 326–333. 2006.

17. R. Watro, D. Kong, S. Cuti, C. Gardiner, C. Lynn, P. Kruss, "TinyPK : Securing sensor networks with public key technology," in Proceedings of the 2nd ACM workshop on Security of Ad Hoc and Sensor Networks, pp. 59–64, 2004, USA.

18. R.L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," Communications of the ACM, vol. 21, no. 2, pp 120–126, February 1978.

19. W. Hu, P. Corke , W. C. Shih, L. Overs, "secFleck : A Public Key Technology Platform for Wireless Sensor Networks," in Proceedings of the 6th European Conference on Wireless Sensor Networks, February 11-13, 2009, Cork, Ireland.

20. G. Gaubatz, J. Kaps, and B. Sunar, "Public key cryptography in sensor networks–revisited," Lecture Notes in Computer Science, vol. 3313, pp. 2–18, 2005.

21. G. Gaubatz, J. Kaps, E. Ozturk, and B. Sunar, "State of the art in ultralow power public key cryptography for wireless sensor networks," in Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications Workshops. IEEE Computer Society Washington, DC, USA, 2005, pp. 146–150.

22. Q. Huang, J. Cukier, H. Kobayashi, B. Liu, J. Zhang, "Fast authenticated key establishment protocols for self-organizing sensor networks," in Proceedings of the 2nd ACM International Conference on Wireless Sensor Networks and Applications, ACM Press, 2003 ; 141–150.

23. P. Kotzanikolaou, E. Magkos, D. Vergados, and M. Stefanidakis, "Secure and practical key establishment for distributed sensor networks," in Security and Communication Networks, Wiley InterScience, 2009.

24. P. Kotzanikolaou, E. Magkos, C. Douligeris, V. Chrissikopoulos, "Hybrid key establishment for multiphase self-organized sensor networks," in Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks WoWMoM'05, IEEE Press, 2005 ; 581–587.

25. E. Magkos, P. Kotzanikolaou, M. Stefanidakis, D. Vergados, "An asymmetric key establishment protocol for multiphase selforganized sensor networks," in Proceedings of the 12th European Wireless Conference (EW'06), March 2006.

26. J. Mache, C.-Y. Wan, and M. Yarvis, "Exploiting heterogeneity for sensor network security," in Proceedings of IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, June 2008, pp. 591–593.

27. R. Riaz, A. Naureen, A. Akram, A. Akbar, K. Kim, and H. Farooq Ahmed, "A unified security framework with three key management schemes for wireless sensor networks," Computer Communications, vol. 31, no. 18, pp. 4269–4280, 2008.

28. M. J. Wiener, "Performance comparison of public-key cryptosystems," RSA Laboratories'CryptoBytes, vol. 4, no. 1, pp. 1+3–5, 1998.

29. S. Seys and B. Preneel, “Power consumption evaluation of efficient digital signature schemes for low power devices,” in Proceedings of the 2005 IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (IEEE WiMob 2005), pages 79–86. IEEE, 2005.

30. L. Reyzin and N. Reyzin, “Better than BiBa : Short one-time signatures with fast signing and verifying,” in Proceedings of the 7th Australian Conference on Information Security and Privacy, ser. Lecture Notes in Computer Science, J. Seberry, Ed., vol. 2384. Springer, 2002, pp. 144–153.

État des lieux de la sécurité des communications cellulaires

Chaouki Kasmi et Benjamin Morin

ANSSI 51 bd. de la Tour Maubourg 75700 Paris Cedex 07 France

Résumé Le GSM fête cette année ses 20 ans d'existence. Le nombre d'utilisateurs est estimé à 80% de la population mondiale, soit 5 milliards d'individus dans plus de 200 pays [22]. La prolifération des terminaux mobiles et la multiplication de leurs usages (y compris dans des secteurs où ils sont utilisés pour des communications entre machines) imposent la satisfaction d'exigences de sécurité fortes.

Le thème de la sécurité des communications mobiles est vaste car il englobe celui de l'accès radio, de l'infrastructure des réseaux, des terminaux et des applications qui s'y exécutent. Cet article se focalise sur les deux premiers thèmes.

Depuis 2002, différents projets indépendants s'intéressent aux principes de sécurité mis en œuvre dans les réseaux de téléphonie mobile de 2ème et 3ème génération. Cette présentation dresse un panorama des objectifs et des impacts de ces projets sur la sécurité des réseaux de téléphonie mobile. Les principes de sécurité sont évoqués, ainsi que les problèmes de géolocalisation.

1 Introduction

Le thème de la sécurité des communications mobiles est vaste car il englobe celui de l'accès radio, de l'infrastructure des réseaux, des terminaux et des applications qui s'y exécutent.

De nombreux articles ont été publiés ces dernières années sur le thème de la sécurité des *smartphones*, en se focalisant essentiellement sur le domaine applicatif. Certains travaux récents portant sur les télécommunications cellulaires ont néanmoins fait des avancées significatives et ont démontré le réalisme d'attaques jusqu'alors réputées théoriques. Ces travaux pointent du doigt une certaine inertie dans la prise en compte des menaces, inertie qui tranche avec les mutations du secteur des terminaux dits intelligents. Cet article s'intéresse essentiellement à ces derniers travaux et présente un état des lieux de la sécurité des communications cellulaires et des projets indépendants qui s'y rapportent.

La première section situe le sujet de cet article dans le paysage actuel des télécommunications mobiles. L'article se poursuit par une description succincte des éléments qui composent un réseau de téléphonie mobile. Ces éléments sont nécessaires à la compréhension de la section 4, consacrée aux principes sur lesquels repose la sécurité des réseaux mobiles et à leurs vulnérabilités, et de la section 5, consacrée aux projets indépendants visant à analyser la sécurité de ces réseaux. La section 6 aborde les problèmes de géolocalisation et la dernière conclue l'article.

2 Contexte technique et économique

Le secteur de la téléphonie mobile a connu de profondes mutations en l'espace de quelques années. Le phénomène dit de « convergence » a progressivement transformé des téléphones portables simples (« *feature phones* ») en terminaux multi-fonctions beaucoup plus sophistiqués (« *smart phones* »).

Pour accompagner cette transformation, ce secteur initialement réservé à un nombre limité d'acteurs (fabricants de terminaux et opérateurs de réseaux de télécommunications [1]) s'est ouvert à d'autres, tels que les développeurs de systèmes d'exploitation ou d'applications, les fournisseurs de contenus, les utilisateurs professionnels, etc. Avec un support adéquat du matériel, les systèmes d'exploitation doivent apporter des garanties d'intégrité et d'isolation afin par exemple d'empêcher une application malveillante de perturber le réseau de télécommunication (que cette application soit installée de façon délibérée ou non par le porteur du terminal). Les utilisateurs n'ont alors pas la possibilité de contrôler intégralement leur terminal. Ce dernier point est un exemple de différence notable entre le modèle de sécurité des terminaux mobiles et celui des ordinateurs, qui illustre la difficulté à concilier les exigences de sécurité des différents acteurs de la téléphonie mobile.

Le besoin de séparer les domaines d'exécution des différentes parties se traduit au niveau de l'architecture logique et physique des plateformes mobiles. Comme nous le verrons dans la section suivante, les plateformes matérielles distinguent généralement le domaine applicatif, au sein duquel sont exécutées les applications de l'utilisateur, du domaine radio, qui gère les communications avec le réseau.

Comme évoqué précédemment, le secteur de la téléphonie mobile est resté clos pendant longtemps. Son ouverture relativement récente est a priori positive sur le plan de la sécurité car elle concourt à une meilleure confiance dans les terminaux, en permettant aux utilisateurs de mieux maîtriser leur fonctionnement. Cette ouverture demeure cependant partielle, et ce pour plusieurs raisons. Elle l'est parce que certains acteurs majeurs (Apple et RIM en particulier) maîtrisent intégralement la chaine de conception des terminaux, depuis la plateforme matérielle jusqu'à la distribution des applications. Le rachat récent de Motorola par Google semble d'ailleurs indiquer que cette « verticalisation » du marché s'accentue. L'ouverture est aussi et surtout partielle parce qu'elle se limite au domaine applicatif des terminaux ; le domaine radio demeure quant à lui relativement opaque.

1. Les fabricants de cartes à puce sont aussi des acteurs du secteur via les cartes SIM, mais ils n'interviennent pas directement sur la conception des terminaux et les éléments du réseau de télécommunication.

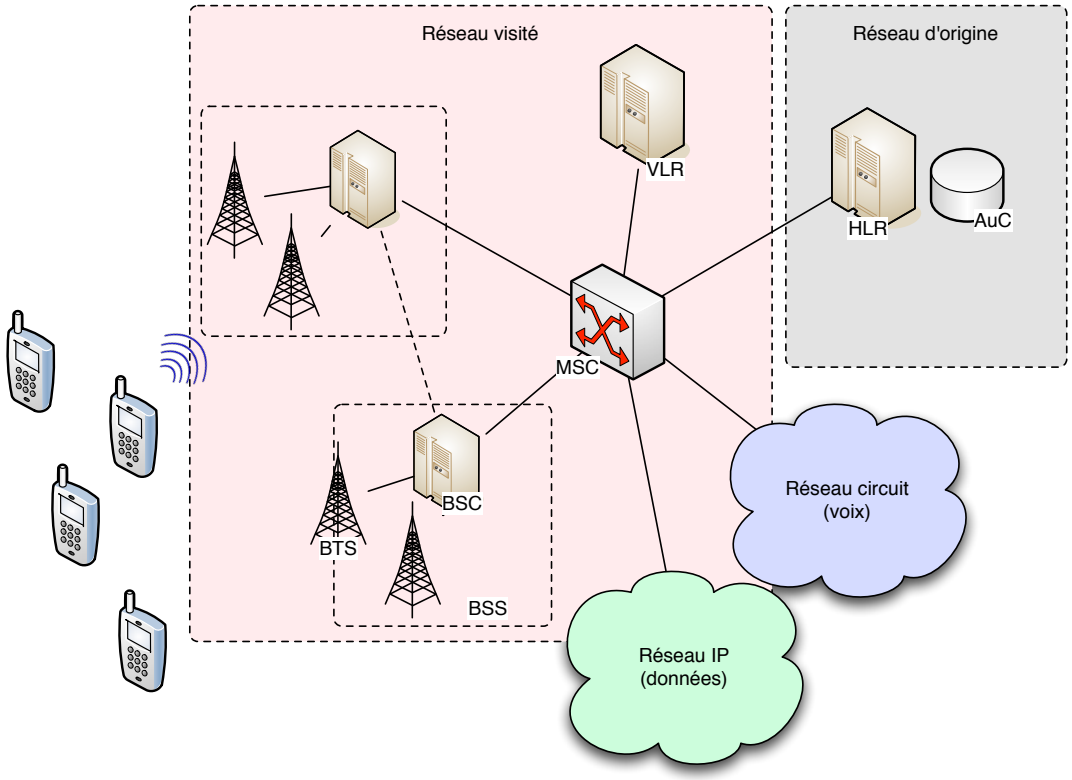


Figure 1. Architecture type d'un réseau cellulaire GSM

3 Éléments d'architecture

Cette section propose un survol des principaux éléments qui composent un réseau cellulaire. Nous séparons cette description en deux parties : la première est consacrée à l'infrastructure du réseau cellulaire et la seconde à l'architecture des terminaux mobiles (également appelés stations mobiles, ou *mobile equipement*, ME).

3.1 Architecture d'un réseau cellulaire

Les éléments qui composent un réseau cellulaire et la façon de les désigner ont évolué avec les différentes générations de systèmes de télécommunications (GSM, EDGE, UMTS, etc.). La description qui suit est délibérément simple. Nous ne rentrons pas dans les détails afin de faciliter la compréhension des sections suivantes. Nous renvoyons le lecteur intéressé aux ouvrages spécialisés sur ce sujet [24].

Comme l'illustre la figure 1, une infrastructure de télécommunication cellulaire distingue deux types réseaux : le réseau d'*origine*, c'est-à-dire le réseau de l'opérateur avec lequel l'usager souscrit un abonnement, et le réseau *visité*, qui peut appartenir à un opérateur différent du précédent. Le réseau visité achemine les communications voix et/ou données de l'usager une fois que le terminal de ce dernier s'est correctement authentifié auprès de son réseau d'origine.

On peut distinguer deux principaux sous-systèmes :

- Le sous-système radio (*Base Station Subsystem, BSS*) assure les transmissions radio-électriques et gère les ressources radio. Il est constitué de stations de base (*Base Transciever Station*, BTS [2]) qui communiquent avec les stations mobiles par un lien radiofréquence, communément appelé « interface air ». Des équipements appelés *Base Station Controler* (BSC) contrôlent les stations de base ;
- Le sous-système réseau comprend des fonctions nécessaires à l'établissement des appels et à la mobilité. Il est notamment constitué de bases de données et de commutateurs :
  - Le centre de commutation des services mobiles (*Mobile Switching Center*, MSC) relie des contrôleurs de station de base au réseau téléphonique public (liaison voix) et à Internet (liaison de données) ;
  - Le HLR (*Home Location Register*) est une base de données de localisation et de caractérisation des abonnés. Pour les besoins d'itinérance, certaines données sont transmises à la base de données de la cellule visitée (*Visitor Location Register*, VLR).

Un élément important de l'architecture est le centre d'authentification (*Authentication Center*, AuC), qui dispose des éléments nécessaires à la sécurisation des communications, notamment les clés cryptographiques associées aux usagers. Ces clés servent notamment à dériver des clés temporaires qui sont transmises par l'AuC au MSC. La section 4 précise les échanges correspondants.

3.2 Architecture des terminaux

L'architecture logique type des smartphones actuels distingue généralement deux environnements d'exécution distincts. Le premier correspond au système d'exploitation applicatif, qui assure notamment l'exécution des applications de l'utilisateur. Le second correspond à la pile logicielle responsable des communications réseau (GSM, 3G, etc.). Ce dernier environnement est généralement appelé le *baseband*. Cette architecture logique peut se décliner sous différentes architectures physiques. Certains terminaux utilisent deux processeurs distincts, physiquement séparés et reliés par un bus de communication. Ces deux processeurs peuvent également être inclus dans une même puce (*system-on-chip, SOC*).

2. On trouve aussi l'appellation BST, *Base Station Trancievер.*

D'autres solutions consistent à utiliser des machines virtuelles pour réaliser la séparation des deux environnements d'exécution sur un seul et même processeur.

Dans tous les cas, les principes de conception des environnements d'exécution des *basebands* ont peu évolué depuis les débuts de la téléphonie mobile : ils sont généralement exempts des protections standards disponibles sur les processeurs de plus haute gamme (unité de gestion de la mémoire ou *MMU*, par exemple) et leur code obéit souvent aux pratiques de développement en vigueur au début des années 90, pratiques qui ne mettaient pas nécessairement l'accent sur la sécurité. La présence de failles logicielles au sein de ces environnements d'exécution est donc à craindre, et leur exploitation pourrait avoir des conséquences importantes en l'absence de mécanisme d'isolation entre les tâches.

La carte SIM (*Subscriber Identity Module*) [12,11] peut également être considérée comme un environnement d'exécution distinct, même si celui-ci est beaucoup plus restreint que les deux précédents. La carte SIM est en effet une carte à puce qui dispose d'une capacité de calcul et qui renferme notamment les éléments d'identification du terminal et de son porteur ainsi que des clés cryptographiques. Son rôle est fondamental dans la sécurisation des télécommunications mobiles. Cette carte appartient à l'opérateur car c'est par elle que passent les autorisations d'accès au réseau. Les cartes USIM (*Universal SIM*) [10,13] sont fonctionnellement équivalentes aux cartes SIM ; elles renferment les mécanismes cryptographiques utilisés dans le cadre des réseaux de téléphonie mobile de troisième génération. Dans l'avenir, l'élément de confiance que représente la carte USIM aura vocation à héberger des applications tierce partie pour des usages de type NFC (*Near-Field Communication*), tels que le paiement sans contact ou le contrôle d'accès. La validation des applications incluses au sein des cartes USIM sera alors primordiale, afin d'empêcher un attaquant d'implanter une application malveillante au sein de cet élément de confiance.

4 Mécanismes de sécurité des réseaux de mobiles

Cette section propose un survol des nombreuses familles de cryptosystèmes utilisés dans les réseaux de télécommunication mobiles pour satisfaire les exigences de confidentialité et d'intégrité des communications et pour authentifier les terminaux mobiles. L'emploi de tel ou tel cryptosystème dépend de la fonction de sécurité visée (authentification, négociation de clé, chiffrement, intégrité), de l'opérateur du réseau de télécommunication et de la génération de réseau considérée. Nous séparerons cette dernière en deux catégories principales : les réseaux GSM (2G), GPRS (2.5G) et EDGE (2.75G), d'une part, et les réseaux UMTS (3G) et LTE (4G), d'autre part (voir figure 2).

Nous débutons cette section par une présentation des éléments utilisés pour identifier les terminaux mobiles. Nous la concluons par un commentaire sur une

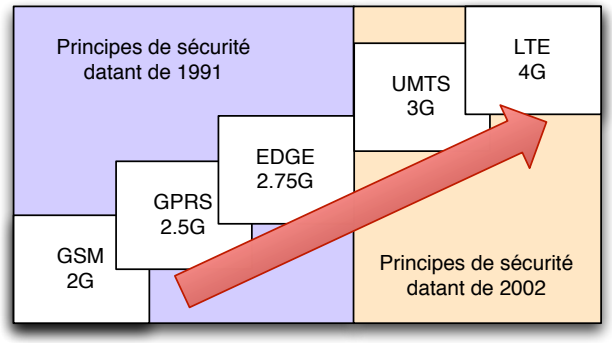


Figure 2. Evolution des réseaux de téléphonie mobile

exigence de sécurité souvent négligée et pourtant essentielle dans le cas des réseaux de télécommunication : la disponibilité.

4.1 Eléments d'identification

On distingue deux principaux éléments d'identification dans les terminaux mobiles selon qu'ils concernent le terminal ou son porteur. Ils sont utilisés dans toutes les générations de réseau (2G à 4G).

L'IMSI (*International Mobile Subscriber Identity*) est un numéro unique permettant à l'opérateur du réseau d'identifier le *porteur* du terminal mobile. Il est stocké dans la carte SIM et est constitué des indicatifs du pays d'origine du porteur (le MCC, *Mobile Country Code*, qui vaut par exemple 208 pour la France) et de l'opérateur (le MNC, *Mobile Network Code*), ainsi que du numéro de l'abonné.

L'IMSI est envoyé en clair aux antennes relai lors du protocole initial d'authentification du terminal auprès du réseau. Afin d'empêcher un attaquant d'identifier et de traçer le porteur d'un terminal à l'aide d'un dispositif d'écoute de communication radio, un numéro temporaire appelé TMSI (*Temporary IMSI*) est attribué au porteur par l'opérateur. Le TMSI est utilisé en lieu et place de l'IMSI dans la suite des communications et seul l'opérateur est en mesure d'établir la correspondance entre l'IMSI et le TMSI.

L'IMEI (*International Mobile Equipement Identity*) est un numéro supposé unique permettant à l'opérateur d'identifier le terminal mobile. Il est constitué des numéros de série et de modèle[3] de l'équipement, ainsi que d'un code

3. L'identifiant de modèle, appelé TAC (*Type Allocation Code*), est propre à un fabricant de terminal et est délivré par une autorité de certification centrale.

de contrôle (formule de Luhn). L'IMEI sert notamment à empêcher un terminal volé ou perdu de joindre le réseau de communication.

En ce qui concerne le réseau, des informations d'identification sont diffusées par l'opérateur au travers des antennes relais et à destination des terminaux mobiles. On y retrouve par exemple le MCC, le MNC, et les informations permettant aux stations mobiles de se connecter au réseau d'opérateur.

4.2 Authentification des équipements mobiles

Hormis le besoin évident de facturation des communications, l'authentification forte des terminaux mobiles vise à contrer des tentatives d'usurpation d'identité d'un abonné. A noter que l'authentification dont il est question ici est bien celle du terminal vis-à-vis du réseau ; le porteur s'authentifie pour sa part auprès de son terminal (plus exactement, sa carte SIM) par la saisie de son code PIN.

Le mode d'authentification des terminaux mobiles est probablement l'évolution la plus importante entre les deux grandes familles de réseaux. Dans le cas des réseaux 2G, l'authentification est unilatérale : seul le terminal s'authentifie auprès du réseau d'opérateur. Dans le cas des réseaux 3G, l'authentification est mutuelle. Cette évolution est fondamentale car elle permet de contrer des attaques par le milieu, dans lesquelles un adversaire tente de se faire passer pour le réseau de l'opérateur auprès du terminal d'un abonné et tenter ensuite d'intercepter ses communications.

Le protocole d'authentification utilisé dans les réseaux GSM repose sur un cryptosystème symétrique appelé A3. L'authentification unilatérale du terminal consiste pour ce dernier à calculer un code d'authentification de message (MAC) en réponse à un challenge RAND envoyé par l'opérateur. Ce challenge est un aléa généré par le *centre d'authentification* (AuC) de l'opérateur, puis acheminé jusqu'au terminal via un centre de commutation (MSC), puis une station de base.

Le calcul de la réponse au challenge, désigné SRES, nécessite une clé symétrique partagée, K_i, connue exclusivement de la carte à puce et de l'AuC [4]. Le calcul de SRES est réalisé conjointement par la carte à puce et par l'AuC. Ce dernier communique SRES au MSC, qui peut ainsi le comparer au SRES envoyé par le terminal en réponse au challenge. L'authentification est acceptée en cas d'égalité. Ce protocole est résumé en figure 3.

Un algorithme de dérivation de clé appelé A8 est utilisé pour produire une clé symétrique K_c à partir de K_i et de RAND. Cette clé de session K_c sert à chiffrer les communications ; elle est générée par la carte à puce du terminal, d'une part, et par l'AuC d'autre part. La carte à puce fournit K_c au terminal, tandis que

4. L'AuC dispose de la base de données des correspondances entre les IMSI et les clés K_i des abonnés.

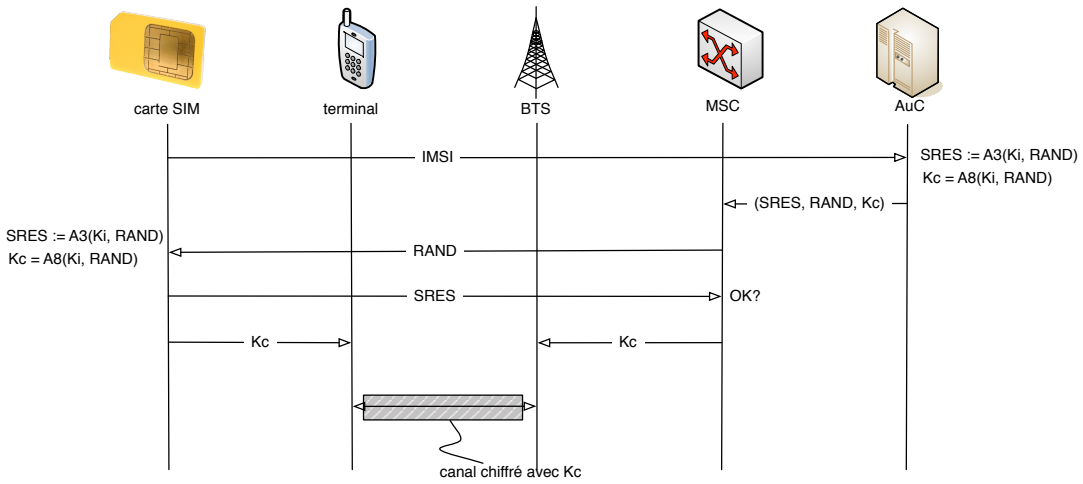


Figure 3. Principe d'authentification dans le cadre de réseau de 2ème génération (GSM)

l'AuC la fournit au MSC, qui la transmet à l'antenne relais. Le chiffrement des communications est donc ensuite réalisé entre le terminal et l'antenne relais.

On peut noter que les algorithmes A3 et A8 ne sont pas normalisés, ce qui signifie que les opérateurs sont libres d'utiliser des algorithmes propriétaires non publics. Ces algorithmes doivent en principe avoir de bonnes propriétés cryptographiques afin de résister à des attaques, mais ce n'a pas toujours été le cas. Ainsi, l'algorithme COMP128 a par exemple été utilisé alors qu'il était vulnérable et permettait au détenteur d'une carte SIM de cloner celle-ci avec quelques dizaines de milliers de couples RAND/SRES.

Le protocole d'authentification utilisé dans les réseaux de troisième génération permet au terminal et au réseau de l'opérateur de s'authentifier mutuellement. La protection des communications dans les réseaux de troisième génération repose sur un ensemble de fonctions de dérivation de clés et une clé symétrique maîtresse K, qui joue un rôle analogue à la clé K_i des réseaux de seconde génération. La clé K est notamment impliquée dans le calcul (non détaillé ici) d'un jeton d'authentification AUTN, qui est envoyé par l'AuC au terminal mobile en plus de l'aléa RAND. Ce jeton permet à la carte USIM du terminal d'authentifier le réseau de l'opérateur avant d'envoyer à son tour la réponse au challenge.

Milenage [9] est un algorithme fondé sur AES qui est par exemple utilisé pour l'authentification et la dérivation de clés dans les réseaux UMTS.

A l'instar de la fonction A8 du GSM, une fonction de dérivation produit une clé CK (analogue à la clé K_c) utilisée pour chiffrer le trafic de données et

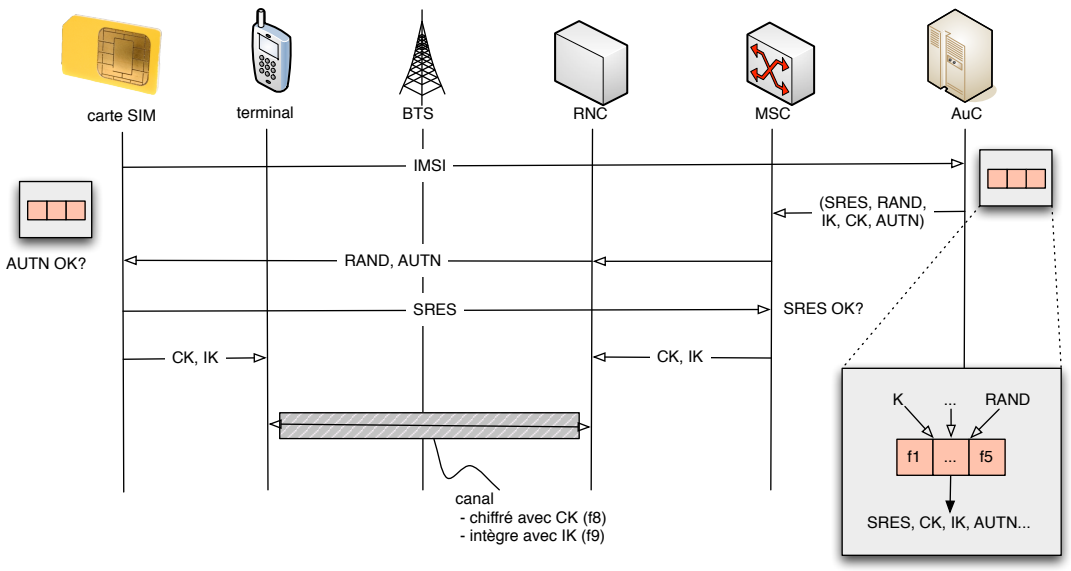


Figure 4. Authentification mutuelle en 3G

de signalisation. Une autre fonction de dérivation génère également une clé IK (*Integrity Key*) utilisée pour l'intégrité des échanges.

La figure 4 illustre le déroulement du protocole d'authentification de la 3G.

4.3 Confidentialité et intégrité des communications

Des algorithmes cryptographiques sont mis en œuvre dans le but de protéger la confidentialité des données échangées. Ces algorithmes plus ou moins robustes en fonction du protocole mis en œuvre ont pour but de protéger les communications radiofréquences. Notons que la protection en confidentialité est optionnelle (le chiffrement est à l'initiative du réseau).

La confidentialité dans les réseaux de seconde génération repose sur l'emploi de la clé K_c et d'un algorithme de chiffrement à flot normalisé désigné A5. Le chiffrement porte sur l'ensemble du trafic et de la signalisation. Il existe en fait quatre algorithmes de chiffrement désignés A5/1 à A5/4. Le choix de l'algorithme à utiliser pour une communication fait l'objet d'une négociation, le réseau choisissant un algorithme dans la liste de ceux que lui propose le terminal. Cette liste contient au moins A5/1. Dans le cas du GPRS, les algorithmes GEA sont utilisés [7,8].

L'algorithme A5/1 est massivement déployé, mais n'offre pas une protection absolue en confidentialité. Plusieurs attaques « à clair connu » contre A5/1 ont en

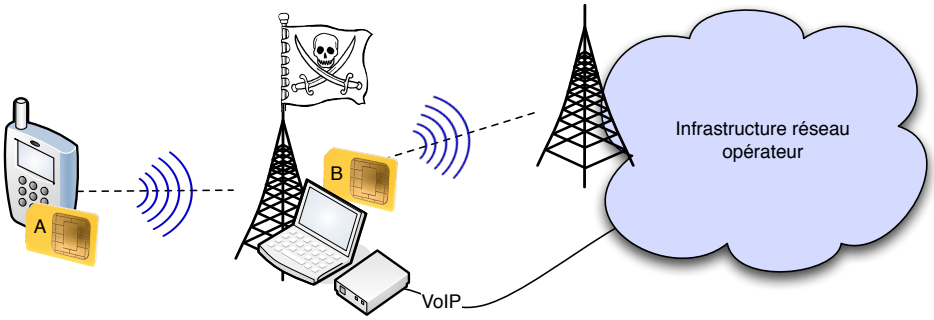


Figure 5. Attaque *man in the middle* dans un réseau cellulaire

effet été publiées depuis la fin des années 1990 [15,17,14]. Ces attaques permettent à un adversaire passif[5] disposant de tables précalculées et de la connaissance d'une partie du trafic clair de déchiffrer une communication en quasi-temps réel. La quantité de données claires nécessaire (issues du trafic de signalisation et/ou de *padding* constant) et surtout le volume des tables précalculées constituaient les principaux obstacles à la réalisation pratique des attaques. En 2010, Karsten Nohl a finalement fait une démonstration publique de déchiffrement de communications chiffrées avec A5/1 à l'occasion du Chaos Computer Congress [27].

L'algorithme A5/2 a essentiellement été conçu pour l'export et n'est utilisé que dans un nombre restreint de pays où la législation en matière de chiffrement est restrictive. Des attaques très réalistes ont été publiées contre A5/2.

Les algorithmes A5/3 [7] et A5/4 [8] sont dérivés de l'algorithme KASUMI utilisé dans la 3G. Ces algorithmes ont été spécifiés plus tardivement (2002) et leur déploiement demeure limité. En 2010, des faiblesses ont été identifiées dans l'algorithme KASUMI [16], mais elles ne permettent toutefois pas de réaliser des attaques pratiques à l'heure actuelle.

L'absence d'attaque pratique contre les algorithmes A5/3 et A5/4 n'élimine pas pour autant la menace d'une interception des communications par un attaquant *actif*. Compte tenu de l'absence d'authentification du réseau par le mobile et du fait que c'est *in fine* le réseau qui choisit l'algorithme de chiffrement, un attaquant disposant d'une fausse station de base peut en effet déchiffrer les communications des terminaux qui s'y connectent.

Une façon de procéder consiste pour l'adversaire à forcer l'utilisation d'un canal de communication en clair, ou bien d'un algorithme faible (par exemple A5/2) pour reconstituer la clé K_c. En théorie, l'utilisation d'un canal de communication

5. C'est-à-dire disposant d'un équipement d'écoute des communications radio.

non chiffré doit être indiqué sur les terminaux, mais ce n'est pas toujours le cas en pratique.

Notons par ailleurs qu'un terminal peut réutiliser la même clé K_c et le même aléa entre des communications chiffrées en A5/1 et A5/3. Après avoir enregistré une communication chiffrée en A5/3 entre un terminal et une station de base légitime, un attaquant muni d'une fausse station de base peut ainsi négocier la réutilisation de la clé de chiffrement K_c dans le cadre d'une communication en A5/1 avec le terminal. L'attaquant peut ensuite obtenir K_c en appliquant les attaques mentionnées ci-dessus sur du trafic de service (*idle frames*) et ainsi déchiffrer le trafic initialement enregistré.

La section 5 donne plus de détails sur les projets de développement d'outils d'analyse GSM.

La confidentialité dans les réseaux de troisième génération repose sur l'utilisation de deux nouveaux algorithmes de chiffrement, KASUMI et SNOW 3G, et de clés de plus grande taille (128 bits). Ces deux algorithmes sont également utilisés pour assurer l'intégrité des communications.

KASUMI est un algorithme de chiffrement par blocs utilisé par défaut dans les réseaux UMTS. SNOW 3G est un algorithme de chiffrement à flot « de repli », dans l'hypothèse où KASUMI serait cassé. Cet algorithme est implanté au sein des terminaux mobiles actuels, mais pas nécessairement dans les équipements d'infrastructure des opérateurs. Comme évoqué précédemment, une attaque « à clés reliées » contre KASUMI a été publiée en 2010 [16], mais le modèle d'attaque considéré est peu réaliste et ne remet pas en cause la sécurité pratique du GSM et de l'UMTS.

Les réseaux de troisième génération apportent donc des améliorations significatives en matière de confidentialité. Le chiffrement demeure à l'initiative du réseau (qui peut décider de rester en clair), cependant le réseau doit authentifier l'ordre de rester en clair, donc être reconnu de l'opérateur d'origine. Les attaques par fausses stations de base sont donc inopérantes.

Les algorithmes de chiffrement et d'intégrité ont été partiellement renouvellés dans le cas des réseaux de quatrième génération. Ils s'appuient sur deux algorithmes, respectivement fondés sur SNOW 3G et AES. Des discussions sont en cours pour intégrer un troisième jeu d'algorithmes, baptisé ZUC [20], à la demande de la Chine, mais plusieurs failles majeures ont été identifiées dans les spécifications intermédiaires de cet algorithme [21].

Notons enfin que les réseaux de troisième génération incluent un équipement appelé RNC (c.f. figure 4) jusqu'auquel se prolonge le chiffrement des communications (celui-ci se termine au niveau des BTS dans le cas du GSM). La protection physique de cet équipement est meilleure que celle des BTS.

4.4 Disponibilité

Le terme « disponibilité » peut soit s'interpréter comme la capacité du réseau de l'opérateur à résister à des attaques visant à le rendre inopérant (c.-à-d. sa résilience), soit être associé à la couverture radiofréquence sur un territoire donné pour les différents services voix et données des réseaux GSM, GPRS, EDGE et UMTS. Nous abordons ici ces deux interprétations.

Disponibilité/couverture : cette interprétation de la disponibilité peut a priori sembler étrangère aux problèmes de sécurité, mais elle a pourtant une importance. Malgré les investissements réalisés par les opérateurs [30], la couverture réseau du territoire en 3G demeure inférieure à celle du GSM et varie selon les opérateurs [28,29,32]. Tous les usagers ne disposent donc pas d'alternative plus sécurisée au GSM, ce qui empêche de « bloquer » les terminaux en 3G. Les problèmes évoqués précédemment demeurent donc d'actualité. En France, la technologie 4G en est encore au stade expérimental (d'autres pays comme les États-Unis ont fait le choix de passer directement à la 4G).

Disponibilité/résilience : la disponibilité revêt une importance toute particulière dans le cas des réseaux cellulaires car les terminaux mobiles se substituent de plus en plus aux terminaux fixes.

La disponibilité au sein des réseaux GSM repose en partie sur l'hypothèse que les terminaux mobiles se comportent « correctement » vis-à-vis du réseau. L'opacité des principes de conception des terminaux mobiles, qui a prévalu pendant longtemps, a largement contribué à ce que cette hypothèse soit considérée valide car elle complexifie la recherche de vulnérabilités.

Cependant, des projets indépendants d'analyse des principes de conception des équipements utilisés dans un réseau GSM ont récemment mis en évidence des vulnérabilités sur certains équipements. Des failles d'implémentation logicielle sont susceptibles d'être déclenchées à distance par l'envoi de trames malformées et provoquer un blocage d'équipements d'infrastructure.

Un effet analogue peut également être obtenu par des attaques visant à épuiser les ressources des équipements par l'envoi de requêtes en rafale. Il a notamment été montré [31] que l'envoi massif de trames (correctes sur le plan protocolaire) peut provoquer une surconsommation de mémoire au niveau des BTS et engendrer un déni de service. Ces attaques ne sont pas facilement contrables sans un mécanisme de gestion de la mémoire robuste.

Il est donc nécessaire que les équipements soient soumis à des tests stricts afin que les déploiements actuels et futurs ne soient plus vulnérables à des attaques en disponibilité des réseaux d'opérateur.

5 Projets indépendants d'analyse des réseaux de télécommunication

Les attaques actives dans les réseaux GSM ou UMTS nécessitaient jusqu'à récemment des dispositifs qui étaient onhéreux et/ou difficiles à se procurer pour un individu, et dont les principes de conception ne sont pas publics. Les informations disponibles pour l'analyse de la robustesse des mécanismes de protection sont très théoriques et ne reflètent pas nécessairement les implémentations réelles.

La difficulté d'analyse des spécifications techniques a poussé la communauté de chercheurs en sécurité des systèmes d'information et de communication à réaliser une retro-conception des équipements de télécommunication. Ces projets portent sur des équipements mobiles et de cœur de réseau et permettent d'émuler un réseau d'opérateur complet.

5.1 Outils d'analyse passifs

Les équipements de radiocommunication ont connu une évolution très forte ces dernières années avec l'avènement de la radio logicielle (*Software Defined Radio, SDR*). Le domaine s'est ouvert à une nouvelle communauté de développeurs et n'est plus réservé aux spécialistes ; des équipements dédiés à la numérisation à la volée permettent aujourd'hui l'acquisition de signaux [6] et leur analyse à l'aide de bibliothèques de traitement numérique du signal telles que GNURadio [6]. Récemment, des terminaux mobiles d'ancienne génération [26] ont été reconfigurés afin d'exploiter l'architecture matérielle dédiée à l'application de téléphonie mobile (numérisation, décodage...).

Le projet Airprobe [1] propose les éléments de conception d'un outil d'analyse passif des trames radiofréquences afin d'évaluer la sécurité en confidentialité du protocole GSM. Il est constitué des modules d'acquisition, de démodulation et d'analyse.

Les attaques visant à retrouver efficacement une clé de chiffrement GSM reposent sur un compromis temps/mémoire qui nécessite de précalculer des tables. Le projet « The Kraken » [5] a notamment eu pour objet la construction de telles tables, en ayant recours à des optimisations pour réduire l'espace nécessaire à leur stockage et le temps de calcul. De plus, des attaques fonctionnelles ont été dévoilées portant sur les algorithmes GEA utilisés en GPRS et EDGE [25].

5.2 Outils d'analyse actifs

Plusieurs projets complémentaires portent sur la conception des équipements qui participent au fonctionnement d'un réseau de téléphonie mobile. Ces projets

6. Voir notamment `http://www.ettus.com/`

ont été réalisés par rétro-conception d'équipements existants, par analyse protocolaire et par l'étude des spécifications techniques.

Le projet OsomcomBB [4,34] propose une implémentation logicielle ouverte d'un *baseband* et de sa pile protocolaire GSM. Cet outil sert à émuler un téléphone mobile. La maîtrise complète du déroulement du protocole GSM que confère cette implémentation ouverte permet notamment à un auditeur de sécurité d'évaluer la robustesse des implémentations de piles protocolaires fermées.

D'autres projets portent sur la réalisation d'équipements réseau sur une base de radio logicielle. Le projet OpenBTS [3] a ainsi pour objectif de concevoir une station de base GSM et GPRS. Il est complété par le projet OpenBSC [2], qui porte sur la conception d'un contrôleur de stations de base GPRS, EDGE et UMTS. Il est important de noter que l'attaque par fausse station de base est limitée aux appels émis depuis l'équipement mobile cible, comme le terminal n'est plus lié à son opérateur de téléphonie mobile le routage des données depuis le réseau légitime ne peut plus être réalisé.

La simulation d'un réseau d'opérateur avec des équipements maîtrisés permet également d'étudier la robustesse des *basebands*. Weinmann [33] a montré que les logiciels exécutés au sein des *basebands* ne sont pas exempts de bogues et qu'il est possible d'exécuter du code arbitraire sur les *basebands* en leur adressant des messages malformés. Il existe d'ailleurs des projets de conception de *fuzzers* basés sur OpenBTS, visant à automatiser la détection de vulnérabilités sur différents modèles de *basebands* [23]. L'absence de protections standards au sein de ces environnements d'exécution expose donc les terminaux mobiles à une prise de contrôle de la partie radio des terminaux par un attaquant. De telles attaques sont certes très spécifiques (leur succès depend entre autres du système d'exploitation et de l'architecture matérielle des équipements), mais il existe relativement peu de fabricants de basebands et les mêmes équipements sont souvent utilisés dans différents modèles de terminaux. La prise de contrôle d'un nombre important de *basebands* pourrait alors servir à provoquer un déni de service sur les infrastructures réseau des opérateurs [23]. L'extension de la prise de contrôle du *baseband* à l'environnement d'exécution applicatif du terminal constitue également une menace crédible.

6 Gestion de la mobilité et géolocalisation

La protection en confidentialité et en intégrité des communications n'est qu'un élément parmi les exigences de sécurité que la téléphonie mobile doit satisfaire. La protection des données à caractère personnel est un autre problème. Cette section se penche plus particulièrement sur la protection des données de localisation géographique des usagers. Nous n'abordons pas ici les informations fournies par

des puces GPS implantées au sein de certains terminaux, mais celles qui émanent des réseaux cellulaires.

La géolocalisation est inextricablement liée à la mobilité. L'opérateur a en effet besoin de connaître le positionnement géographique des terminaux à des fins de routage des communications (gestion de l'itinérance, ou *roaming*). Dans le cas des réseaux GSM, l'opérateur gère une base de donnée centrale, appelée HLR (*Home Local Register*), au sein de laquelle la position de chaque abonné est renseignée et mise à jour régulièrement. Cette information permet de connaître l'adresse réseau du MSC auquel le mobile est rattaché à un instant donné.

Plusieurs techniques de géolocalisation GSM existent. La plus répandue, appelée Cell ID, repose sur la connaissance du positionnement géographique des antennes relais GSM. Il est possible de calculer par triangulation la position d'un terminal à partir des antennes relais environnantes auxquelles celui-ci est connecté. La précision de la position dépend de la densité d'antennes à un emplacement donné et peut aller de quelques dizaines de mètres à plusieurs kilomètres [18]. Plusieurs projets privés ou communautaires de cartographie des réseaux GSM constituent des bases de données établissant le lien entre les identifiants de cellules et leur position géographique [7].

Des sociétés telles qu'Apple, Google ou Microsoft, qui disposent de terminaux largement déployés et équipés de puces GPS, peuvent « déléguer » cette cartographie à leurs utilisateurs [8]. Alternativement, les techniques dites de *War Driving* [25] recensent également les identifiants de cellules GSM et Wi-Fi environnantes. Le projet WASP [9] (*Wireless Aerial Surveillance Platform*) propose même d'utiliser des drones à cet effet. Parallèlement, des applications malveillantes ont également été diffusées afin de localiser des abonnés en temps réel en exploitant les périphériques radios Wi-Fi, GSM/UMTS et GPS.

La géolocalisation a plusieurs applications légitimes, qui répondent à un besoin des utilisateurs. D'autres applications sont litigieuses (par exemple, le marketting ciblé), voire illégales (espionnage). Hormis quelques applications très spécifiques pour lesquelles un encadrement strict est nécessaire (par exemple, la localisation de personnes en détresse), la divulgation et/ou l'utilisation des données permettant de positionner un individu devraient donc requérir le consentement explicite de ce dernier. Plusieurs affaires récentes ont montré que ce n'est pas toujours le cas (voir par exemple l'application iPhoneTracker [10]).

7. Voir par exemple `http://www.opencellid.org/`, `http://crowdflow.net/`.
8. Chaque terminal est en mesure d'enregistrer la localisation géographique des stations de base environnantes à l'aide des informations fournies par sa puce GPS.
9. `https://www.defcon.org/html/defcon-19/dc-19-speakers.html\#Tassey`
10. `http://petewarden.github.com/iPhoneTracker/`

7 Conclusion et recommendations

Cet article s'est focalisé sur les mécanismes et les problèmes de sécurité associés à la couche radio des réseaux de télécommunication cellulaires, tant sur le plan protocolaire que sur celui des équipements terminaux qui les composent.

La migration progressive des protocoles de seconde génération vers ceux de troisième et quatrième génération apporte des améliorations significatives en matière de sécurité des échanges. L'authentification mutuelle d'un terminal et du réseau de l'opérateur et l'utilisation d'algorithmes de chiffrement robustes font partie de ces améliorations notables. Toutefois, la couverture des protocoles de dernière génération au niveau du territoire demeure inférieure à celles des protocoles plus anciens, tels que le GSM. Les faiblesses intrinsèques de ce dernier, qui mettent notamment en péril la confidentialité des échanges, sont connues sur le plan théorique depuis plus de dix ans et sont maintenant exploitables en pratique pour un coût modique. En attendant une généralisation des réseaux UMTS et LTE, ces vulnérabilités militent en faveur de l'utilisation de l'algorithme A5/3 au sein des réseaux GSM afin d'empêcher *a minima* les interceptions passives de communications.

La réalisation pratique de ces attaques a en partie été rendue possible par les travaux d'analyse et de rétro-conception de chercheurs indépendants. Ces travaux ont également pointé du doigt des vulnérabilités logicielles sur les équipements terminaux des réseaux cellulaires, à savoir les *basebands* (du côté des terminaux mobiles) et les stations de base (du côté de l'infrastructure des opérateurs). L'opacité des principes de conceptions de ces équipements complexifie les audits de sécurité visant à évaluer leur robustesse. Il est pourtant nécessaire d'entreprendre de telles évaluations, car la présence de vulnérabilités au sein ces équipements constitue une menace tout à fait crédible.

Plusieurs projets communautaires visant à concevoir les équipements d'un réseau cellulaire ont ainsi vu le jour ces deux dernières années. Ces travaux peuvent faciliter les évaluations de sécurité des équipements, évaluations qui seraient à même d'améliorer la confiance globale dans la robustesse des réseaux cellulaires.

Ces mêmes travaux permettent en contrepartie à des individus malveillants de mettre en place des dispositifs d'interception actifs et/ou d'identifier des vulnérabilités au sein des équipements terminaux. L'exploitation de ces vulnérabilités pourrait mettre en péril la confidentialité, l'intégrité et la disponibilité des réseaux cellulaires. C'est la raison pour laquelle il est nécessaire d'encadrer strictement les travaux d'analyse de ces réseaux. Notons à ce sujet que les équipements d'analyse dont il est question font partie des appareils dont l'utilisation est régie par les articles R226 du code pénal.

En particulier, l'atteinte à la disponibilité des réseaux de cellulaires pourrait avoir des répercussions graves. L'utilisation de ces réseaux n'est en effet plus

« limitée » aux seules communications entre individus. Elle s'étend également aux communications « de machine à machine » et ce dans de nombreux secteurs, incluant ceux d'importance vitale (transports, énergie, etc.).

En ce qui concerne les problématiques de géolocalisation, il est nécessaire de laisser aux usagers le choix de divulguer ou non à des tiers les informations présentes sur leur terminal qui permettent de les localiser. Les opérateurs doivent pour leur part limiter la fourniture des données de localisation nécessaires au routage à des applications extrêmement spécifiques et encadrées par la loi. La CNIL s'est par ailleurs prononcée à plusieurs reprises sur ce sujet [19].

La question du contrôle des informations de localisation par les usagers pose celle, plus générale, de la maîtrise des terminaux par leurs utilisateurs (particuliers ou institutionnels). De ce point de vue, le modèle de sécurité des *smartphones* est différent de celle des ordinateurs conventionnels car les terminaux n'« appartiennent » pas complètement à leurs usagers. Ceci s'explique en partie par le nombre de parties prenantes au niveau de chaque terminal mobile, parties dont les exigences ne sont pas nécessairement les mêmes que celles des usagers. La confiance globale dans les terminaux s'en trouve diluée.

Notamment en ce qui concerne les usages professionnels, la question de la maîtrise des terminaux mobiles est pourtant fondamentale. De ce point de vue, et en raison de la prolifération des outils d'attaques qui se déploient sur les plateformes mobiles, il est fondamental de séparer les usages personnels et professionnels.

La sécurité de la couche radio n'est qu'un des nombreux aspects que revêt la sécurité des réseaux cellulaires. L'accroissement des performances des terminaux mobiles et de leurs moyens de communication promet une multiplication des usages qu'il est impératif de sécuriser. Ainsi, l'ouverture de la carte USIM, élément de confiance des terminaux, à des applications tierces requiert une vigilance particulière. Dans une autre mesure, son remplacement prévisible par des implémentations logicielles reposant sur des fonctions de cloisonnement des processeurs telles que ARM/TrustZone nécessite également des précautions. Plus généralement, la maîtrise des terminaux par les usagers est un élément fondamental de la sécurité ; cette maîtrise plaide en faveur d'une ouverture accrue des systèmes exécutés au sein des terminaux.

Références

1. Airprobe GSM sniffing. `https://svn.berlin.ccc.de/projects/airprobe/`.
2. OpenBSC. `http://openbsc.osmocom.org/`.
3. OpenBTS : An opensource telephone network. `http://bipinb.com/openbts-an-opensource-telephone-network.htm`.

4. OsmocomBB. http://bb.osmocom.org/.
5. The Kraken. http://srlabs.de/research/decrypting_gsm/.
6. GNU Radio. http://gnuradio.org/redmine/wiki/gnuradio, 1998.
7. 3GPP. 3G Security ; Specification of the A5/3 encryption algorithms for GSM and ECSD, and the GEA3 encryption algorithm for GPRS ; Document 1 : A5/3 and GEA3 specifications (TS 55.216). http://www.3gpp.org/ftp/Specs/html-info/55216.htm.
8. 3GPP. 3G Security ; Specification of the A5/4 Encryption Algorithms for GSM and ECSD, and the GEA4 Encryption Algorithm for GPRS (TS 55.226). http://www.3gpp.org/ftp/Specs/html-info/35226.htm.
9. 3GPP. 3G Security ; Specification of the MILENAGE algorithm set : An example algorithm set for the 3GPP authentication and key generation functions f1, f1*, f2, f3, f4, f5 and f5* ; Document 2 : Algorithm specification (TS 35.206). http://www.3gpp.org/ftp/Specs/html-info/35206.htm.
10. 3GPP. Characteristics of the Universal Subscriber Identity Module (USIM) application (TS 31.102). http://www.3gpp.org/ftp/Specs/html-info/31102.htm.
11. 3GPP. Specification of the SIM Application Toolkit (SAT) for the Subscriber Identity Module - Mobile Equipment (SIM-ME) interface (TS 51.014). http://www.3gpp.org/ftp/Specs/html-info/1114.htm.
12. 3GPP. Specification of the Subscriber Identity Module - Mobile Equipment (SIM-ME) Interface (TS 51.011). http://www.3gpp.org/ftp/Specs/html-info/1111.htm.
13. 3GPP. Universal Subscriber Identity Module (USIM) Application Toolkit (USAT) (TS 31.111). http://www.3gpp.org/ftp/Specs/html-info/31111.htm.
14. Elad Barkan, Eli Biham, and Nathan Keller. Instant ciphertext-only cryptanalysis of gsm encrypted communication. In *CRYPTO*, 2003.
15. Alex Biryukov, Adi Shamir, and David Wagner. Real Time Cryptanalysis of A5/1 on a PC. In *International Workshop on Fast Software Encryption*, 2000.
16. Orr Dunkelman, Nathan Keller, and Adi Shamir. A Practical-Time Attack on the A5/3 Cryptosystem Used in Third Generation GSM Telephony. Cryptology ePrint Archive, Report 2010/013, 2010. http://eprint.iacr.org/.
17. Patrik Ekdahl and Thomas Johansson. Another Attack on A5/1. In *IEEE Transactions on Information Theory*, 2002.
18. Tobias Engel. Locating Mobile Phones using SS7. 25th Chaos Communication Congress, 2009.
19. Commission Nationale Informatique et Liberté. Les applications de géolocalisation sur mobile en questions. http://www.cnil.fr/en-savoir-plus/fiches-pratiques/fiche/article/les-applications-de-geolocalisation-sur-mobile-en-questions/, 2010.
20. ETSI/SAGE. Specification of the 3GPP Confidentiality and Integrity Algorithms 128-EEA3 & 128-EIA3. Document 2 : ZUC Specification. Technical report, ETSI, 2011. http://www.gsmworld.com/documents/EEA3_EIA3_ZUC_v1_5.pdf.
21. ETSI/SAGE. Specification of the 3GPP Confidentiality and Integrity Algorithms 128-EEA3 & 128-EIA3. Document 4 : Design and Evaluation Report. Version 1.3, 18th January 2011. Technical report, ETSI, 2011. http://www.gsmworld.com/documents/EEA3_EIA3_Design_Evaluation_v1_3.pdf.
22. European Information Technology Observatory. More than five billion mobile phone users. http://www.eito.com/pressinformation_20100811.htm, 2010.
23. Grugq. Base Jumping – Attacking the GSM Baseband and Base Station. Blackhat Abu Dhabi, https://media.blackhat.com/bh-ad-10/Grugq/BlackHat-AD-2010-Gurgq-Base-Jumping-slides.pdf, 2010.
24. Xavier Lagrange, Philippe Godlewski, and Sami Tabbane. *Réseaux GSM*. 2000.
25. Karsten Nohl and Luca Melette. GPRS Intercept : Wardriving your country. Chaos Communication Camp, 2011.

26. Karsten Nohl and Sylvain Munaut. Wideband GSM sniffing. 27th Chaos Communication Congress, 2010.
27. Karsten Nohl and Chris Paget. Gsm - srlsy ? In *26th Chaos Communication Congress*, 2009.
28. Orange. Couverture réseau en france. `http://couverture-reseau.orange.fr/france/netenmap.php`.
29. SFR. Couverture réseau en france. `http://assistance.sfr.fr/mobile_forfait/mobile/couverture-reseau/en-48-62267`.
30. Bruno Sido. Rapport d'information sur la couverture du territoire en téléphonie mobile. `http://www.senat.fr/rap/r10-348/r10-3481.pdf`, 2011.
31. Dieter Spaar. A practical DoS attack to the GSM network. DeepSec, 2009.
32. Bouygues Télécom. Couverture réseau en france. `http://www.cartographie.bouyguestelecom.fr/eCouverture/eCouverture.aspx`.
33. Ralf-Philipp Weinmann. All Your Baseband Are Belong To Us – over-the-air exploitation of memory corruptions in GSM software stacks. Laboratory for Algorithmics, Cryptology & Computer Security University of Luxembourg `https://cryptolux.org`, 2010.
34. Harald Welte. OsmocomBB - A tool for GSM protocol level security. In *SSTIC*, 2010.

XSS Test Driver et les navigateurs web sur mobile

Erwan Abgrall[1], Yves Le Traon[2], Sylvain Gombault[3], and Alain Ribault[4]

[1] KEREVAL erwan.abgrall(@)kereval.com
[2] University of Luxembourg yves.letraon(@)uni.lu
[3] Telecom Bretagne sylvain.gombault(@)telecom-bretagne.eu
[4] KEREVAL alain.ribault(@)kereval.com

Résumé Abstract. Les attaques de type Cross Site-Scripting (XSS) constituent l'une des principales menaces pour les applications web. Ces attaques se propagent au travers des différents composants web : le site web, les composants de sécurité internes et externes, et sont in fine exécutées par le navigateur. Tester le degré d'exposition des applications web ou évaluer la nocivité d'un XSS n'est pas simple, dans la mesure où cela dépend beaucoup du navigateur et des vecteurs XSS utilisés pour l'exploiter. Pour rendre un diagnostic précis de l'impact d'un XSS donné, il convient d'identifier précisément la sensibilité de chaque composant à l'attaque, en s'assurant que l'environnement de test n'est pas perturbé les vecteurs XSS eux-mêmes. Dans cet article, nous proposons une méthodologie reposant sur un nouvel outil pour évaluer l'impact des XSS sur les navigateurs : les vecteurs d'attaques XSS sont présentés comme des cas de tests unitaires dont on observe l'effet sur les différents navigateurs web. Notre outil a été testé et validé sur un grand échantillon de navigateurs. Nous présentons ici les résultats de l'étude sur des navigateurs mobiles, elle démontre la pertinence de tester chaque version d'un navigateur avec notre outil, dans la mesure où une nouvelle version peut présenter des vulnérabilités XSS absentes de la version précédente.

1 Introduction

Le Cross-Site Scripting (XSS) est une catégorie d'attaques polymorphiques qui peuvent infecter les applications web comme leurs clients, de façon directe ou indirecte. Beaucoup de contre-mesures sont (ou devraient être) déployées pour contrer cette menace : ces mécanismes de sécurité peuvent être localisés au sein même de l'application (p.ex. des contrôles de validation), ou sur des composants de sécurité externes (reverse-proxies, web application firewalls (WAF) comme ModSecurity[1]), ou encore côté navigateur. Comme le montre la figure 1, un scenario typique d'attaque XSS se déroule en deux étapes principales : insérer un code XSS sur un serveur web, puis propager l'attaque sur les clients en faisant exécuter du code malicieux par leurs navigateurs. Dans ce papier nous allons nous intéresser à la seconde étape et particulièrement à l'exposition des navigateurs aux attaques XSS. Une attaque XSS est composée d'un vecteur (pour pénétrer dans le système) et d'une *payload* qui constitue l'attaque effective. De part la nature dynamique des applications web actuelles, de part la grande variété des navigateurs et de leurs techniques d'interprétation HTML et JavaScript (JS), il

est très complexe d'identifier si un vecteur *passant*(qui entre effectivement dans l'application web) représente une menace ou non pour l'utilisateur.

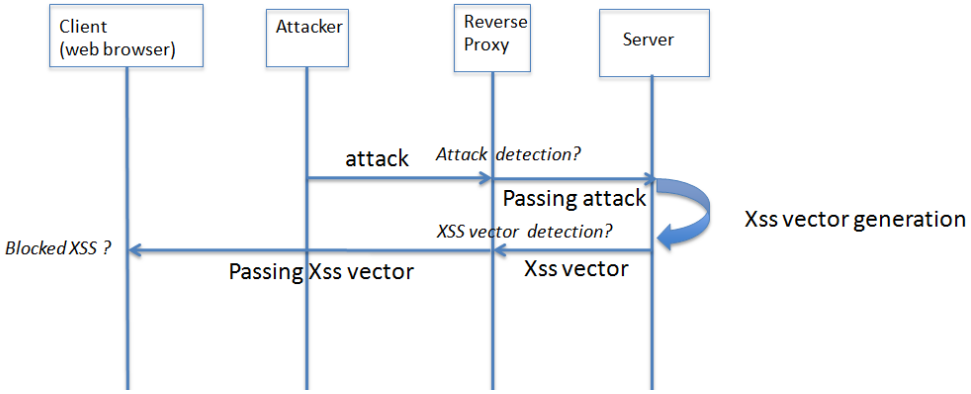


Figure 1. Déroulement d'une attaque XSS

La plupart des mécanismes de sécurité jouent leur rôle face aux attaques XSS basiques, mais peuvent échouer face à des attaques plus sophistiquées. Ces XSS évolués exploitent des comportements rarement connus issus d'interprétations spécieuses de l'HTML ou d'autres ressources web. Pour échapper à la détection d'attaque par recherche de signatures connues, un attaquant cherche à insérer des appels JavaScript (JS) dans des propriétés inattendues de certaines balises telles que :

```
<DIV STYLE="width:expression(eval(
String.fromCharCode(97,108,101,114,116,40,39,120,115,115,39,41,32)
));">
```

Pour comprendre cet exemple, il est important de noter que l'expression d'une propriété CSS (Cascading Style Sheet) exécute du code JS sur un navigateur Internet Explorer (IE). L'expression CSS appelle la fonction JS eval(), qui elle-même appelle la fonction de conversion d'une donnée ASCII en chaîne de caractères et génère la charge utile simple et non destructive suivante :

```
<script>alert(xss)</script>
```

Deux techniques sont principalement utilisées pour bloquer la propagation d'une attaque XSS d'un serveur vers le client : l'approche basée sur des signatures où l'on recherche le pattern du XSS et l'approche utilisée par SWAP [2] faisant analyser les pages web par un navigateur pour en identifier les scripts s'exécutant et les éventuels XSS. La limite connue de la première approche est qu'elle ne

peut bloquer les nouvelles attaques ou un nouveau mécanisme d'évasion (pas de signatures connues). SWAP n'est pas parfait non plus car il ne peut détecter les attaques spécifiques à un navigateur donné, comme l'attaque présenté en [3] ne pouvant être exécutée que par Internet Explorer (IE). En particulier, un moteur de détection XSS basé sur un analyseur HTML et JS donné, ne peut détecter un XSS spécifique à un autre navigateur, dans la mesure où il n'utilise probablement pas exactement le même analyseur HTML que le navigateur ciblé. En conséquence, il est nécessaire de d'étudier l'ensemble des navigateurs web et chacun de manière spécifique pour connaître leur degré d'exposition aux XSS.

Comme le montre notre exemple, une difficulté majeure pour protéger les clients web contre les attaques XSS est la sophistication technique de chaque XSS : des mécanismes HTML et JS peu usuels sont en effet déclenchés. Savoir prévoir quel mécanisme peut être exploité par une attaque n'est pas trivial, la démarche de test prend alors toute sa dimension pour estimer la sensibilité et l'exposition aux attaques XSS de façon systématique. Cependant le test doit répondre à trois exigences : la sélection des cas de tests pertinents, la définition de l'environnement d'exécution des tests et la mesure de la menace réelle portée par le vecteur XSS. La contribution de ce papier est une méthodologie, basée sur l'outil XSS Test Driver, pour tester de façon systématique l'impact d'un large jeu de vecteurs XSS sur les navigateurs web, y compris les clients sur téléphones ou tablettes mobiles.

La suite de cet article s'articule en trois parties. Le chapitre 2 présente les travaux relatifs aux tests XSS et les limitations des solutions existantes, puis décrit la nouvelle approche et la « bonne pratique »proposée. Le chapitre 3 décrit notre outil XSS Test Driver et montre qu'il s'agit d'un environnement sécurisé et isolé, permettant de tester un large jeu de vecteurs que la communauté des utilisateurs peut enrichir. Le chapitre 4 valide la pertinence de l'outil et de son approche en comparant le degré de nocivité des différents jeux de tests de XSS ainsi que le degré d'exposition des navigateurs mobiles, permettant ainsi de déterminer si un vecteur XSS passant au travers d'une application web représente une réelle menace pour les clients habituels de l'application web.

2 Tests de vecteurs de XSS et sécurité

Les tests de sécurité sont un vaste sujet, tant par le large spectre de domaines à prendre en compte qu'en fonction du type de propriété ciblée. Bien que les attaques XSS soient un sujet souvent abordé, la dimension test l'est, elle, beaucoup moins. L'analyse d'une application web conduit à considérer 3 couches pour le déploiement des contre-mesures : le navigateur web client, la sécurité côté serveur et les composants intermédiaires. A l'intérieur de chacun de ces niveaux, différents composants de sécurité peuvent être déployés pour examiner le trafic.

La combinaison des protections sur chaque couche permet d'améliorer la sécurité globale : toutefois, chaque couche constitue un élément qui peut présenter à lui seul des vulnérabilités face aux scenarii XSS. Chaque couche doit donc être testée indépendamment des autres. Le niveau de menace d'un scenario d'attaque XSS donné étant directement lié à la nature et à la distribution des navigateurs sur le site web client, il est très important d'évaluer chaque navigateur. Dans la suite de ce chapitre, nous détaillons la méthodologie proposée : elle repose sur l'utilisation de vecteurs XSS comme cas de test unitaire et a pour objectif d'étudier tous les navigateurs connus.

2.1 Études relatives

D'après nos recherches, aucune étude antérieure ne traite la façon de sélectionner et de comparer automatiquement un jeu de tests XSS. Toutefois plusieurs travaux, dont certains des auteurs eux-mêmes, proposent des méthodes et des outils pour tester automatiquement les politiques de sécurité (politiques de contrôle d'accès) [4],[5],[6],[7]. D'autres proposent des frameworks et des techniques pour tester le système à partir de ses interfaces [8],[9]. Offut propose une approche de test par bypass[4],[10] plus proche des techniques de tests XSS que nous présentons. La démarche décrite dans notre article en suit les grandes lignes, mais en apportant un éclairage spécifique sur la sélection des tests XSS et un comparatif systématique au travers de ces tests (et ce, sans évincer le navigateur des tests, dans la mesure où il s'agit d'une cible XSS à part entière). De façon similaire à la méthode proposée par Su's [11], Yao-Wen et al. [12] proposent de muter et d'injecter des entrées erronées, dont des injections SQL et XSS dans les applications web (outil WAVE), mais ils ne fournissent pas de techniques de diagnostic pour distinguer l'impact sur les différentes couches et valider la capacité d'un vecteur XSS à les traverser jusqu'au navigateur web.

La seule méthodologie d'évaluation de cas de test XSS que nous avons trouvée est basée sur du test de mutation [13] : un jeu de données de test est qualifié en mutant le code PHP de cinq applications web. Les attaques XSS sont alors utilisées pour tuer les mutants. Mais cette étude ne tient pas compte de l'impact du navigateur sur l'efficacité d'un vecteur XSS, introduisant de ce fait un biais dans l'expérimentation. Des sources similaires aux nôtres sont utilisées pour les vecteurs XSS, mais sans adaptation à un point d'injection spécifique. Cette pratique introduit un biais dans l'efficacité de l'attaque. Les attaques doivent en effet être ajustées à chaque point d'injection. Un même vecteur peut n'avoir aucun effet sur un point d'injection donné, et réussir sur un autre. La plupart des études XSS se concentrent, soit sur la détections d'attaques XSS [2], [3], [14], soit sur la recherche de vulnérabilités XSS [15], [16]. D'autres études se penchent sur les vulnérabilités ou les vers XSS [17], [18]. Enfin, une étude très complète de l'état de l'art sur les problématiques XSS et les parades est disponible en [19].

2.2 Méthodologie de test

Comme le montrent la figure 1 et notre premier exemple, un scénario d'attaque XSS se divise en deux étapes : l'attaquant adapte son attaque pour exploiter une vulnérabilité située sur un serveur web ou une application web pour envoyer au navigateur un vecteur XSS avec une charge utile (payload). La charge utile contient en général du code JavaScript à exécuter par le navigateur. Elle peut être inoffensive, comme dans notre exemple, ou nocive, en redirigeant vers un site malveillant, pour exploiter une faille dans le navigateur, menant ainsi à l'exécution de code arbitraire sur le système client, comme pendant l'attaque Aurora visant les employés de Google [20]. La charge utile est exécutée si le navigateur comprend le vecteur XSS, le succès d'une attaque par XSS dépend donc aussi du navigateur utilisé par le mobile, téléphone ou tablette.

Comme nous voulons évaluer la capacité de n'importe quel navigateur à exécuter ou non la charge utile contenue dans un vecteur XSS, nous avons décidé de supprimer la première partie d'une attaque XSS complète, notre outil jouant le rôle de serveur web. Dans notre contexte, un cas de test est composé d'un vecteur d'attaque transportant une charge utile non destructive.

Comme présenté figure 2, un cas de test échoue (FAIL) si le navigateur n'exécute pas la charge utile, ou s'il attend indéfiniment, empêchant l'exécution du JavaScript et donc l'attaque. Un cas de test réussit (PASS) si le navigateur exécute la charge utile. Cela signifie qu'un cas de test qui réussit reflète une réelle menace pour le navigateur. PASS veux donc dire faille de sécurité (alors qu'en général dans le domaine du test PASS signifie qu'il n'y a pas de problème détecté).

Si l'on exécute les cas de tests sur un jeu prédéfini de navigateurs, le résultat est une liste contenant les vecteurs qui s'exécutent (tests qui réussissent) par navigateur. Comme XSS Test Driver agit comme un serveur web, les navigateurs doivent s'y connecter pour lancer la suite de tests. Les tests unitaires XSS ont été recueillis sur divers sites web de la communauté sécurité [21],[22],[23].

Ayant plus de 80 vecteurs d'attaques XSS, les principales difficultés rencontrées dans le développement de XSS Test Driver ont été :

1. l'adaptation de la charge utile à tous les navigateurs ;
2. l'adaptation de la charge utile à tous les vecteurs ;
3. l'automatisation des tests, afin d'exécuter tous les tests en une session pour chaque navigateur.

3 XSS Test Driver : un outil léger pour la sélection et l'exécution de vecteurs de XSS

Ce paragraphe présente dans le détail XSS Test Driver. Pour atteindre notre but, nous avons développé un environnement léger, mais non moins efficace, en

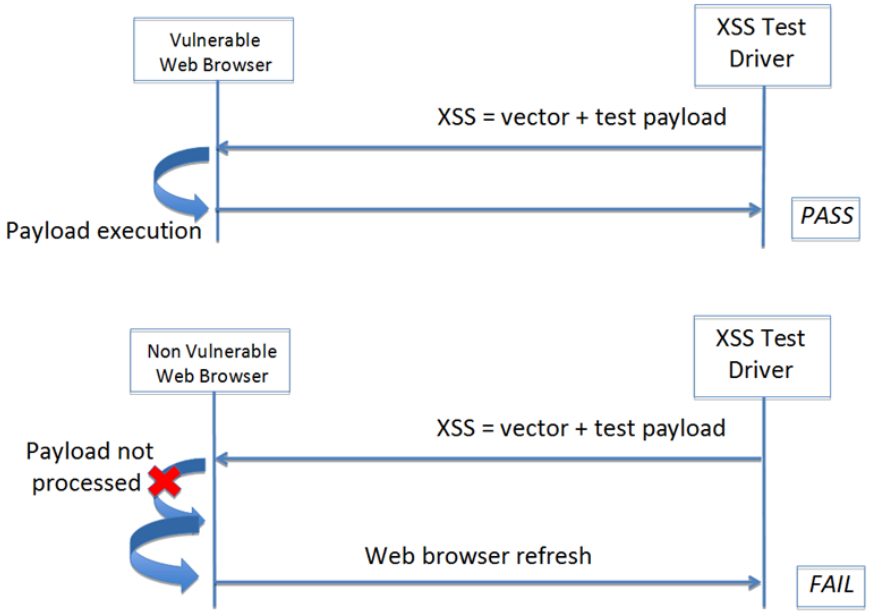


Figure 2. Logique d'un test unitaire de XSS Test Driver

python. Celui-ci fournit les vecteurs XSS aux navigateurs, et peut ainsi obtenir un verdict sur l'exécution du XSS. XSS Test Driver permet de mesurer l'exposition d'un navigateur à un ensemble donné de vecteurs XSS. Côté navigateur, l'exécution d'un XSS se fait en deux temps : le navigateur analyse le code HTML, permettant ainsi de reconnaitre les parties du Document Object Model et d'en construire une représentation interne. Il appelle ensuite le code JavaScript identifié (situé entre des balises <script> ou dans les propriétés de certaines balises) et l'exécute si nécessaire (ce n'est pas toujours le cas, par exemple les propriétés onevent comme onload ou onmouseover ne sont pas exécutées à chaque fois). Une démonstration de XSS Test Driver est disponible sur internet [24], vous pouvez y tester le navigateur de votre téléphone ou de votre tablette.

3.1 Exigences dans le déroulement des tests

L'objectif principal d'une attaque XSS est d'exécuter du JavaScript dans une page web, et l'une des difficultés est de s'assurer que les mécanismes sous-jacents déclenchés par les vecteurs XSS ne peuvent avoir un effet de bord sur ceux utilisés pour les tests ou le monitoring d'un système sous test.

Pour répondre aux problématiques de test standard et à l'objectif spécifique des tests de XSS, notre outil doit répondre à trois exigences. La première est que le JavaScript doit s'exécuter à l'intérieur du navigateur testé, ce qui semble

évident, car le vecteur XSS a pour cible le navigateur, mais certains outils de test incorporent leur propre environnement d'exécution JavaScript pour les tests, comme RhinoUnit[25] par exemple. Un environnement de test basé uniquement sur un tel moteur néglige l'interprétation du HTML faite par le navigateur, et n'est pas capable d'analyser les vecteurs XSS basé sur HTML. Le deuxième prérequis est la capacité à exécuter une fonction de rappel qui sera déclenchée à partir de la charge utile pour valider l'exécution du XSS par le navigateur : cela est nécessaire pour informer l'oracle du succès de l'attaque. Enfin, dernière exigence, l'outil doit avoir un contrôle total sur le DOM fourni au navigateur, car l'exécution du XSS dépend de la manière dont le HTML est analysé et interprété. Cette dernière exigence est atteinte par la conception des cas de tests qui spécifient le vecteur XSS qui est fourni au navigateur.

3.2 Logique de test

Pour éviter l'utilisation de bibliothèques JavaScript, ou toute interaction avec le DOM, nous avons utilisé la logique suivante pour enchaîner les tests et récupérer les résultats :

1. chaque attaque XSS est servie par une URL différente, avec une charge utile indépendante de toute bibliothèque JavaScript ;
2. la charge utile d'une attaque XSS contient une routine de validation JavaScript et un mécanisme de redirection pour passer au test suivant ;
3. quand le mécanisme de validation est déclenché, le test est marqué comme réussi ;
4. quand l'URL d'un test est rafraîchie, le serveur vérifie si l'une des routines de validation a été exécutée, sinon le test est marqué comme échoué ;
5. dans les deux cas (réussi/échoué), un message 302 Redirect est envoyé au navigateur pour le rediriger vers le test suivant.

3.3 Format de test et charge utile

Les cas de test sont fournis sous forme de tuples python, constitués du vecteur avec le format de la charge utile en paramètre et de sa description :

```
("""<script>%(payload)s</script>""","basic script payload")
```

Une suite de tests est composée d'une simple liste de cas de tests à enchaîner. Elle est traitée par XSS Test Driver qui construit les vecteurs XSS et les envoie au navigateur. La charge utile est générée lorsque le test est appelé. Elle contient une routine de redirection vers une adresse de validation, ainsi qu'une

requête `XMLHttpRequest` vers une autre adresse de validation pour les navigateurs qui bloquent les redirections JavaScript. En effet, alors que certains navigateurs bloquent les redirections JavaScript, ils ne bloquent pas les appels `XMLHttpRequest` . Une technique de validation basée sur les cookies a été ajoutée pour prendre en compte les anciens navigateurs ne supportant pas les requêtes `XMLHttpRequest`, car ils supportent tous au moins les cookies.

Nous utilisons deux types de charges utiles : celles que nous avons déjà décrites et une générique contenant `alert('xss')`. Cette dernière est envoyée pour pouvoir confirmer manuellement l'exécution d'un vecteur.

Plusieurs formats de charge utile sont disponibles pour couvrir les besoins de vecteurs spécifiques : quelques attaques XSS nécessitent de présenter le JavaScript dans un fichier spécifique pour tromper certains navigateurs (IE6 dans ce cas) :

```
<LINK REL="stylesheet" HREF="http://ha.ckers.org/xss.css">
```

Les formats supportés sont :

- payload : code JavaScript à exécuter
- jscript : adresse pour charger un .js contenant la charge utile
- eval_payload : charge utile XSS fournie sous la forme d'une fonction `eval(String.[‘fromCharCode’](XX,XX,XX))`
- scriptlet : XSS dans une petite page HTML
- css : XSS dans un css
- htc : XSS dans un fichier .htc
- jpg : code JavaScript dans un `.jpg` servi comme un fichier jpeg

3.4 Automatisation des tests

Les cas de tests peuvent s'enchaîner automatiquement : une balise

```
<meta http-equiv="refresh" content="5" />
```

est ajoutée au vecteur pour rafraîchir la page. L'enchaînement des tests peut néanmoins s'interrompre si ce mécanisme entre en conflit avec quelques rares vecteurs XSS, comme par exemple :

```
<META HTTP-EQUIV="refresh" CONTENT="0;url=javascript:alert('XSS');">
```

Quand cette attaque XSS échoue, le navigateur peut être bloqué avec une adresse invalide, nécessitant alors une intervention humaine : naviguer vers une URL de reprise des tests. En fait, le W3C déconseillant l'usage de la méthode meta-refresh, le seul mécanisme permettant d'enchainer les tests n'est donc pas compatible avec tous les navigateurs [26] (alors que la navigation vers une URL de reprise l'est).

Table 1. User Agents Identification

Browser	User Agent String
Chrome 11.0.696.68	`Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/534.24 (KHTML, like Gecko) Chrome/11.0.696.68 Safari/534.24`
Firefox 7	`Mozilla/5.0 (X11; Linux i686; rv:7.0.1) Gecko/20100101 Firefox/7.0.1`
Safari Mac OS X Leopard	`Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10_5_8; fr-fr) AppleWebKit/533.21.1 (KHTML, like Gecko) Version/5.0.5 Safari/533.21.1`
IE 8.0.6001.19048	`Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.0; WOW64; Trident/4.0; SLCC1; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET4.0C; .NET4.0E; .NET CLR 3.0.30729)`

3.5 Identification des navigateurs et collecte des résultats

En tant que serveur web, notre outil peut identifier chaque navigateur connecté via son champ user-agent : Cette information permet à un serveur web d'adapter son comportement à chacun de ses clients ; elle va permettre à notre outil de reconnaître chaque type de navigateur et ses multiples versions.

4 Expérimentations et détermination de l'exposition des navigateurs webs face à un ensemble de vecteurs de XSS

Nous avons choisi d'illustrer les résultats bruts de notre outil avec deux indicateurs pour simplifier l'analyse des résultats. Il s'agit du niveau de nocivité de chaque cas de tests XSS à l'encontre d'un ensemble de navigateurs. En complément nous calculons le niveau d'exposition d'un navigateur à l'ensemble des cas de tests.

4.1 Définitions des indicateurs

Soit *Ts* une suite de tests. Soit *Verdict(Tc, Wb)* le résultat de l'exécution du cas de test *Tc* provenant de *Ts*, contre le navigateur web *Wb*. *Verdict(Ts, Wb)* retourne soit *PASS* (le XSS a fonctionné) ou *FAIL*.

Le *niveau de nocivité Nox(Ts, Wb)* de la suite de tests *Ts*, contre un ensemble de navigateurs web *WB*, est défini par le pourcentage de tests retournant *PASS* sur un ensemble de navigateurs web testé.

(1) $Nox(Ts, Ws) = |Verdict(Tc, Wb) = Pass, Wb \in Wb|/|Wb|$

La surface d'attaque *ThExp(Wb, Ts)* d'un navigateur web à la suite de tests XSS *Ts* est le pourcentage de résultats *PASS* lorsque nous l'utilisons avec l'ensemble des éléments de *Ts* :

(2) $ThExp(Wb, Ts) = |verdict(Tc, Wb) = Pass, Tc \in Ts|/|Ts|$

Ces indicateurs sont utilisés pour simplifier l'analyse des résultats et déterminer quels vecteurs sont les plus efficaces pour une catégorie de navigateurs web, et d'identifier rapidement parmi ceux étant les plus exposés. Néanmoins, ces estimations ne remplacent pas une analyse plus précise des attaques affectant un navigateur web précis. Par exemple, nos résultats montrent que certaines attaques ne réussissent que sur certains navigateurs, qui ne sont pas nécessairement les plus exposées.

4.2 Conditions d'expérimentation

Classificaton des vecteurs d'attaque Les sources de vecteurs d'attaque sont variées. Beaucoup proviennent de l'étude des normes et des comportements de navigateurs, mais une partie d'entre elles est directement issue d'éléments non documentés, tels que des comportements particuliers comme l'interprétation spécifique de l'échappement d'une chaine de caractère, etc... Donc si nous utilisons comme référence une norme, nous pouvons classer les vecteurs par rapport à celle-ci, permettant ainsi la génération de vecteurs XSS basés sur des modèles. Les autres vecteurs ne respectant pas la norme, tombent finalement dans la catégorie des bugs et défauts applicatifs qui se doivent d'être corrigées.

Génération de nouveaux vecteurs d'attaques XSS Afin de découvrir de nouveaux vecteurs, nous avons combiné des ensembles de balises et propriétés d'HTML4 avec des appels JavaScript, et utilisé le produit scalaire de ces ensembles (balise, propriété, appel) en vue de générer des vecteurs XSS. Avec cette

approche, sur les 44000 cas de tests générés, nous n'avons trouvé que quelques vecteurs fonctionnels, ainsi que des variations de quelques vecteurs déjà connus. Les interdépendances entre les balises et les propriétés ne sont pas prises en compte dans cette génération, et quelques contraintes se doivent d'être respectées pour obtenir un vecteur valide, certaines propriétés étant simplement ignorées par les analyseurs syntaxiques de par leur contexte d'utilisation (par exemple l'utilisation de balises svg ou HTML5 sans le bon type mime spécifié dans le content-type).

Avec une modélisation correcte de ces contraintes, il serait possible de générer tous les vecteurs d'attaques pour une norme donnée, et de s'en servir pour évaluer les techniques de détections de XSS. Il serait également possible d'utiliser ce modèle pour détecter des attaques, ou pour améliorer les IDS actuels. Ces travaux feront l'objet d'un autre article.

4.3 Choix des cas de tests

Du point de vue d'un navigateur, la sécurité de l'ensemble des composants du navigateur importe, alors que d'un point de vue XSS, seul l'analyseur syntaxique du code HTML importe : ainsi Safari, Chrome et les autres navigateurs bases sur webkit ont globalement le même comportement avec nos cas de tests. Les navigateurs testés furent : Internet Explorer, Netscape, Mozilla, Firefox, Opera, Safari and Chrome, dans des versions publiés entre juillet 1998 et mars 2011.

Les vecteurs XSS furent construit à partir du « XSS Cheat Sheet »[21], du « HTML5 Security Cheat Sheet »[22] and « UTF-7 XSS Cheat sheet » [27], et quelques vecteurs découverts en générant les triplettes balise/propriété/appel JavaScript qui firent leur apparition plus tard dans [22].

Les cas de tests XSS représentent un grand nombre de vecteurs d'attaques. Nous les avons adaptés pour avoir une charge utile dédiée à l'interprétation des résultats. Dans le but de reproduire ces essais, tous les cas de tests sont disponibles en ligne. Une nouvelle version de l'outil est disponible sur `http://xss.labosecu.rennes.telecom-bretagne.eu` et est en cours d'amélioration avec pour objectif de simplifier le traitement des résultats.

Certains vecteurs ont réussi sur la majorité des navigateurs, alors que d'autres ne fonctionnent que sur des versions bien précises. Ceci est dû à la qualité de l'implémentation des normes (et de l'analyseur syntaxique sous-jacent). Par exemple, il y a eu un effort important de respect de la norme entre IE6 et IE7.

Nous pouvons aisément imaginer une charge utile ne visant qu'une version spécifique d'un navigateur, l'identification du navigateur se faisant par l'interprétation (ou non) du vecteur d'attaques XSS. Le comportement de l'analyseur syntaxique devient alors une empreinte, de la même manière que l'empreinte d'une pile réseau peut être obtenue et servir de préliminaire à une attaque

4.4 Étude des résultats de tests

Les tableaux suivants (Table 2) listent les résultats de 84 vecteurs XSS de test pour 3 familles de navigateurs. La table de droite est la suite de la table de gauche. Dans la table de gauche, nous présentons les résultats des cas de tests 3 à 45, la seconde table présentant quant à elle les résultats des tests 45 à 87 (le résultat 45 étant répété pour des raisons de présentation). En ordonnées nous trouvons les 3 familles de navigateurs, et pour chacun d'eux nous affichons en première ligne le niveau d'exposition à la menace (ex : 33 pour IE8 signifiant 33 % de niveau d'exposition à la menace). Le niveau de nocivité pour chaque cas de tests est donné dans la dernière colonne à droite. Nous fournissons ces niveaux pour tous les navigateurs testés. La table montre donc à la fois le niveau d'exposition de chaque navigateur et le niveau de nocivité potentiel de chaque vecteur. La table montre un moyen simple de sélectionner un sous-ensemble de navigateurs permettant un maximum d'attaques.

Nous pouvons utiliser cette matrice pour sélectionner les cas de tests qui peuvent être utilisés pour tester une certaine catégorie de navigateurs web. Par exemple les cas de tests 7, 8, 9, 10, 14, 18... ne sont pas nocifs pour les navigateurs web modernes. Quelques cas de tests ont un niveau de nocivité de 0 %, ce qui signifie qu'ils sont inutiles pour l'ensemble des navigateurs choisis pour cet article. Si nous corrélons ces résultats avec les statistiques provenant de sites web (comme celles publiées sur W3School [26]), nous pouvons aisément déterminer le niveau de menace d'un vecteur XSS donné transitant par un site web. Le développement rapide des navigateurs actuels rend complexe la tâche de suivi de l'efficacité d'un vecteur XSS, et lorsqu'un nouveau vecteur est découvert, il est fastidieux de le tester sur de nombreux navigateurs. XSS Test Driver répond à ces problématiques tout en facilitant les comparaisons.

Table 2. Résultats d'exécution de 84 vecteurs par des navigateurs Récents, Anciens et Mobiles

Vector / Browser	Chrome 11.0.696.68	IE 8.0.6001.19048	Opera mobile 11	Opera 11.11 rev2109	ie mobile	Safari Mac OSX	iPhone 3GS	Android 2.2	Firefox 5 Android	Firefox 8.0a1	IE 6.0.2900.2180	Firefox 2.0.0.2	Netscape 4.8	ie 4.01	opera 4.00
3	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
4	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1
5	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1
6	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
7	0	0	0	0	1	0	0	0	0	0	1	0	1	1	0
8	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
9	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0
10	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
11	1	1	0	0	1	1	1	1	1	1	1	1	0	1	0
12	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
13	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
14	0	0	0	0	1	0	0	0	0	0	1	0	1	1	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	1	1	1	1	1	0	0	0	1	1	1	1	0	0	0
17	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
18	0	0	0	0	1	0	0	0	0	0	1	0	1	1	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0
21	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0
22	0	0	0	0	1	0	0	0	0	0	1	0	1	1	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	1	0	0	0	0	0	1	0	1	1	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
29	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
31	1	0	1	1	0	1	1	1	0	0	0	1	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33	1	1	0	0	1	1	1	1	1	1	1	1	0	1	0
34	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0
35	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0
36	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0
37	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0
38	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0
39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
41	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0
42	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0
43	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0
44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Vector / Browser	Chrome 11.0.696.68	IE 8.0.6001.19048	Opera mobile 11	Opera 11.11 rev2109	ie mobile	Safari Mac OSX	iPhone 3GS	Android 2.2	Firefox 5 Android	Firefox 8.0a1	IE 6.0.2900.2180	Firefox 2.0.0.2	Netscape 4.8	ie 4.01	opera 4.00
45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
46	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
47	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
48	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0
49	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
50	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
51	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
52	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
53	1	0	0	0	0	1	1	1	1	1	0	0	0	0	0
54	1	0	0	0	0	1	1	1	1	0	0	0	0	0	0
55	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
56	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
57	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
58	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
59	1	0	1	1	0	1	1	1	1	1	0	0	0	0	0
60	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
61	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
62	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
63	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
64	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
65	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
66	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
67	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
68	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0
69	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
70	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0
71	0	1	0	0	1	0	0	0	0	0	1	0	0	1	0
72	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
73	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
74	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
75	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
76	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0
77	0	1	0	0	1	0	0	0	0	0	1	0	0	1	0
78	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
79	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
80	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
81	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
82	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
83	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0
84	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
85	0	0	0	0	0	0	0	1	1	1	0	0	1	1	0
86	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
87	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Résultats pour les navigateurs modernes Avec les navigateurs récents considérés, 32 des 84 cas de test sont concluants, c'est-à-dire que les vecteurs XSS sont effectivement exécutés au sein de l'outil.

Le comportement de Safari et Chrome pour les 84 cas de test est identique excepté pour les tests #16 et #83. Ceci s'explique par l'utilisation de Apple Webkit 534.3 comme moteur de rendu pour Chrome et l'utilisation de la version 531.22.7 (version indiquée par le User-Agent) pour Safari. Confirmant la prépondérance du moteur HTML dans l'execution des vecteurs de XSS.

Peu de cas de test sont efficaces sur tout le panel des navigateurs. Les vecteurs 3 à 6 sont des balises `<script>` standard basées XSS contenant différentes charges utiles. Les tests #12,#13 et #15 sont des balises `<body>` basées XSS avec des OnLoad events pour exécuter les charges utiles. Le test #17 est une balise avec double chevrons pour échapper aux filtres classiques. Enfin, les données du test #19 offrent une forme très intéressante d'évasion basée sur une balise semi-ouverte `<iframe>` chargeant la payload d'une page HTML dédiée : `<iframe src=/inc/16/payload.html <`

Résultats pour les navigateurs mobiles Pour les navigateurs mobiles, 43 des 84 cas de test sont concluants, avec cependant des comportements sensiblement différents des navigateurs web modernes. Si l'on compare les résultats des 2 navigateurs basés sur Webkit, Safari Mobile et Chrome, les résultats sont similaires car ils sont tous deux basés sur webkit.

Comparaison des versions mobile et standard

La comparaison de la version mobile et standard d'une même famille de navigateurs fait apparaître de légères différences comme entre Opera Mobile et Desktop ou Firefox 4 Mobile et Desktop (Table 3) par exemple.

Entre Opera mobile et standard, seule l'exécution du vecteur n°53 suivant met en évidence une différence de comportement :

```
<input onfocus=javascript:eval(String['fromCharCode']
(97,108,101,114,116,40,39,120,115,115,39,41,32)) autofocus>
```

Dans la mesure où ils embarquent le même moteur presto, ils reconnaissent les mêmes vecteurs mais la navigation mobile induit une implémentation différente des evénements Javascript, onfocus dans notre cas. On remarquera qu'une légère différence est aussi présente entre la version desktop et la version « mobile emulator ».

On peut observer la même différence de comportement entre les versions mobile et standard de Firefox.

Table 3. Comparaison Mobiles & Desktop

Comparatif entre Navigateurs Standards & Mobiles

Vector	/ Browser	Opera 11.11 windows	Opera mobile 11 Android	Opera mobile Emulator	Opera Archos edition
3		1	1	1	1
4		1	1	1	1
5		1	1	1	1
6		1	1	1	1
7		0	0	0	1
8		0	0	0	1
11		0	0	0	1
12		1	0	1	1
13		1	1	1	1
14		0	0	0	1
16		1	1	1	0
17		1	1	1	1
18		0	0	0	1
20		0	0	0	1
23		0	0	0	1
26		0	0	0	1
31		1	1	1	0
33		0	0	0	1
34		0	0	0	1
35		0	0	0	1
46		0	0	0	1
50		1	1	1	1
53		1	0	0	1
54		0	0	0	0
56		1	0	0	0
59		1	1	1	0
61		0	1	1	0
64		1	1	1	1
76		1	1	1	1
83		0	0	0	0
85		0	0	0	1

Vector	/ Browser	Firefox 4.0.1	Firefox 4.0.2 Android.....
3		1	1
4		1	1
5		1	1
6		1	1
7		0	0
8		0	0
11		1	1
12		1	1
13		1	1
14		0	0
16		1	1
17		1	1
18		0	0
20		0	0
23		0	0
26		0	0
31		0	0
33		1	1
34		0	0
35		0	0
46		0	0
50		1	1
53		1	1
54		0	1
56		1	1
59		1	1
61		0	0
64		1	1
76		1	1
83		0	1
85		0	1

Comparatif IE 6 mobile & desktop

Vector	/ Browser	IE mobile	IE 6.0.2900.2180
3		1	1
4		1	1
5		1	1
6		1	1
7		1	1
9		1	1
11		1	1
12		1	1
13		1	1
14		1	1
16		1	1
17		1	1
18		1	1
20		1	1
21		1	1
22		1	1
23		0	0
26		1	1
27		0	0
33		1	1
34		1	1
35		1	1
36		1	1
37		1	1
38		1	1
40		0	0
41		1	1
42		1	1
43		1	1
44		0	0
48		1	1
49		0	1
50		1	1
51		0	0
64		1	1
65		0	1
66		0	1
67		0	0
68		1	1
69		0	0
70		1	1
71		1	1
72		0	0
76		1	1
77		1	1
78		0	0
83		1	1

Comparatif entre Navigateurs Mobiles

Vector	/ Browser	Firefox 4.0.2 Android	Opera mobile 11 Android	ie mobile	n810 tablet browser	iPad 2	Nokia E65	archos 5 internet tablet	iPhone 3GS	Android 2.2 Htc desire z	Android 3.1 GALAXY TAB
3		1	1	1	1	1	1	1	1	1	1
4		1	1	1	1	1	1	1	1	1	1
5		1	1	1	1	1	1	1	1	1	1
6		1	1	1	1	1	1	1	1	1	1
7		0	0	1	0	0	0	0	0	0	0
9		0	0	1	0	0	0	0	0	0	0
11		1	0	1	1	1	0	1	1	1	1
12		1	0	1	1	1	1	1	1	1	1
13		1	1	1	1	1	1	1	1	1	1
14		0	0	1	0	0	0	0	0	0	0
16		1	1	1	1	0	0	0	0	0	1
18		0	0	1	0	0	0	0	0	0	0
20		0	0	1	0	0	0	0	0	0	0
21		0	0	1	0	0	0	0	0	0	0
22		0	0	1	0	0	0	0	0	0	0
26		0	0	1	0	0	0	0	0	0	0
31		0	1	0	0	1	1	1	1	1	1
33		1	0	1	1	1	0	1	1	1	1
34		0	0	1	0	0	0	0	0	0	0
35		0	0	1	0	0	0	0	0	0	0
36		0	0	1	0	0	0	0	0	0	0
37		0	0	1	0	0	0	0	0	0	0
38		0	0	1	0	0	0	0	0	0	0
41		0	0	1	0	0	0	0	0	0	0
42		0	0	1	0	0	0	0	0	0	0
43		0	0	1	0	0	0	0	0	0	0
46		0	0	0	0	0	0	0	0	1	0
48		0	0	1	0	0	0	0	0	0	0
50		1	1	1	1	1	1	1	1	1	1
53		1	0	0	0	1	0	1	1	1	1
54		1	0	0	0	1	0	1	1	1	1
56		1	0	0	0	0	0	0	0	0	1
59		1	1	0	0	1	0	0	1	1	1
61		0	1	0	0	0	0	0	0	0	0
64		1	1	1	1	1	1	1	1	1	1
68		0	0	1	0	0	0	0	0	0	0
70		0	0	1	0	0	0	0	0	0	0
71		0	0	1	0	0	0	0	0	0	0
74		0	0	0	0	0	0	0	0	1	0
76		1	1	1	1	1	1	1	1	1	1
77		0	0	1	0	0	0	0	0	0	0
83		1	0	1	1	1	0	1	1	1	1
85		1	0	0	0	0	0	0	0	1	0

Résultats pour les anciens navigateurs Comme nous pouvons le constater, bien qu'encore largement utilisé dans le monde de l'entreprise, IE6 présente la plus haute exposition aux menaces, avec 45 % des vecteurs interprétés. L'exposition spécifique d'une entreprise peut être calculée à partir de sa population de navigateurs web (obtenue à partir d'un inventaire software ou de statistiques internes) et pondérée par la répartition de ses navigateurs. Sachant cela, et en utilisant le bon sous-ensemble de vecteurs d'attaques XSS, il est facile d'évaluer un intranet face aux menaces XSS et de déterminer précisément les risques associés à une vulnérabilité XSS donnée. Cette démarche apporte des informations concrètes pour prioriser les solutions à mettre en place pour les vulnérabilités détectées. Enfin, les vecteurs basés sur des propriétés et balises HTLM5, sont de fait, sans effet sur les navigateurs anciens.

Pourquoi certains vecteurs ne sont jamais exécutés ? Les résultats permettent d'observer que 16 vecteurs ne sont exécutés par aucun des navigateurs testés, ce qui s'explique par les raisons suivantes :

- des vecteurs très spécifiques à des versions précises des navigateurs, comme le test #7 issu de Xss Cheat Sheet [23], ciblant seulement les versions Firefox 2.0 et Netscape 8.1 basées sur le moteur Gecko. Firefox 2.0.0.6 et Netscape 8.1.2 avaient fait l'objet de la dernière correction.
- des erreurs dues à des contextes de test inappropriés, comme le charset utilisé pour la suite de test ou une mauvaise DTD ou encore un content-type mal défini, montrent que selon les cas, les vecteurs peuvent être dépendants ou non du contexte de test.
- des vecteurs pouvaient rendre les navigateurs instables, voire conduire au crash, comme
  `<DIV STYLE="width:expression(eval(String['fromCharCode']`
  `(97,108,101,114,116,40,39,120,115,115,39,41,32)));">`
  , qui plongeait IE dans une boucle d'attente du serveur.

Les travaux sur l'amélioration de la gestion des contextes des tests ont rapidement fait apparaître la problématique d'explosion combinatoire des cas de test. L'utilisation de stratégies de génération de cas de test, dont la méthode Pair Wise a permis de restreindre le jeu de vecteurs de test.

4.5 Tests de non-régression sur les appels JavaScript

Pour le W3C, les efforts de validation se font sur le formatage du HTML, mais très peu sur le comportement des navigateurs. La raison principale vient du coût des tests sur les navigateurs, lié à la difficulté d'automatisation. XSS Test Driver répond à cette problématique en proposant un outil et une méthodologie de test de non régression des appels à JavaScript, permettant de en évidence pour chaque navigateur :

– L'évolution dans le temps de sa robustesse face aux XSS
– La stabilité de son comportement d'une version à l'autre

La figure 3 présente l'évolution de ses caractéristiques pour Opera, qui propose une nouvelle version de son navigateur tous les six mois. Le nombre des vecteurs XSS exécutés est indiqué dans les colonnes sombres. Le nombre d'évolutions entre deux versions est indiqué dans les colonnes grises et est construit de la façon suivante :

PASS(n) représente le jeu de vecteurs XSS passant pour une version n d'un navigateur Web *wb*, *n-1* représentant la version précédente de *wb*.

(3) $PASS(n) = tc/verdict(tc, wb_n) = Pass, tc \in TS,$

Delta représente le nombre d'évolutions entre deux versions de *wb* :

(4) $Delta(n, n-1) = |PASS(n) \cup PASS(n-1) - PASS(n) \cap PASS(n-1)|$

Entre Opera 10.50 *(n)* et 10.10 *(n-1)*, alors que le nombre de vecteurs passant est proche (23 et 17), les évolutions *Delta (10.50, 10.10)* sont importantes (17). Ceci met en évidence une forte instabilité entre ces deux versions mineures plutôt qu'un comportement stabilisé. Cela révèle aussi un manque de tests de non régression systématique d'une version à l'autre. Ces évolutions ne peuvent s'expliquer par les seules nouveautés d'implémentation de la norme HTML. XSS Test Driver offre donc une solution pour systématiser les tests de non régression et de nouvelles opportunités de recherche dans le domaine du test et du diagnostic des navigateurs web avec un historique des différentes versions.

5 Conclusion et perspectives

Cet article a présenté une méthodologie et un outil pour évaluer de façon précise le comportement des navigateurs web sur mobiles face aux attaques XSS. XSS Test Driver est le Framework dédié à cette approche. La sélection des cas de test XSS pertinents pour chaque navigateur permet de lancer le bon vecteur XSS contre la bonne cible.

Pour démontrer la faisabilité de l'approche, un jeu de test XSS a été lancé contre trois types de navigateurs différents ; les anciens, les modernes et les navigateurs mobiles.

Les résultats montrent que le degré de nocivité d'un cas de test XSS varie en fonction de la vulnérabilité des navigateurs web. Concrètement, un navigateur avec un faible degré d'exposition aux menaces peut tomber face à un cas de test XSS faiblement nocif. La démarche proposée peut être précieuse pour sélectionner

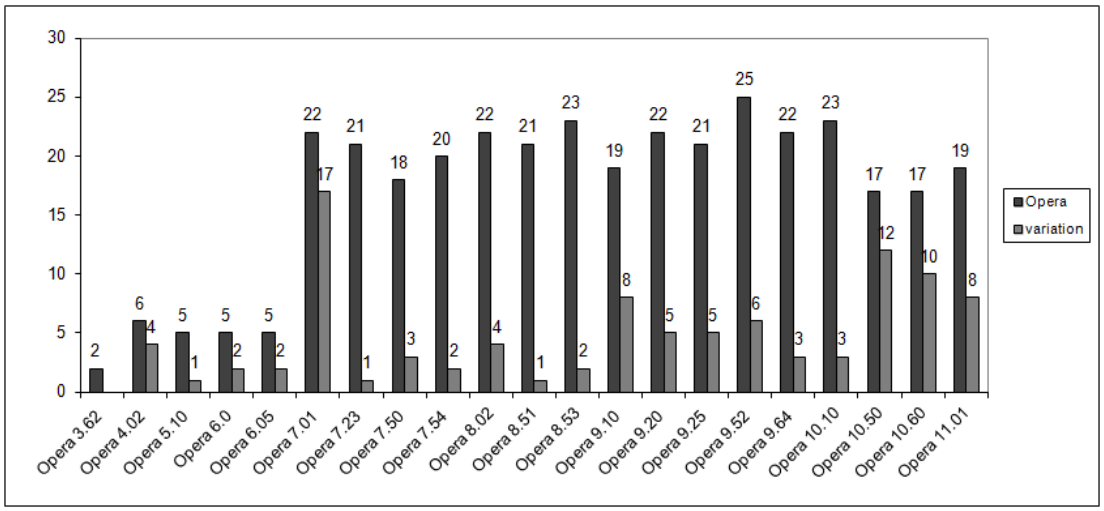


Figure 3. Tests de non regression des versions de Opera. passing vectors / *Delta(n, n-1)*

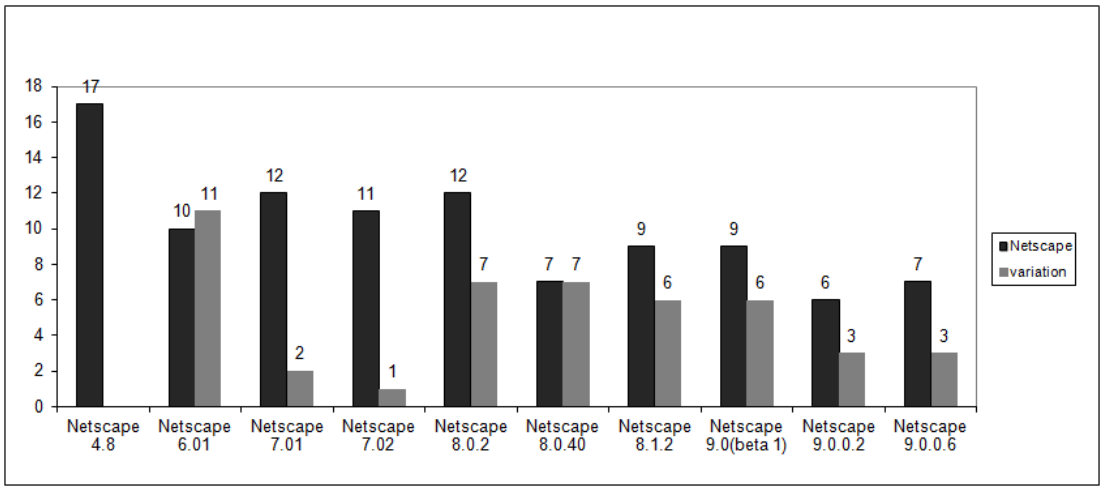


Figure 4. Tests de non regression des versions de Netscape. passing vectors / *Delta(n, n-1)*

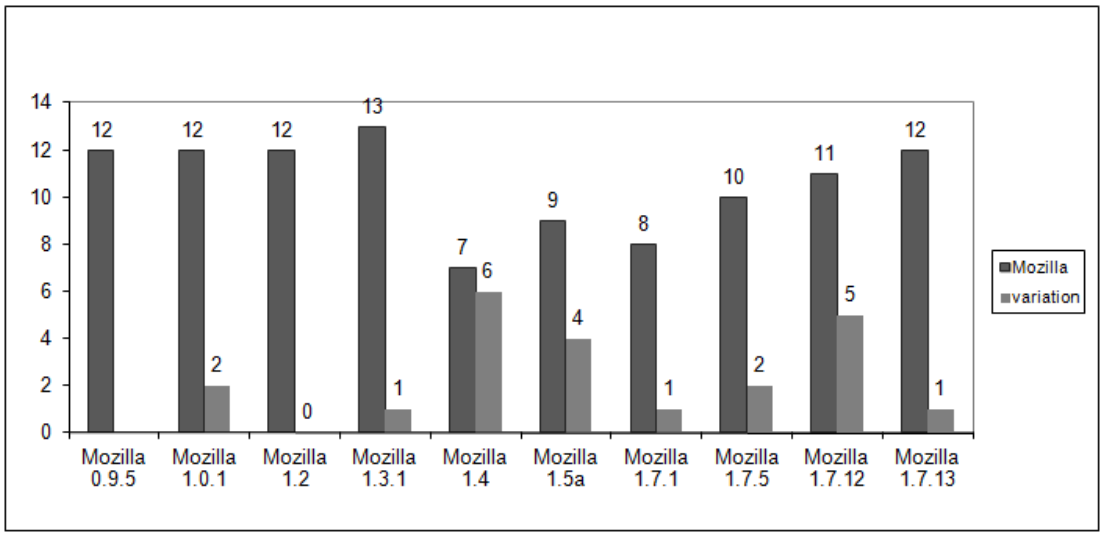


Figure 5. Tests de non regression des versions de Mozilla. passing vectors / *Delta(n, n-1)*

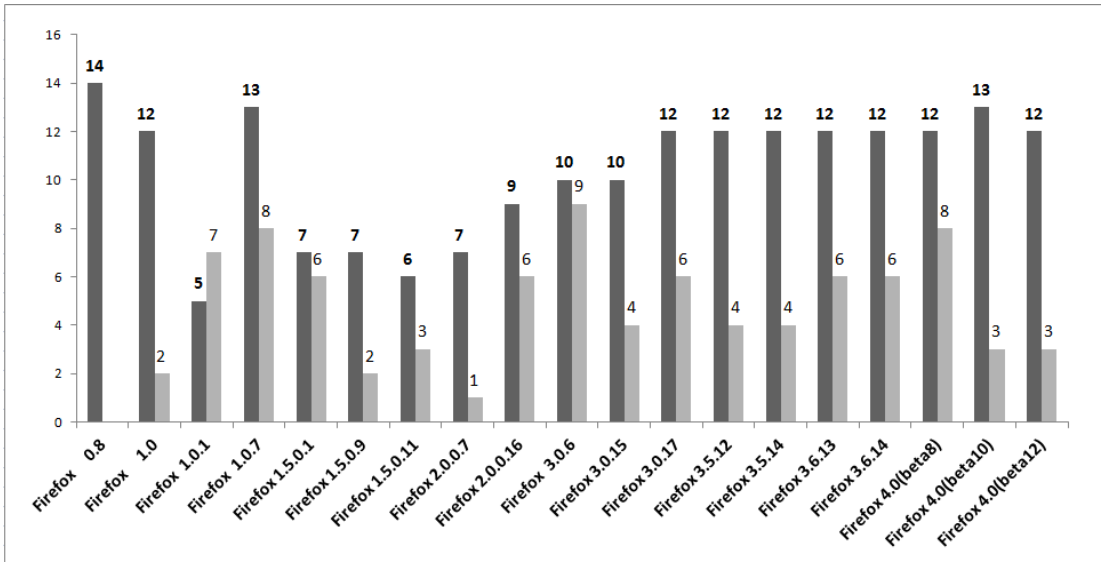


Figure 6. Tests de non regression des versions de Firefox. passing vectors / *Delta(n, n-1)*

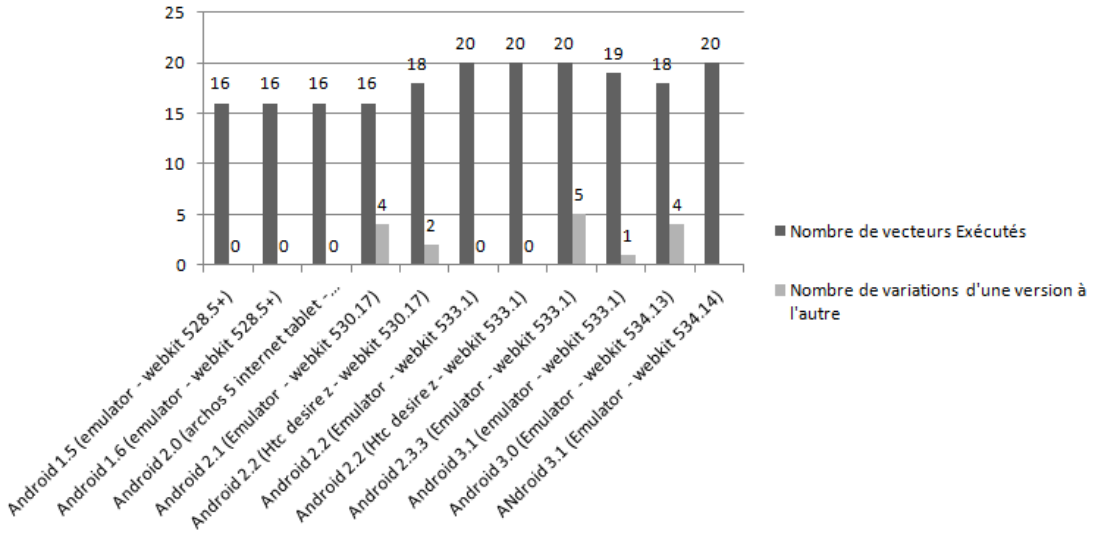


Figure 7. Tests de non regression des versions des navigateurs Android. passing vectors / *Delta(n, n-1)*

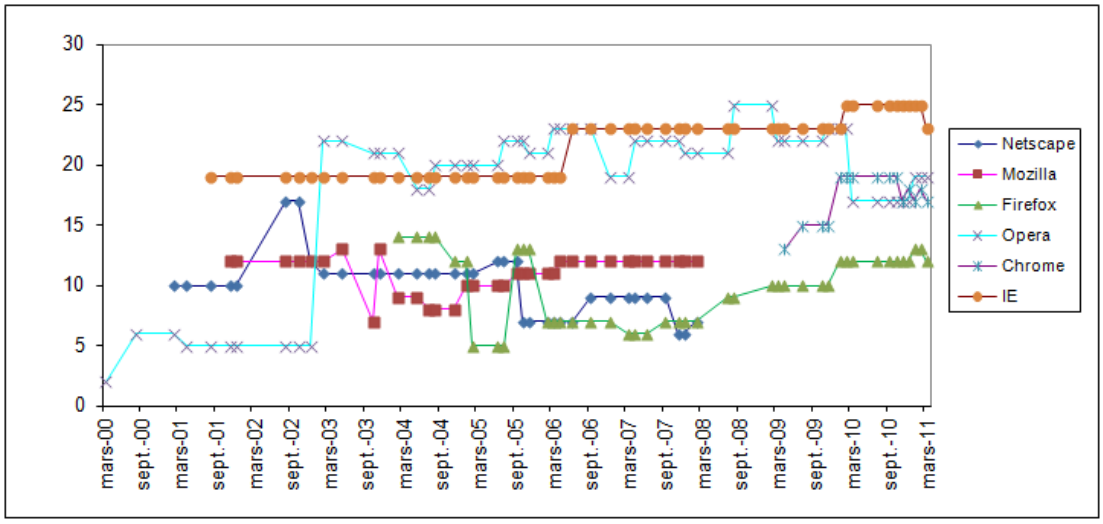


Figure 8. Evolution de la surface d'exposition des navigateurs aux XSS sur 10 ans

les tests XSS pertinents lors du déploiement d'une application web et de ses composants sécurité associés. Un axe fort de la poursuite de nos travaux s'intéressera à la construction méthodique d'une base enrichie de vecteurs XSS en utilisant des techniques telles que le *model based testing*, le *fuzzing* ou le *reverse-engineering* sur les analyseurs HTML des navigateurs web.

Remerciements

Les auteurs tiennent à remercier Ingrid Kemgoum, Télécom Bretagne, campus de Rennes, pour sa contribution pertinente aux tests des navigateurs, François Sorin pour ses commentaires avisés, Guillaume Couteau pour ses corrections. Les travaux de KEREVAL et Télécom Bretagne entrent dans le cadre du projet DALI (Design and Assessment of application Level Intrusion detection systems) financé par l'agence nationale de la recherche (ANR ARPEGE 2008)

Références

1. Mod security. `http://www.modsecurity.org/`.
2. P. Wurzinger, C. Platzer, C. Ludl, E. Kirda, and C. Kruegel. Swap : Mitigating xss attacks using a reverse proxy. In *Proceedings of the 2009 ICSE Workshop on Software Engineering for Secure Systems*, pages 33–39. IEEE Computer Society, 2009.
3. E. Nava and D. Lindsay. Abusing internet explorer 8's xss filters. *BlackHat Europe*, 2010.
4. J. Offutt, Y. Wu, X. Du, and H. Huang. Bypass testing of web applications. In *Proc. of ISSRE*, volume 4.
5. E. Martin and T. Xie. Automated test generation for access control policies via change-impact analysis. In *Proceedings of the Third International Workshop on Software Engineering for Secure Systems*, page 5. IEEE Computer Society, 2007.
6. Y. Le Traon, T. Mouelhi, and B. Baudry. Testing security policies : Going beyond functional testing. 2007.
7. T. Mouelhi, F. Fleurey, B. Baudry, and Y. Le Traon. A model-based framework for security policy specification, deployment and testing. *Model Driven Engineering Languages and Systems*, pages 537–552, 2008.
8. H. Liu and H.B. Kuan Tan. Testing input validation in web applications through automated model recovery. *Journal of Systems and Software*, 81(2) :222–233, 2008.
9. A. Tappenden, P. Beatty, and J. Miller. Agile security testing of web-based systems via httpunit. 2005.
10. J. Offutt, Q. Wang, and J. Ordille. An industrial case study of bypass testing on web applications. In *2008 International Conference on Software Testing, Verification, and Validation*, pages 465–474. IEEE, 2008.
11. Z. Su and G. Wassermann. The essence of command injection attacks in web applications. In *ACM SIGPLAN Notices*, volume 41, pages 372–382. ACM, 2006.
12. Y.W. Huang, S.K. Huang, T.P. Lin, and C.H. Tsai. Web application security assessment by fault injection and behavior monitoring. In *Proceedings of the 12th international conference on World Wide Web*, pages 148–159. ACM, 2003.
13. H. Shahriar and M. Zulkernine. Mutec : Mutation-based testing of cross site scripting. 2009.

14. F. Sun, L. Xu, and Z. Su. Client-side detection of xss worms by monitoring payload propagation. *Computer Security–ESORICS 2009*, pages 539–554, 2009.
15. J. Bau, E. Bursztein, D. Gupta, and J. Mitchell. State of the art : Automated black-box web application vulnerability testing. In *2010 IEEE Symposium on Security and Privacy*, pages 332–345. IEEE, 2010.
16. G. Wassermann and Z. Su. Static detection of cross-site scripting vulnerabilities. In *Software Engineering, 2008. ICSE'08. ACM/IEEE 30th International Conference on*, pages 171–180. IEEE, 2008.
17. J. Shanmugam and M. Ponnavaikko. Xss application worms : New internet infestation and optimized protective measures. In *Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2007. SNPD 2007. Eighth ACIS International Conference on*, volume 3, pages 1164–1169. IEEE, 2007.
18. M.R. Faghani and H. Saidi. Social networks' xss worms. In *2009 International Conference on Computational Science and Engineering*, pages 1137–1141. IEEE, 2009.
19. D.M. Jayamsakthi Shanmugam1. Cross site scripting-latest developments and solutions : A survey. *Int. J. Open Problems Compt. Math*, 1(2), 2008.
20. détails de l'attaque aurora. `http://fr.wikipedia.org/wiki/Op%C3%A9ration_Aurora`.
21. Xss cheat sheet. `http://ha.ckers.org/xss.html`.
22. html5 security cheat sheet. `http://html5sec.org/`.
23. Sla.ckers web application security forum. `http://sla.ckers.org/forum/`.
24. Xss test driver demo. `http://xss.technomancie.net/`.
25. Rhinounit, framework de tests unitaires javascript. `http://code.google.com/p/rhinounit`.
26. W3c automatic page refresh. `http://www.w3.org/TR/WCAG10-CORE-TECHS/#auto-page-refresh`.
27. Utf7 xss cheat sheet. `http://openmya.hacker.jp/hasegawa/security/utf7cs.html`.

TazSecure, système d'authentification forte basé sur la biométrie et un composant sécurisé

Ismail SABRY

TazTag et natural security

Le développement des identités numériques et des systèmes de contrôle d'accès entraine une prolifération des mots de passe, des badges et des systèmes de contrôle. Diverses solutions techniques existent, mais leur utilisations doivent être simples, rapides et sécurisées et conformes avec les recommandations imposées par la CNIL.

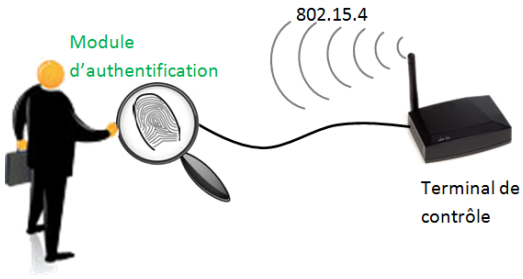
Une enquête réalisée par le Club des directeurs de la sécurité des entreprises (CDSE) sur les enjeux des directeurs de sécurité d'entreprise pour l'année 2010, révèle que plus de 18% des directeurs de sécurité considèrent la sécurisation des bâtiments et des données, avec des technologies telles que la biométrie, comme une préoccupation importante.

Face à de telles contraintes, sur quels critères doit se baser le cahier des charges d'une entreprise ?

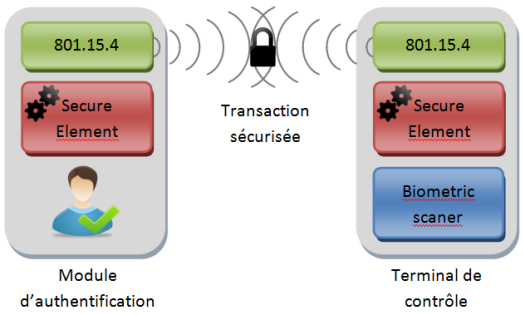
Aujourd'hui, le badge est une tendance forte dans le domaine de la sécurité ; moins contraignant. Il autorise à la fois l'accès au site, au restaurant d'entreprise, au parking ou encore pour se connecter à son ordinateur. Mais lorsque les responsables de la sécurité des entreprises choisissent de s'équiper en systèmes de sécurité plus performants, ils s'orientent davantage vers les systèmes biométriques, considérés aujourd'hui comme le moyen le plus fiable en matière de gestion des biens, des personnes et des informations, mais ils sont souvent confrontés aux contraintes imposées par la CNIL.

1 La solution TazSecure

La solution TazSecure est fondée sur une authentification biométrique sans trace, basée sur un support individuel sécurisé offrant ainsi à l'utilisateur la maîtrise de sa donnée biométrique qui ne peut pas être utilisée pour l'identifier à son insu. Les identifiants et mots de passe ne sont pas nécessaires, l'authentification se fait via une connexion sans fil (802.15.4) rapide et sécurisée, quel que soit le lieu et le type d'accès et cela sans avoir à présenter son moyen d'identification ni à saisir un code secret.



TazSecure permet de garantir l'identité d'un individu ou d'authentifier qu'il est le porteur légitime d'un module d'identification, là où les autres technologies ne peuvent garantir l'identité de l'usager ou du porteur d'un badge. Avec TazSecure, l'usager devient sa propre clef infalsifiable. La solution TazSecure pallie au risque d'usurpation d'identité, de copie, d'emprunt ou vol de badge, d'indiscrétion ou de piratage de mot de passe. Avec TazSecure, nul autre que l'usager ne peut accéder à sa place aux locaux ni à aux informations.



Dans cette solution, le gabarit des informations biométrique de la personne concernée est chiffré par l'intermédiaire d'un algorithme cryptographique et exclusivement enregistré dans le SE (Secure Element) sur le module d'identification portable détenu par elle seule et dont le contenu ne peut être lu à son insu. Cette solution apporte une avancée significative en matière de gestion des biens, des personnes et des informations tout en respectant les contraintes imposées par la CNIL.

Le module d'authentification incorpore un SE pour protéger l'accès aux données personnelles et aux applications. Il est basé sur des plateformes logicielles sécurisées certifiées Common Criteria EAL 4+/EAL 5+ et/ou FIPS140-2 et offre un stockage sécurisé des clés, des certificats et des données utilisateurs.

Le choix TazTag a été guidé par le souci de trouver la meilleure combinaison entre le niveau de sécurité (fiabilité) de la solution, son coût et sa facilité d'utilisation. Il a donc été plus judicieux de coupler la biométrie à un second dispositif pour une d'authentification forte. C'est en ce sens qu'un élément de sécurité additionnel de protection des données fondées sur la cryptographie asymétrique,

a été retenu. Ce dispositif repose sur l'exploitation de clés publiques et de clés privées qui sont embarquées dans le module d'identification et dans le terminal de contrôle. Cette méthode permet de conserver le même chiffrement lors d'un déplacement sur l'ordinateur ou sur le réseau de l'organisation. Les utilisateurs n'ont plus besoin de partager des clés secrètes via une voie de transmission sécurisée. Les communications impliquent uniquement l'utilisation de clés publiques et plus aucune clé privé n'est transmise ou partagée. Les mécanismes mis en œuvre par la solution TazSecure présentent les propriétés suivantes :

Authentique : permet au terminal de contrôle d'authentifier un usager (personne ou entité signataire).

Infalsifiable : l'identité biométrique ne peut pas être falsifiée. Quelqu'un ne peut se faire passer pour un autre.

Non réutilisable : l'identité biométrique fait partie du module d'identification et ne peut être déplacée sur un autre module.

Inaltérable : un module est inaltérable. Une fois qu'il est crypté, on ne peut plus le modifier.

Irrévocable : la personne qui a validé une transaction ne peut le nier.

Le TazSecure propose une gamme complète, de services embarqués dans le module, adaptée à chaque contrat et à tous les types de marchés : documents gouvernementaux, santé, transport, accès physique et logique, badges d'entreprise.

Afin de rendre les nouvelles technologies encore plus efficientes et attractives et contribuer à leurs utiles exploitations. TazTag et natural security mettent à disposition leurs compétences dans l'analyse, le développement, la production et l'organisation nécessaires à l'exploitation de ces nouveaux outils et la déclinaison « marché » de leurs utilisations.

Cas d'usage : Paul et la gestion de son identité numérique mobile

Expliquons l'utilisation de TazSecure de façon imagée, Paul utilise sa voiture chaque matin pour se rendre sur son lieu de travail dans une société qui fabrique des terminaux mobiles pour les agents diplomatiques. Mais Paul ne fait pas confiance aux solutions standards en matière de protection de son véhicule, d'accès à son bureau et d'authentification de ses documents et de ses emails.

Quelle solution globale de sûreté doit permettre à Paul de proteger l'ensemble de ses biens et les données de son entreprise ?

La solution badge (module personnel) unique sans contact associée à la biométrie pour l'accès à certaines zones et/ou documents plus sécurisés, est la tendance

forte dans le domaine de la sécurité. L'insertion de différentes données dans le badge de Paul permet à la fois l'accès à son véhicule, à son domicile, aux locaux de son entreprise et aux données sécurisés. Les modes d'authentification peuvent ainsi être modulés selon la politique de sécurité souhaitée. Pour se rendre à son bureau le matin, Paul va d'abord prendre son véhicule, dont sa femme et lui seuls possèdent la clé. Paul va accéder à sa voiture, qui s'ouvre automatiquement grâce à une double authentification suite à la détection du module de Paul au voisinage du véhicule.

Sur le module de Paul et celui de sa femme sont inscrites certaines informations relatives au propriétaire (identifiant de Paul, identifiant du module, ...), et une valeur de signature calculée en utilisant la clé secrète. La clé secrète (S) de Paul est stockée d'une manière sure dans le SE du module.

$$Signature_{Paul} = S(info)$$

Lorsque le module de Paul est détecté par le terminal de sa voiture, celui-ci lit les informations portées par le module, et la valeur de signature. Il calcule alors une nouvelle signature calculée en utilisant la clé publique (P).

$$Signature_{Voiture} = S(info)$$

Puis le terminal de contrôle de la voiture compare les deux signatures : pour que le module soit valide, il faut que :

$$Signature_{Paul} = Signature_{Voiture}$$

Ces fonctions à clé secrète et à clé publique sont basées sur le RSA.

Une fois l'authentification faite Paul accède à sa voiture et se dirige vers son bureau. La première mesure de sécurité dans l'entreprise de Paul est l'authentification à l'entrée du site. L'identité de l'utilisateur est contrôlée par des agents.

Sur le module de Paul sont inscrites certaines informations relatives à lui-même et à son entreprise (identifiant de Paul, identifiant du module, nom, photo, date de validité ...). Lorsque le module de Paul est détecté par le terminal de contrôle de l'entrée du site, celui-ci lit les informations portées par le module, et envoi au module de Paul une valeur aléatoire (X). Le module calcule alors une nouvelle signature calculée en utilisant la valeur aléatoire et une clé secrète (K) fournie par son employeur et enregistrée dans le SE de son module

$$Signature_{Paul} = f(X, K)$$

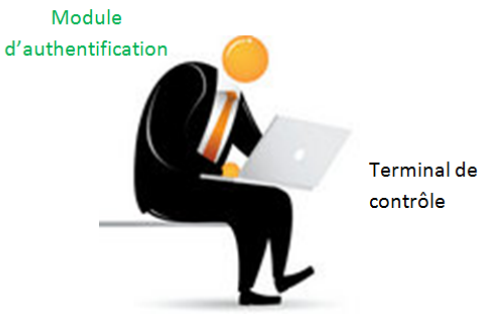
f est la fonction de chiffrement du triple DES La valeur de la signature est retransmise au terminal de contrôle de l'entrée du site et donne ou non l'autorisation. Le terminal de contrôle de l'entrée du site interroge un centre de contrôle central qui à connaissance de toutes les cartes.

Une fois Paul arrivé à son emplacement de stationnement, le véhicule est identifié, reconnait le site de l'entreprise et envoie au gestionnaire du parc les données relatives à l'entretien du véhicule.

Paul travaille dans le pôle Recherche et Développement où l'accès est restreint aux personnes autorisées uniquement.

Lorsque l'accès est demandé à cette zone, la personne présente sa main au lecteur biométrique, qui ensuite vérifie que les mesures biométriques de cette personne correspondent à celles stockées sur son module.

Lorsque le module de Paul est détecté par le terminal de contrôle de l'entrée de la zone sécurisée, celui-ci lit les informations portées par le module, capture le gabarit biométrique de Paul qu'il envoie au module de Paul. Après le succès de la comparaison de ce gabarit avec gabarit d'empreinte de référence stoké dans le SE du module, le terminal de contrôle et le module procèdent à une authentification identique à celle de l'entrée du site. Une fois arrivé dans son bureau, Paul s'installe devant son poste de travail, la session d'authentification s'exécute automatiquement.



Le terminal utilise la RF (802.15.4) pour localiser Paul dans son bureau. Si la distance entre Paul et son poste de travail est inférieure à quelque mètres, la session reste ouverte jusqu'‘a ce que Paul décide de la fermer ou qu'il ne soit plus détecté par le terminal de contrôle relié à son ordinateur. Durant la période de connexion, Paul peut communiquer et envoyer des messages à ses collègues en utilisant sa signature numérique. Grace à la signature numérique, les collègues de Paul sont sûrs qu'il en est l'auteur et que le contenu du message n'a pas été altéré entre l'instant où Paul l'a signé et le moment où le destinataire le consulte Lorsque Paul souhaite signer un message M, il commence par générer une nouvelle signature de son message en utilisant une fonction (h) de hachage et sa clé privé S.

$$Signature_{Paul} = S(h(M))$$

Le résultat de cette opération, permet de s'assurer de l'intégrité du document et que c'est bien Paul qui a rédigé le message M. C'est ce résultat qui constitue la signature du message, aux côtés duquel il est transmis. Le destinataire, qui connait la clé publique de Paul, reçoit le message M ainsi que la signature associée. Afin de vérifier son authenticité Le nouveau résultat de hachage du message est généré au moyen de la même fonction (h). Parallèlement, la signature $(Signature_{Paul})$ est déchiffrée au moyen de la clé publique. En cas d'égalité, le message M est authentifié car seul Paul, avec sa clé privée, est capable de générer un résultat compatible avec sa clé publique et l'intégrité du message. Paul peut utiliser son module pour la multi-signature électronique, aussi appelée signature de groupe, pour répondre aux besoins de contrôle de l'authentification des partenaires et collègues ou de l'origine des informations, de l'intégrité des données transmises (vérifier que les informations n'ont pas été modifiées ou altérées), de la non répudiation de la transaction (s'assurer de la preuve de l'émission ou de la réception des informations), ainsi qu'aux besoins plus ou moins élevés de confidentialité.

La signature de groupe se base sur les mêmes principes développés par le groupe de travail « XML Signature WG » qui est un groupe conjoint du W3C et de l'IETF. De ces travaux sont issus une « Recommandation » du W3C et une « Standard track » de l'IETF, « XML-Signature Syntax and Processing » / RFC3275

Références

IEEE Std 1363-2000, IEEE standard specifications for public-key cryptography, IEEE Computer Society, August 29, 2000.

ETSI TS 101 903. XML Advanced Electronic Signatures (XAdES). February 2002.

[XML/DSig] XML Signature (Syntax and Processing) - `http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/`

RFC 1321. The MD5 message digest algorithm. Internet Request for Comments 1321, Ronald Rivest, April 1992. Available at `http://www.ietf.org/rfc/rfc1321.txt`

RFC2560. Myers, M., Ankney, R., Malpani, A., Galperin, S., Adams, C., X.509 Internet Public Key Infrastructure : Online Certificate Status Protocol - OCSP. June 1999.

RFC3275.Eastlake 3rd D., Reagle J., Solo D., (Extensible Markup Language) XML Signature Syntax and Processing. (XML-DSIG) March 2002.

RFC 4359. The Use of RSA/SHA-1 Signatures within Encapsulating Security Payload (ESP) and Authentication Header (AH), B.Weis, January 2006, Available at `http://www.ietf.org/rfc/rfc4359.txt`.

Troisième partie

30 novembre 2011

Comment allier objectifs d'usage et objectifs de sécurité ?

Pierre-Yves Gouardin, Sahra Zaim, Charles Capron, Claire Premont

Orange Consulting. pierreyves.gouardin(@)orange.com, sahra.zaim(@)orange.com, charles.capron(@)orange.com, claire.premont(@)orange.com

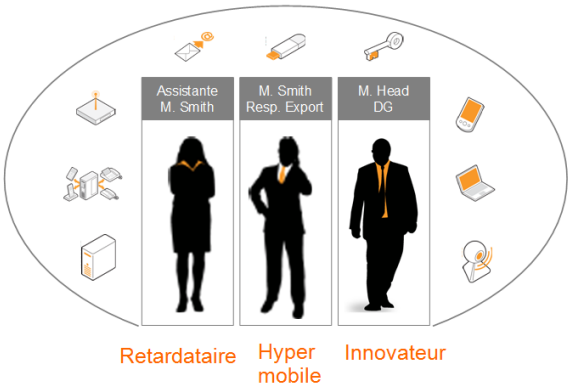
Résumé Nous allons, dans cet article, expliciter, au travers de l'étude de cas d'un industriel français de l'armement, les nouveaux usages et les risques qui découlent des situations de mobilité. Notre scénario met en scène trois protagonistes : M. Head, Directeur Général, M. Smith Responsable Export et son assistante. Dans le cadre d'un appel d'offre, ces trois membres de la société « TargetSA » vont faire l'objet de tentatives d'attaques de la part d'un concurrent, la société « WarSA ». Leurs usages en mobilité donneront l'opportunité à « WarSA » de récupérer des informations concernant cet appel d'offre, et ainsi maximiser leurs chances de le remporter. Ce scénario sera également l'occasion de présenter notre approche pour adresser cette problématique, ou comment allier objectifs d'usage et objectifs de sécurité ?

1 Introduction

Durant ces dernières années, une nette évolution des comportements de mobilité qu'il soit contraint ou non contraint, professionnel ou personnel a été observée. De nouveaux modèles de comportement en mobilité sont apparus, et ces comportements quotidiens devraient être davantage modifiés par l'intégration des technologies de l'information et de la communication (géo-socialisation, télé-présence, environnement, instantanéité, etc.).

La sécurité en situation de mobilité est aujourd'hui un challenge stratégique pour les entreprises. La banalisation des PC portables, des Smartphones ainsi que les structures multi-sites des entreprises ont profondément modifié les comportements et habitudes de travail des employés.

Une entreprise française de technologie de pointe, TargetSA, ayant assisté à l'émergence de cet environnement nomade, a identifié les différents usages de ses employés. Trois profils d'employés se distinguent :



Ce document se propose de mettre en perspective les difficultés pour les entreprises à travailler dans un contexte de plus en plus concurrentiel soumis à des règles strictes en matière de confidentialité et de sécurité, avec un nomadisme accru des cadres et personnes stratégiques de l'entreprise. La détention de l'information est clé car elle est à l'origine du rapport de force asymétrique entre l'entreprise et ses concurrents. C'est dans ce contexte que nous nous proposons de répondre à la problématique suivante :

Comment allier objectifs d'usage et objectifs de sécurité ?

Nous traitons de cette question à travers un scénario mettant en œuvre une entreprise française d'armement, TaretSA, répondant un appel d'offre d'envergure mondiale et étant confrontée à cet environnement nomade. L'entreprise TargetSA fera l'objet d'attaque de la part d'un concurrent, WarSA. Les aspects sécurité et usages seront alors confrontés pour en déduire une analyse et profile de risque.

2 L'émergence d'un environnement nomade et connecté bouleverse les comportements et habitudes de travail des salariés

2.1 Quels enjeux pour le salarié nomade ?

De plus en plus de personnes se déplacent chaque jour et ce chiffre ne fera qu'augmenter dans les années à venir. La durée des trajets s'accroît également. A titre d'exemple, dans la grande couronne d'Île-de-France, on compte environ 500 000 travailleurs qui se déplacent à plus de 2 heures [1] (aller et retour) de leur lieu de travail par la voiture ou les transports en commun.

L'accroissement de l'utilisation des smartphones, tablettes et mini PC permet à ses utilisateurs de faire l'expérience d'une mobilité connectée. La couverture

1. INSEE

du réseau 3G+ et le déploiement de la 4G en France permettent d'apporter plus de confort aux utilisateurs. En 2011, le haut débit mobile est très largement développé en France avec un réseau 3G+ disponible pour 96% de population [2] et une couverture WI-FI très dense. En effet, avec pas moins de 9 000 points d'accès Wi-Fi et quelques dizaines de milliers d'utilisateurs constatés, la France est située au 3ème rang mondial en termes de couverture nationale Wi-Fi du territoire, derrière deux pays anglo-saxons, les États-Unis et le Royaume-Uni. La connexion à un réseau public ou privé en mobilité est ainsi fortement faciliter par des infrastructures performantes et fiables.

Le développement des réseaux de transport intelligents est une des priorités des futures villes intelligentes que l'on trouvera en France. Les projets de « smart cities » vont continuer à clairement influencer l'offre de trasnport pour les personnes en mobilité, qui seront poussées à davantage utiliser les nouvelles technologies de l'information et de la communication. Les aspects environnementaux sont également pris en compte et participent aux changements de comportements des voyageurs (auto-partage, covoiturage, véhicule hybride ou électrique, etc.)

Cette évolution, met en avant les nouvelles exigences des voyageurs qui insistent sur le fait d'être connecté tout le temps, n'importe où et avec n'importe quel terminal. Dans un tel contexte, la question de la sécurité des usages se pose. Assurer la sécurité des données, des échanges, des terminaux en mobilité est aujourd'hui un enjeu majeur pour les particuliers et les entreprises.

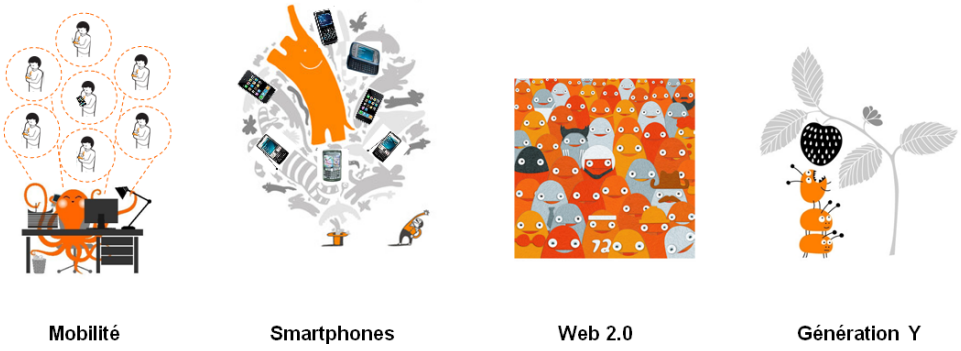
En outre de la banalisation des PC portables, tablettes et smartphones ainsi que de l'évolution des usages en mobilité, les entreprises sont confrontées à des évolutions structurelles telles que la multiplicité des structures multi-sites, l'accroissement du télétravail, le développement des télé-centres, etc. Aujourd'hui, 1 entreprise sur 2 en France possède des salariés itinérants alors que seulement 7% des entreprises ont mis sur pied un réseau privé d'entreprise [3] (VPN). Le foisonnement rapide des mobilités autour de l'entreprise, renforcé par la mondialisation des échanges, rend plus difficile la mise en place de politiques sécuritaires répondant à la forte mobilité de leur employés. Toute la difficulté demeure dans le bon dosage entre liberté des usages et sécurité des données/accès, et cela est d'autant plus vrai dans les grandes entreprises où les profiles d'utilisation sont très variés.

Malgré la crise, la mobilité et le nomadisme restent une priorité pour les entreprises et les directions informatiques. Ainsi, en 2011 43% des entreprises prévoient d'investir dans des solutions de mobilité au cours des prochains mois. Le résultat monte jusqu'à 46% pour les entreprises de plus de 500 salariés [4]. Cette problématique est donc prise au sérieux par les entreprises, faut-il encore savoir quel outil adopter et dans quelle mesure ouvrir ses accès !

2. Site Orange.fr
3. Site CoAxion, article « Internet et le nomadisme »
4. baromètre IDC pour le compte de Dell, sondage auprès de 200 entreprises

2.2 Les enjeux de la mobilité connectée pour les salariés et les entreprises

La révolution des Smartphones, associée au besoin croissant d'être toujours connecté, où que l'on se trouve, ont bouleversé les habitudes et les comportements des salariés. Le web 2.0 et l'utilisation massive de sites communautaires et de forums avant, pendant et après la journée de travail ont également influencé le comportement du salarié, qui se veut plus averti et davantage demandeur de technologies de communication modernes et fiables. Ce phénomène s'est amplifié avec l'arrivée de la génération Y au travail qui se soucie toujours plus de l'e-réputation de leur entreprise et qui mélangent la sphère privée et professionnelle.



L'arrivée des nouvelles technologies a radicalement modifié les comportements des personnes en mobilité. Le salarié est désormais hyper-connecté et a tendance à mélanger sphère privée et professionnelle. Les commerciaux, cadres dirigeants, secrétaire de direction sont autant de populations exposées au nomadisme dans l'entreprise. Souvent équipés d'un PC portable et/ou d'un smartphone, ils se confrontent quasi quotidiennement aux problématiques de sécurité en mobilité. Les équipements choisis par leurs entreprises sont souvent rendus obsolètes par les derniers terminaux à la mode (iPhone 4, iPad, Blackerry 9900, etc.) qui finissent toujours par envahir le monde du travail. Le succès des smartphones ne risque pas de s'affaiblir puisque leur nombre est estimé à 11,8 millions en France en 2011 [5]. Il n'est pas rare également que le salarié nomade utilise son mobile personnel, possédant un forfait data plus adapté et étant plus en accord avec ses usages, à la place de son mobile professionnel.

Les technologies de l'information et de la communication n'ont pas seulement fait émerger de nouveaux usages auprès des salariés, mais elles ont également permis aux entreprises de proposer de nouveaux outils à leurs salariés en termes de communication en mobilité. Les nouvelles solutions open source ont aussi permis

5. Étude GFK, bilan 2010 et prévisions 2011 des biens techniques

de baisser le coût des solutions proposées. Ainsi, sécuriser la mobilité de ses employés n'est plus réservé seulement aux grandes organisations. Les solutions bâties sur les logiciels open sources fonctionnant sur IP permettent de créer des VPN (réseau virtuel sécurisé privé) qui centralisent pratiquement toutes les activités informatives, gestion, voix sur IP sécurisée etc. à moindre coût.

Le nombre d'employés nomades à de ce fait fortement augmenté ces dernières années. La population mondiale de travailleurs nomades devrait atteindre le million d'ici fin 2011 [6]. Une conjonction de facteurs fait que ce chiffre continuera de grimper : la mondialisation économique et financière a eu un impact non négligeable sur les modes d'organisation des entreprises (délocalisation des sites à l'étranger, création de locaux de passage, etc.), la maturité de certaines technologies (smartphones, tablette, mini PC, etc.), la volonté politique d'investir dans les réseaux de télécommunications (déploiement des réseaux 3G+, 4G et Wi-Fi), le développement d'offres de transports fiables (smart grid et construction de transports intelligents connectés). En outre, malgré la crise de 2008, les entreprises continuent d'investir dans des solutions de protection destinées à leurs employés nomades (antivol, chiffrement, authentification, sauvegarde externalisée, etc.).

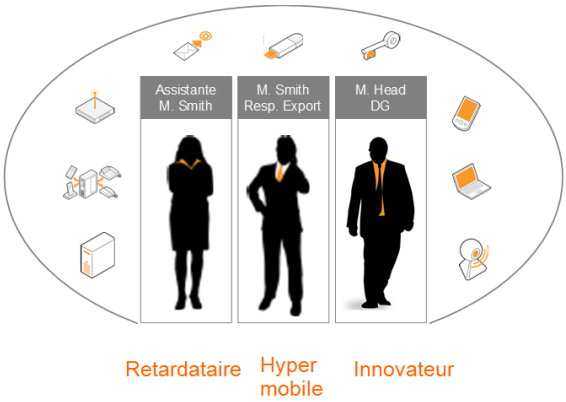
L'arrivée au travail de la Génération Y devrait faire évoluer les schémas traditionnels des entreprises en matière de nomadisme. De nouvelles solutions émergent proposant aux employés d'acheter leur propre terminal et de s'en servir dans le cadre professionnel (stratégie de « Bring Your Own Device »). Le BYOD peut être aussi une opportunité économique pour l'entreprise. Lors de sa keynote aux dernières Assises de la Sécurité, Tom Gillis (fondateur d'Iron Port rachetée par Cisco, et vice-président de Security Technology Business Unit) a indiqué que chez Cisco, la politique depuis 3 ans était d'accepter les terminaux personnels (smartphones, tablettes). « Nous avons baissé de 25% nos coûts informatiques et 200% de satisfaction pour les utilisateurs ».

La porosité entre vie privée et vie professionnelle, la méconnaissance des pratiques de sécurité en mobilité, l'addiction aux smartphones, la vie communautaire et l'hyper-connectivité font de la génération Y, et plus globalement des utilisateurs une cible idéale pour les potentiels hackers voulant mettre à mal une entreprise.

3 Analyse des populations d'utilisateurs et des usages

Afin de mettre en avant les difficultés et les risques qu'encourt une entreprise maîtrisant mal les usages de ses salariés, nous allons décrire les profils choisis dans notre scénario en insistant sur la description de leurs équipements par rapport à leur profil d'utilisation des technologies.

6. Site Orange.fr



Description des populations cibles dans le choix de notre scénario Nous avons identifié trois populations d'utilisateurs pour ce cas d'étude aux usages de communication et de sécurité différents :

M. Head

Le premier personnage intervenant dans notre scenario est M. Head, directeur général de TargetSA. Son profile est celui d'un cadre dirigeant peu mobile à forte responsabilités.

M. Head, en tant que directeur général de TargetSA, a un emploi du temps très chargé. Il court de réunion en réunion et ne quitte jamais son smartphone. Malgré ses hautes fonctions, M. Head voyage peu comparativement à son directeur export, M. Smith. Son rôle est d'assurer la pérennité de l'entreprise et de la maintenir toujours à la pointe dans un marché fortement concurrentiel qui est celui de l'armement. Pour arriver à ses fins, il répond régulièrement à plusieurs appels d'offres notamment en Amérique et en Asie. Son seul souci étant d'être meilleur que ses concurrents qui sont internationaux et qui se sont récemment développés en Asie.

M. Head est un amoureux des nouvelles technologies et aime tester tous les derniers gadgets numériques qui sortent sur le marché. Il a de ce fait choisi comme mobile personnel un iPhone 4, beaucoup plus performant selon lui que le mobile distribué par sa société à tous les collaborateurs, et qui est de type Windows Phone. Ce dernier fait l'objet d'un renforcement de sa sécurité (chiffrement local, gestion de flotte centralisée avec notamment la possibilité de réaliser un « remote wipe ») et est donc beaucoup plus sûr que son iPhone personnel. Cependant, malgré le fait que M. Head soit souvent sensibilisé aux risques numériques, il continue d'utiliser son mobile personnel dans le cadre de ses fonctions. Il aimerait d'ailleurs avoir un iPad et iPhone professionnels qui ne sont malheureusement pas au catalogue de service de son entreprise, TargetSA.

M. Smith

Le deuxième personnage intervenant dans notre scénario est M. Smith, directeur des exportations et membre du comité de direction de TargetSA. Son profile est celui d'un cadre dirigeant nomade fortement sensibilisé aux problématiques de sécurité.

M. Smith en tant que responsable des exportations est très souvent amené à voyager et notamment dans des pays dits « sensibles ». Il se déplace souvent physiquement pour répondre à des appels d'offres, rencontrer certains grands comptes et tester la maturité du marché sur de nouveaux produits de TargetSA. Étant une personne voyageant énormément, M. Smith est très au courant des bonnes pratiques à suivre en situation de mobilité. Il a d'ailleurs un équipement spécifique avec un smartphone type Windows Phone et un laptop qui font l'objet d'un renforcement de sécurité supérieur lors de ses déplacements.

Sa fonction l'oblige donc à redoubler de vigilance et c'est pour cela qu'il suit régulièrement des formations en matière de sécurité. Il utilise rarement voire jamais son mobile personnel dans le cadre de son travail et est un salarié averti en termes de confidentialité des données. Il fait donc un usage réfléchi de ses outils et tout particulièrement lorsqu'il travaille sur un sujet critique dans un pays sensible.

Il accorde toute confiance à son assistante, Mme Johannsen, avec qui il travaille depuis plus de 10 ans.

Mme Johannsen

Le troisième personnage de notre scénario est l'assistante de M. Smith, Mme Johannsen. Son profile est celui d'une assistante sédentaire.

L'assistante de M. Smith est une personne loyale qui au cours de ses années de métier a appris à maitriser les technologies. Elle s'est forgée une réputation solide et est une personne de confiance aux yeux de M. Smith et de l'ensemble des collaborateurs de TargetSA. Elle maîtrise les technologies simples de type smartphone et laptop mais ne s'aventure pas plus dans les aspects SI et sécurité. Elle a des usages basiques.

Sa fonction d'assistante l'oblige souvent à être joignable 24/24h et 7/7j ce qui parfois l'amène à travailler de chez elle et même en vacances. Les outils qu'elle utilise sont de ce fait mal adaptés à ses usages car elle ne dispose pas d'un smartphone professionnel mais d'un simple mobile. Elle n'a pas de laptop professionnel non plus, ce qui peut-être gênant surtout quand elle est forcée de travailler à distance.

Bien qu'elle soit très loyale, elle peut commettre des erreurs en termes de sécurité sans réellement mesurer les risques qu'elle peut prendre. Elle n'a pas été sensibilisée aux enjeux de sécurité de l'entreprise bien qu'elle soit au courant du caractère confidentiel des données qu'elle traite.

3.1 Description détaillée des équipements et mesures de sécurité de l'entreprise TargetSA

Nous avons identifié trois populations d'utilisateurs pour ce cas d'étude aux usages de communication et de sécurité différents : M. Head, M. Smith et Mme Johannsen. Désormais, nous allons présenter l'équipement de chacun de ces collaborateurs.

	Équipement	Équipement en cas de déplacement dans un pays	Équipement personnel
M. Head	• Windows Phone « classique » • Laptop « classique »	-	• Iphone 4 (IOS 4.0)
M. Smith	• Windows Phone « classique » • Laptop « classique »	• Windows Phone « sécurité + » • Laptop « sécurité + »	• Iphone 4 (IOS 4.3.5)
Mme Johannsen	• Téléphone cellulaire sans 3G	-	• Iphone 4 (IOS 4.0)

L'entreprise TargetSA a mis en place certains dispositifs de sécurité avec l'aide de sa direction des systèmes d'information. Ceci place l'entreprise TargetSA dans la moyenne haute en matière de sécurité de ses SI. Les différentes mesures mises en place sont les suivantes :

	Profil	Mesures de sécurité
Windows Phone	« classique »	• Gestion de flotte centralisée • Possibilité de « Remote Wipe » • Proxy Web • Chiffrement des données locales • Security Patch Management
	« sécurité + »	• Mesure « classique » • Authentification forte • Chiffrement des communications (Data et voix) • Pas de données rémanentes • Chiffrement « physique » des données locales • Security Patch Management spécifique (Check systématique avant toute mise à disposition, …)
Laptop	« classique »	• Authentification forte • Accès VPN au système d'information • Gestion de flotte centralisée • Security Patch Management • Proxy Web
	« sécurité + »	• Pas de données rémanentes • Chiffrement « physique » des données locales • Accès VPN dédié (zone cloisonnée)

4 Scénarios

4.1 La cible : un industriel français de l'armement

Nous avons choisi pour notre scénario une entreprise française, TargetSA, de technologie de pointe qui exporte la majeure partie de sa production. Cette entreprise emploie plusieurs centaines d'ingénieurs et docteurs en recherche et développement.

Elle est sur un secteur de marché très concurrentiel. Ces principaux concurrents sont situés en Amérique du Nord, en Israël, en Russie et depuis peu en Inde et en Chine.

L'entreprise est présente sur tous les continents avec principalement des relais commerciaux. L'entreprise compte plusieurs milliers d'employés principalement basés en France, parfois sur des bassins d'emplois à fort taux de chômage.

De part la sensibilité de sa technologie, l'entreprise est fortement sensibilisée aux risques que représentent l'espionnage industriel. Par conséquent, elle est suivie avec beaucoup d'attention par les services de l'État français.

4.2 Le contexte : phase de Lobbying suite à un appel d'offre aux USA

Dans le cadre d'un appel d'offre, M. Smith est amené à se déplacer aux USA pour défendre les intérêts de sa société auprès de l'organisation ClientSA.

Le concurrent le plus sérieux est également américain, il s'agit de la société WarSA. Depuis la mise en place du « Patriot Act », l'état américain peut légalement demander l'accès aux données numériques des personnes entrant sur le territoire national. Le risque d'espionnage est donc extrêmement élevé, et c'est pourquoi M. Smith utilisera des moyens de communication à sécurité renforcée.

4.3 L'objectif : maitriser cette phase de lobbying

L'objectif de WarSA est d'obtenir des informations concernant la stratégie de Lobbying utilisée par TargetSA et ainsi maximiser ses chances de prendre l'avantage concernant cet appel d'offre.

4.4 Développement du scénario d'attaque

M. Smith utilise exclusivement le téléphone renforcé lors de ses déplacements. Les douanes américaines ont en effet l'autorité de fouiller et de retenir sans motif particulier tout dispositif pouvant servir à sauvegarder de l'information électronique [7]. Ils peuvent examiner un dispositif électronique sans que le voyageur soit

7. Border Searches of Electronic Devices `http://www.dhs.gov/xlibrary/assets/ice_border_search_electronic_devices.pdf`

présent ; ils peuvent copier le contenu du dispositif ou retenir le dispositif ; et ils n'ont pas besoin du consentement du voyageur pour effectuer une fouille.

Après le passage en douane de M. Smith, et l'examen minutieux de son Smartphone et de son Laptot, la WarSA se rend à l'évidence, elle ne sera pas en mesure d'obtenir la moindre information de cette manière. Les mécanismes de sécurité mis en œuvre ont permis de garantir qu'aucune donnée rémanente n'est présente sur les équipements de M. Smith. Elle doit donc trouver un autre moyen d'accéder à ces informations.

Dans le cadre de ces négociations, M. Smith échange des documents avec sa secrétaire ou le Directeur général. Certains d'entres eux sont des documents sensibles, et concernent la stratégie de lobbying que M. Smith utilisera lors des négociations.

Après renseignement, la WarSA découvre que M. Head utilise son Iphone pour accéder à sa messagerie professionnelle, contrairement à la politique en vigueur au sein de la TargetSA. La WarSA décide alors de créer une application regroupant différent hobby du DG. Il lance une campagne de spams à l'attention de M. Head pour l'inciter à installer cette application. L'Iphone de M. Head ne fait pas l'objet d'un security patch management et est jailbreakable. La WarSA à donc accès, en live, à l'ensemble des emails de M. Head, dont ces fameux documents relatifs à la stratégie de lobbying.

- M. Smith ne peut se l'expliquer, mais des arguments qu'il pensait pertinents, n'ont pas l'effet escompté auprès de ses interlocuteurs.
- La divulgation d'information par attaque ciblée (exploitation d'une vulnérabilité publique à l'aide d'un malware) ruine ses négociations.

Dans le même temps, M. Smith a transmis à sa secrétaire des documents faisant état de propositions à la limite de la légalité (transmis pour relecture avant envoi pour validation à M. Head), dont l'impact éventuel sur l'image de marque de sa société pourrait être désastreux. Mais, Mme Johannsen est en RTT ce jour là, ne dispose pas d'un Smartphone professionnel (mais simplement d'un téléphone classique), ni d'aucun autre moyen de connexion au système d'information de TargetSA. Elle décide alors d'utiliser son Smartphone personnel pour se connecter au webmail de sa société, et ainsi avoir accès au document en question. Après relecture, elle transfert le document à M. Head, et a le sentiment du travail accompli.

Malheureusement, plusieurs semaines plus tard, lors d'un déplacement en Bretagne, Mme Johannsen perd son Smartphone personnel, qui se retrouve entre les mains d'Émilien, jeune geek Rennais. Émilien ne tarde pas à connecter l'appareil à son ordinateur afin d'assouvir sa curiosité. « Pas de mot de passe pour protéger l'appareil, ça n'est même pas drôle ! ». Sa curiosité va très vite être récompensée lorsqu'il découvre le document décrivant la stratégie de Lobbying de TargetSA. Il ne comprend pas tout, mais le nom de cette société, et la mention « Confidentiel »

lui font penser qu'il est tombé sur quelque chose d'intéressant. Il contacte alors un journaliste, et lui propose le document en question qui sera à l'origine d'un buzz.

– Atteinte à l'image de marque par divulgation d'information (suite à la perte d'un appareil).

4.5 Analyse des risques illustrés dans ce scénario

Concernant les menaces mises en évidence dans notre scénario d'attaque, compte tenu des mécanismes de sécurité mis en œuvre auprès des devices « corporate », une analyse des risques résiduels préalable aurait probablement eu les résultats suivants :

ID	Description	Menace associée	Prob of occ *	Impact	Niveau de risque
RIS.01	Exploitation d'une vulnérabilité publique impactant le device	• Divulgation d'information	Négligeable	Fort	Faible
RIS.02	Perte ou vol du téléphone portable	• Divulgation d'information	Fort	Négligeable	Faible

Mais cette analyse ne prend pas en considération le risque de contournement des mécanismes de sécurité mis en œuvre par l'utilisation de device non « corporate » pour des usages professionnels. Dans ce contexte, les résultats de cette analyse seraient :

ID	Description	Menace associée	Prob of occ *	Impact	Niveau de risque
RIS.01	Exploitation d'une vulnérabilité publique impactant le device	• Divulgation d'information	Moyenne	Fort	Fort
RIS.02	Perte ou vol du téléphone portable	• Divulgation d'information	Fort	Fort	Fort

5 Éléments de réponses aux besoins

5.1 La dépérimétrisation de la sécurité

Ce scénario nous montre qu'il n'est pas possible de garantir la sécurité des usages en mobilité, sans analyser les besoins dans leur ensemble. Les éléments de contexte et de population d'usager doivent être pris comme postulat de cette analyse des besoins, présents et futurs.

5.2 Définition des objectifs d'usages, présents et futurs

Ou comment prendre en considération l'ensemble des composantes du contexte particulier des usages en mobilité, et non pas simplement le contexte technique, d'une population donnée. Nous faisons ici référence aux nouveaux usages qui en découlent, qu'ils soient issus de la sphère privée ou professionnelle. Répondre à cette question permettrait aux entreprises de baser leur réflexion sur des fondations solides... où en sommes-nous et où voulons nous aller ? En plus de se donner le maximum de chance de satisfaire les utilisateurs, l'organisation est alors en mesure d'identifier de manière exhaustive les postulats de l'analyse sécurité à réaliser. Ci-dessous un exemple, non exhaustif des usages des trois populations citées dans notre scénario :

ID	Description	Population(s) concernée(s)	Impact
Téléphone mobile	communication vocale	Comité de direction	**Fort**
		Directeurs à l'international	**Fort**
		Assistante de direction	**Fort**
Accès aux applications métiers	se connecter à distance au système d'information de la société pour accéder aux services et données nécessaires à votre activité	Comité de direction	**Faible**
		Directeurs à l'international	**Fort**
		Assistante de direction	**Moyen**
Email personnel	accès aux services gratuits de courriel	Comité de direction	**Faible**
		Directeurs à l'international	**Faible**
		Assistante de direction	**Faible**
email professionnel	synchroniser ses données avec celle de la boîte mail d'un ordinateur. consulter le courrier électronique de l'entreprise	Comité de direction	**Moyen**
		Directeurs à l'international	**Fort**
		Assistante de direction	**Moyen**
Calendrier	Service de base qui donne à son utilisateur la possibilité de planifier, de noter son emploi du temps, ses rendez-vous, etc.	Comité de direction	**Moyen**
		Directeurs à l'international	**Moyen**
		Assistante de direction	**Fort**

Cette analyse préalable aurait permise de mettre en évidence les besoins de notre assistante, notamment pour les accès depuis chez elle au calendrier et à sa boite mail professionnel. Elle aurait ainsi été en mesure de répondre à l'attende de M. Smith à l'aide de moyen mis à disposition par son entreprise, et non pas personnel.

5.3 Définition des objectifs de sécurité

Cette étape, quelle que soit la méthode choisie (Ebios, MEHARI, ISO 27005), peut permettre d'éviter de tomber dans l'un de ces deux pièges : investir massivement pour l'intégration de l'une de ces technologies alors que d'autres solutions, moins coûteuses auraient suffit ; ou a contrario, passer à côté de risques,

induits par les spécificités de son organisation. L'étape précédente, de définition des usages par population d'utilisateurs, nous a permis de définir clairement les postulats de départ. Ci-dessous un exemple, non exhaustif, des risques auxquels sont exposés les trois populations citées dans notre scénario :

Resource	Indicateur	Besoin de sécurité	Niveau de risque	Évaluation du Risque
R_APPELS	Disponibilité	forte	moyen	significatif
	Intégrité	forte	forte	intolérable
	Confidentialité	forte	moyen	significatif
R_CONNEXION (INTRANET)	Disponibilité	forte	moyen	significatif
	Intégrité	forte	moyen	significatif
	Confidentialité	forte	forte	intolérable
R_MESSAGERIE, R_CARNET_D_ADRESSE et R_CALENDRIER	Disponibilité	forte	moyen	significatif
	Intégrité	moyen	forte	significatif
	Confidentialité	forte	forte	intolérable
R_SI	Disponibilité	forte	forte	intolérable
	Intégrité	forte	forte	intolérable
	Confidentialité	forte	forte	intolérable
R_ACTIVITES_PERSO	Disponibilité	faible	moyen	acceptable
	Intégrité	faible	moyen	acceptable
	Confidentialité	faible	moyen	acceptable

5.4 Matrice d'expression des besoins

Une fois cette réflexion sur les usages, puis les risques, menée, il est alors possible de définir une matrice d'exigences, à laquelle les solutions techniques et organisationnelles devront répondre.

Conclusion : La sécurité des usages en mobilité, une démarche globale

Une prise en compte de tous les usages, présents et futurs, au sein de l'entreprise est nécessaire pour définir des objectifs et des mesures de sécurité. La prise en compte de la sécurité concernant des projets de déploiement de flotte de Smartphones, ou de tout autre moyen de communication en mobilité, ne doit pas être traitée de manière classique. L'analyse des risques, la définition des objectifs de sécurité et la définition des solutions à mettre en œuvre, ne peuvent aboutir à une solution satisfaisante sans prendre en considération les objectifs d'usage.

Satisfaire les besoins des utilisateurs pour garantir la sécurité en mobilité. La mise en œuvre de technologie ne répondant pas aux besoins des utilisateurs aboutira fatalement au contournement des mesures de sécurité. Les personnes disposant d'un Smartphone personnel s'accroit jour après jour. Le risque de voir ces employés utiliser leur device personnel pour des usages professionnels est proportionnel à l'insatisfaction relative aux moyens technologiques mis à leur disposition par leur entreprise.

La dépérimétrisation : facteur clé de succès Cette démarche ne doit surtout pas être menée avec des postulats « rigides », communément utilisés dans le monde de la sécurité. Se limiter à un périmètre des usages strictement professionnels, des populations d'utilisateurs les plus visibles, aux besoins les plus évident, ne permettra pas d'aboutir à une analyse exhaustive des besoins en termes d'usage et de sécurité. Ce manque d'exhaustivité risque de ne contenter personne : ni les utilisateurs et leurs usages, ni les responsables de la sécurité et leurs risques.

Vers une politique de mobilité

Pierre-Yves Bouf

AVIXA

1 Pourquoi la mobilité est-elle un problème ?

Se déplacer en emportant avec soi une partie du patrimoine immatériel de l'entreprise est vécu comme une situation à risque dont le traitement est compliqué.

Dans l'entreprise ou l'Administration où nous travaillons, les questions de sécurité de l'information ne sont pas toujours évidentes, le fait de quitter physiquement les lieux ajoute une difficulté supplémentaire. Isolement, éloignement, vulnérabilité, complexité... sont les termes les plus souvent évoqués par le collaborateur en mission.

Quels sont les facteurs de cette complexité :

- les déplacements se font dans l'urgence, le principal souci est l'organisation matérielle de la mission. L'aspect « sécurité de l'information » est considéré, souvent, comme un accessoire supplémentaire, pénible et coûteux,
- la décision de partir est prise au dernier moment : même si la politique de l'entreprise prévoit des procédures de sécurité, elles sont évitées...
- en l'absence de réflexion préalable, on se focalise sur « l'outil » ou « la solution de sécurité », c'est-à-dire le dispositif numérique miracle qui nous met à l'abri des ennuis : la sécurité n'est dans ce cas pas un fait mais un sentiment procuré par le côté rassurant de la technique,
- on ignore tout simplement quelle est la vraie sensibilité de l'information qu'on manipule, on ne sait où placer sa paranoïa,
- on ignore ce que contiennent vraiment les supports numériques qu'on emporte avec soi,
- on se focalise sur le numérique en oubliant totalement les autres formes d'information. On s'étonne ensuite de « fuites » en soupçonnant des intrusions compliquées alors qu'un simple bavardage suffit...
- on ne s'est pas renseigné sur les spécificités du pays visité et on se trouve dans des situations imprévues – et vécues comme ingérables (que dire à l'officier de l'aéroport qui exige le déchiffrement de vos données ?).

2 Procéder par ordre

Une politique pensée suffisamment tôt, au calme, en concertation avec les parties prenantes, constitue un premier rempart efficace. La technique sera alors – enfin – un complément utile et justifié.

La démarche n'est pas une nouveauté, elle ne fait que décliner à la mobilité le bon sens d'une politique de sécurité de l'information.

Il s'agit surtout de bon sens.

2.1 Savoir quoi protéger

Bien qu'évident, ce point est rarement clair : raisonner en terme d'impact est une approche pragmatique : « et si... »

– l'information est connue de tous
– l'information est perdue
– l'information est modifiée

est-ce grave ?

Ceci permet de caractériser les besoin en sécurité des informations, avant même de parler de menace. L'intérêt de la démarche, si elle n'a pas déjà été faite dans la PSSI de l'organisation, c'est que la conclusion est toujours vraie, c'est le cahier des charges de la sécurité de l'information (on utilise souvent les critères DICP).

Dans cette réflexion, il convient de ne pas tomber dans le piège (fréquent) de confondre « besoin en sécurité » et « dispositif de sécurité ».

2.2 Analyser les menaces

Les menaces ne s'exercent pas sur l'information elle-même (ce qui n'aurait pas tellement de sens) mais sur les supports de cette information. Or l'information existe sous 3 formes : numérique, physique et intellectuelle.

Un bref aperçu des menaces classiques est évoqué en conférence.

Le retour d'expérience de missions précédentes, les documents de l'ANSSI, une discussion avec les professionnels de la SSI, les ambassades... sont des sources d'information intéressantes pour préciser ces menaces pour notre organisation, en se focalisant sur celles liées aux missions.

Il convient à ce stade de rester sur des scénarios réalistes qui correspondent au contexte (on peut négliger le risque de chute de météorite sur le missionnaire, mais pas l'écoute des conversations dans les chambres d'hôtel...).

2.3 Vers des mesures de sécurité

La « boite à outil » des mesures de sécurité à mettre en place est définie en examinant comment les informations sensibles, sous les 3 formes évoquées, peuvent être vulnérables aux menaces envisagées.

Cette boite à outils comporte obligatoirement 4 composantes :
– la technique
– l'organisation
– les procédures
– le comportement.

Ces 4 composantes sont complémentaires ; en particulier la technique seule n'a jamais résolu un problème de sécurité et est souvent responsable d'une fausse impression d'immunité.

2.4 Les spécificités de la mobilité

La réflexion évoquée se fait au calme, pas la veille du départ.

Pour que la mission se fasse sereinement, il convient de respecter quelques pratiques de bon sens : les personnels de l'organisation doivent être personnellement convaincus, motivés et moteurs dans l'application de la politique de sécurité de l'information : ceci se fait au moyen de campagnes de sensibilisation, pertinentes régulières... et agréables, les outils (technique, procédures...) doivent consister en des kits bien identifiés qui auront été expliqués en détail par le RSSI. Ceci peut aussi se faire sous forme de formations préalable pour les populations concernées, la dimension « humaine » de la formation est importante : l'expérience montre que pour obtenir une information, il suffit souvent de la demander. Les techniques classiques de renseignement et de manipulation devront faire partie de la formation et de la sensibilisation des personnels, un missionnaire est un individu pressé, parfois stressé, tout doit être fait pour lui faciliter la tâche : check-lists, outils automatiques, liste de numéros à appeler au cas où, liste des procédures... pour qu'il (elle) n'ait pas le sentiment que la sécurité est un boulet de plus à traîner ou une source d'angoisse, déculpabiliser les incidents : la nature humaine est ainsi faite que nous sommes tous statistiquement victimes de notre inattention, oubli, fatigue, etc.. La remontée d'incident ne doit pas être uniquement une perspective de punition pour le missionnaire, sous peine de créer des situations ingérables et perdre le bénéfice de l'expérience. Ce point est à doser avec soin, les rapports d'adulte à adulte dans l'organisation sont une base incontournable.

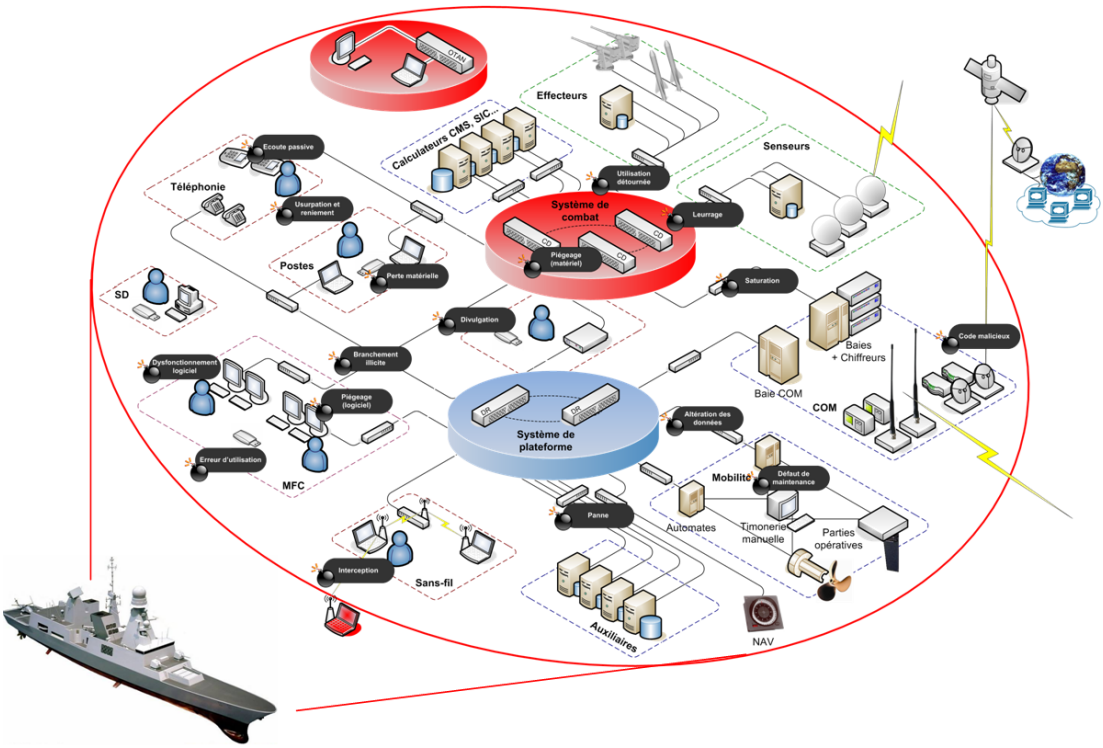
Enfin, les conseils pratiques sont à établir au cas par cas, suivant l'activité et les pays visités, il serait fastidieux d'en faire le tour dans le cadre de la conférence. Le guide du voyageur de l'ANSSI constitue une bonne base de départ. La réflexion interne, les retours d'expérience, les informations glanées auprès des professionnels feront le reste.

Security in navy systems, Industrial point of view

Patrick Hébrard, Laurent Comte

DCNS

Most of navy systems (on a battleship or navy base operational centre) integrate harware/software civilian technologies (PC windows, PC Linux, IP Network, IP Phones, WIFI, WEB server, SQL...) and as a consequence integrate also their vulnerabilities as depicted as below :



With the increase of the risk level and because of these systems handle military devices (Radar, Electronic Warfare systems, Weapons, Missiles, Drones...) the consequences of corruption could be very dangerous on both combat system (e.g. illicit weapon usage, sensitive data bases corruption) and ship management system (e.g motors and navigation bare management system corruption, navigation, GPS data corruption...).

The specificities of a battleship at sea increase the problem complexity (updating of antivirus bases, security management – SIEM...).

Another point is the difficulties the industry has to protect critical software against retro engineering, particularly for exportation under TOT requirements.

This presentation depicts these different aspects where security leak could have major impacts.

Communications Opportunistes : Défis de Sécurité

Abdullatif Shikfa[1] ⋆, Melek Önen[2] et Refik Molva[2]

[1] Alcatel-Lucent Bell Labs,
Route de Villejust, 91620 Nozay, France,
[2] EURECOM,
2229, route des crêtes, 06560 Sophia Antipolis cedex, France

Résumé. Les réseaux opportunistes reposent sur un paradigme de communication très prometteur qui dépasse de loin les possibilités des réseaux mobiles ad-hoc (MANET). L'établissement de communication dans cet environnement est à lui seul un défi important, mais le défi est encore plus grand pour sécuriser ces communications. Les caractéristiques des réseaux opportunistes, et en particulier l'absence de connectivité bout-en-bout, demandent en effet de complètement repenser les solutions de sécurité traditionnelles. Cet article aborde donc divers défis liés aux communications opportunistes du point de vue de la sécurité, allant des problématiques de coopération juste entre les noeuds à l'authentification et l'intégrité des messages, sans oublié les difficiles problèmes de confidentialités des échanges et de respect de la vie privée des utilisateurs.

Mots-clé:: Réseaux mobiles, Communication opportuniste, Sécurité des réseaux.

1 Introduction

Imaginons un nouveau type de communication, dans lequel l'émetteur n'aurait pas besoin de spécifier le destinataire du message explicitement (via une adresse par exemple). Le destinataire du message serait au contraire implicitement défini via le contenu du message, et il en irait de même pour le routage. L'architecture réseau classique divisée en couches devrait alors être remplacée par une architecture condensée.

Ce type de communication est le fondement des communications opportunistes, et ce n'est pas une utopie mais d'ors et déjà une réalité, un domaine de recherche important au sujet duquel des centaines d'articles scientifiques ont été publiés, et qui est le sujet principal de nombreux projets de recherche récents (dont le projet Haggle [8]) et d'un groupe spécial de l'Internet Research Task Force appelé le Delay-Tolerant Networking Research Group. La raison pour laquelle ce nouveau paradigme est l'objet d'autant d'attentions est le nombre important d'applications qui peuvent en découler. Les communications opportunistes visent en effet à rendre possible la communication en présence de conditions hostiles dans lesquels les méthodes de communication classiques échouent

⋆. Travail effectué en partie à EURECOM.

(par exemple pour des communication ad hoc très dynamiques, en réponse à des catastrophe naturelles qui mettraient les infrastructures de communications traditionnelles hors-service, ou tout simplement dans des cas où le déploiement d'une infrastructure classique n'est pas rentable, comme dans les régions à faible densité de population), mais elles renouvellent également l'intérêt porté à des architectures de communication alternatives, comme les systèmes pub/sub basés sur le contenu, ou encore les réseaux basés sur le contenu promus récemment par Van Jacobson.

Du point de vue de la sécurité, le principal problème réside dans le fait que la protection de la vie privée ou de la confidentialité des données requiert un chiffrement du contenu de ces données, et dans le même temps ce contenu est à la base des décisions de transmission et de routage. Les exigences de sécurité entrent donc en conflit avec les besoins de transmission. Afin de dépasser cet apparent paradoxe et d'être en mesure de proposer des protocoles de communication opportuniste sécurisés, il faut donc mettre en oeuvre des solutions permettant de réaliser certaines opérations utiles à la transmission sur des données chiffrées. La suite de cet article introduit plus précisémment les différents types de routage opportunistes, puis exposent les principaux problèmes de sécurité dans cet environnement.

2 Routage et Transmission dans les Réseaux Opportunistes

Les principaux objets d'études de cet article sont les communications opportunistes sécurisées et, pour cerner ces problèmes efficacement, il est essentiel dans un premier temps de bien comprendre la définition des réseaux opportunistes et les contraintes qui en découlent.

Les réseaux opportunistes peuvent être vus comme une extension des réseaux ad hoc (MANET) et partagent donc leurs caractéristiques :

- ad hoc, qui implique essentiellement l'absence d'infrastructure et une organisation spontanée du réseau,
- ressources limitées car les noeuds du réseau sont souvent des appareils portables (des ordinateurs portables, des assistant personnels, des téléphones mobiles, ou même de simples senseurs),
- topologie dynamique du réseau due à la mobilité des noeuds.

Pour le dernier point, il est intéressant de noter que les MANET supportent la mobilité de façon très incomplète : le routage des messages se base en effet sur une route fixée entre l'émetteur et le récepteur, et toute modification de topologie durant une communication requiert de recalculer l'intégralité de la route. Cette démarche considère la mobilité comme un obstacle à surmonter et la communication n'est possible qu'en présence d'une topologie considérée comme stable

pour un moment puis qui change vers une nouvelle configuration stable : cette approche n'est applicable que dans le cadre de réseaux à mobilité lente. Les réseaux opportunistes considèrent au contraire que la mobilité des noeuds peut être importante et exploite cette mobilité comme un avantage plutôt qu'un inconvénient : la mobilité physique est une possibilité supplémentaire de porter les messages pour palier à l'éventuelle l'absence de moyens de communications plus rapides et efficaces.

De là se dégage une autre caractéristique fondamentale des réseaux opportunistes, à savoir la tolérance au délai. Les réseaux opportunistes offrent en effet un support complet de la mobilité et, à ce titre, ne supposent pas l'existence d'un chemin de bout-en-bout. Le but de la communication opportuniste est en effet de porter le message à des relais de plus en plus près de la destination, et pour ce faire une stratégie dite *"store, carry and forward"* est adoptée en lieu et place du routage traditionnel :

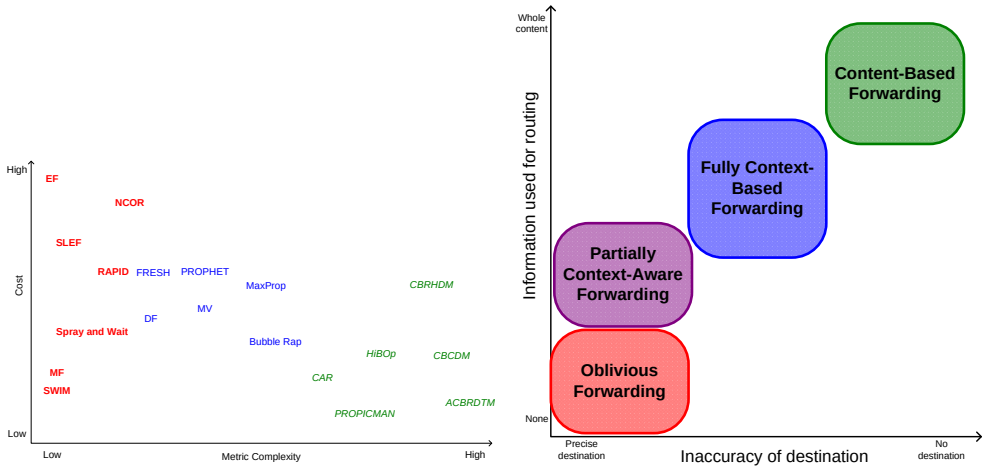
- store (stoker) : les messages sont stockés en mémoire des noeuds mobiles en attendant une opportunité de communication,
- carry (porter) : les messages sont portés sur une certaine distance en exploitant la mobilité physique des utilisateurs,
- forward (transmettre) : lorsqu'une opportunité de communication avec un noeud plus proche de la destination se présente, le message est transmis au noeud en question.

Cette stratégie ne requiert donc pas d'établissement de route de bout-en-bout et s'accommode de la mobilité ou de la défaillance de certains noeuds : elle compense l'absence de connectivité bout-en-bout par la mobilité physique et la tolérance au délai.

Enfin, une troisième caractéristique majeure des communications opportunistes réside dans la structure condensée des messages. En effet, les messages sont transmis via des réseaux hétérogènes, et la structure des messages ne doit donc pas être dépendante d'un protocole en particulier. Ainsi toutes les informations concernant à la fois le contenu du message et la description de la destination (pour le routage) doivent être disponibles à un haut niveau d'abstraction, ce qui justifie une structure condensée. Cette structure est particulièrement adaptée aux communications multicast (multi-transmission) pour disséminer une information à plusieurs récepteurs, car elle permet d'envisager des protocoles de communication riches qui exploitent l'intégralité du contenu du message pour prendre les décisions de transmission et pas seulement un identifiant unique (une adresse par exemple) de la destination.

Prenant en compte ces différentes caractéristiques de nombreux protocoles adaptés aux communications opportunistes ont été proposés par la communauté scientifique ces dernières années. Nous les avons répertoriés puis classifier en fonction du coût de ces protocoles en terme d'utilisation du réseau et de la complexité

de l'opération d'évaluation de la distance d'un noeud à une destination dans un premier temps (Fig. 1(a)), puis, faisant abstraction des problèmes de coût, en fonction de la quantité d'information utilisée pour prendre les décisions de transmission et de la précision avec laquelle est définie la destination (Fig. 1(b)). Nous



(a) (b)

Fig. 1. Classifications des protocoles de communication opportunistes.

avons alors dégagé trois grandes catégories de protocoles de communication opportunistes :

- Les protocoles de transmission aveugles, au sein desquels la destination du message est définie de façon très précise et explicite (via un identifiant unique). Ces protocoles adoptent principalement des stratégies de transmission épidémiques pour atteindre la destination, en se focalisant sur des heuristiques visant a réduire l'impact d'une telle approche sur la charge du réseau.
- Les protocoles de transmission basés sur le contexte, dans lesquels la destination du message est définie implicitement en fonction de son profil. Les décisions de transmission sont prises en comparant le contexte du message (correspondant au profil de la destination) avec le profil du noeud rencontré, et les messages sont transmis à des noeuds avec taux de correspondance de plus en plus élevé. Au sein de cette catégorie il est possible de distinguer deux sous-ensembles : les protocoles complètement basés sur le contexte (qui prennent en compte l'intégralité du contexte dans le cadre des décisions de transmissions) et les protocoles partiellement basés sur le contexte qui ne prennent en compte qu'une information contextuelle bien précise (comme l'historique de rencontre ou les relations sociales des noeuds).

– Les protocoles de transmission basés sur le contenu, qui ne définissent pas de destination du tout : les noeuds expriment leurs intérêts pour un certain type de contenu, et les messages leur sont adressés en fonction de leur contenu.

L'intérêt de cette classification est de mettre en lumière les différences conceptuelles fondamentales entre ces trois catégories et l'évolution de la façon dont est considérée la destination (pour simplifier destination explicite, destination implicite et pas de destination). Ces trois catégories de transmission radicalement différentes posent des défis distincts, tant du point de vue de la transmission que du point de vue de la sécurité. C'est ce dernier aspect qui nous intéresse et nous détaillons donc les problématiques de sécurité dans les réseaux opportunistes dans la section suivante.

3 Problématiques de Sécurité dans les Réseaux Opportunistes

Les problématiques de sécurité sont une composante essentielle de tout système de communication, et de nombreuses solutions ont été apportées pour résoudre les divers aspects de ce problème dans les réseaux de communication traditionnels. Ces solutions ne sont toutefois pas adaptées aux besoins et contraintes spécifiques des réseaux opportunistes. En effet, en plus des contraintes classiques des réseaux ad hoc mobiles MANET, qui requièrent des solutions de sécurité dynamiques, locales et auto-organisantes, la tolérance au délai et l'absence de connectivité bout-en-bout signifient que les protocoles interactifs entre émetteur et récepteur sont irréalisables dans les réseaux opportunistes et que l'accès à un serveur de sécurité ne peut être envisagé au cours de la communication. Enfin, l'architecture condensée soulève également de nouvelles difficultés, car elle implique que les problèmes de sécurité liés au contenu des applications et ceux liés aux informations de transmission doivent être traités de manière globale, contrairement au cas de l'architecture classique où la sécurité de chaque couche peut être assurée indépendamment des autres. Nous présentons donc dans cette section les différentes problématiques de sécurité dans les réseaux opportunistes en commençant par les problèmes généraux de coopération entre les noeuds puis ceux liés à l'intégrité, la confidentialité et le respect de la vie privée.

3.1 Coopération

La coopération entre les noeuds est essentielle au bon fonctionnement de tous les réseaux pairs-à-pairs, en particuliers les MANETs et réseaux opportunistes. Les solutions classiques pour éviter le développement de noeuds égoïstes qui ne participent pas aux opérations de transmission pour économiser leurs ressources énergétiques peuvent être classées en deux grandes catégories :

- mécanismes basés sur la réputation [4,5,10], où les noeuds acceptent de coopérer avec leurs voisins en fonction de l'historique de comportement de ces derniers, qui est mesuré par leur réputation (cette dernière croit lors d'un bon comportement et décroit pour les noeuds qui adoptent des comportements égoïstes),
- mécanismes basés sur les récompenses [14,9,7,6,17] dans lesquels les noeuds reçoivent une certaine récompense lorsqu'ils coopèrent, récompense qui peut ensuite être utilisée dans leur propre intérêt lorsqu'ils ont besoin du réseau à leur tour.

Les solutions existantes dans ces deux catégories ne sont en général pas adaptées aux réseaux opportunistes car elles font appel à une entité tierce de confiance qui doit être accessible à tout moment de la communication. Nous avons donc proposé une solution alternative [12] basée sur le principe de la patate chaude, dans laquelle les noeuds prennent la décision d'accepter ou non un message de façon aveugle.

Lorsque un noeud N_1 a un message M à transmettre, il en informe ses voisins sans spécifier la destination du message M. Les voisins doivent alors décider à l'aveugle s'ils sont intéressés par ce message ou non. Supposons que le noeud N_2 est intéressé, N_2 envoie une récompense à N_1 qui lui transmet ensuite M. Si M intéresse effectivement N_2, alors N_2 a payé pour un message qui lui est destiné ce qui est équitable, mais si N_2 n'est pas intéressé par M, N_2 sera incité à transmettre le message à d'autres noeuds pour récupérer la récompense et ainsi la transmission du message et la coopération entre les noeuds sont assurées. Cette approche est optimiste dans le sens où une autorité de confiance n'est requise que pour :

- convertir les récompenses reçues par les noeuds en ressource utilisable dans le système,
- résoudre les conflits entre les noeuds qui se produisent si le noeud N_1 n'envoie pas le message M à N_2 après avoir reçu une récompense de la part de N_2.

Ce protocole assure donc une transmission équitable optimiste car l'autorité de confiance n'est requise qu'en cas de conflit entre les noeuds et même dans ce cas, l'accès à cette autorité n'a pas besoin d'être immédiat : cette autorité est dite hors-ligne (offline). Enfin ce protocole est flexible car il permet aux noeuds qui ne désirent pas coopérer (parce que leurs ressources restantes sont faibles) de ne pas le faire s'ils acceptent le risque de rater des messages qui leurs sont destinés, et il est indépendant du protocole de transmission et notamment du processus de choix du prochain noeud.

3.2 Authentification et intégrité

L'authentification des messages est un besoin de sécurité essentiel dans tout système de communication, et les solutions classiques consistent pour la source à signer le message avec une clef privée et un certificat associé. Cette solution peut être directement déployée dans les réseaux opportunistes car elle ne requiert pas de connectivite de bout-en-bout, mais simplement une phase antérieure à la communication, au cours de laquelle chaque noeud doit faire établir un certificat auprès d'une autorité de certification, qui est donc considérée hors-ligne et n'entre pas en conflit avec la communication opportuniste à proprement parler.

L'authentification bout-en-bout ne pose donc pas de souci particulier si les messages n'ont pas besoin d'être modifiés en cours de route. Cette dernière situation se produit toutefois dans deux cas intéressant dans le cadre des réseaux opportunistes :

- Lorsque les messages ont besoin d'être fragmentés pour avoir une taille de paquet conforme à une technologie réseau particulière. Une solution évidente est alors d'authentifier chaque fragment, mais des solutions plus intelligentes qui permettent l'authentification de l'ensemble des fragments de façon plus efficace existent (en se basant notamment sur des arbres de Merkle [2]).
- Lorsque le codage réseau (network coding) est utilisé comme protocole de transmission. Le codage réseau est en effet très intéressant du point de vue performance pour disséminer une information dans une approche épidémique, mais il requiert de nombreuses modifications des paquets à chaque transmission. En effet chaque noeud doit transmettre une combinaison linéaire de l'ensemble des messages qu'il a reçu. En contrepartie de sa performance, le codage réseau est exposé à un risque très important de pollution : si un noeud envoie une mauvaise combinaison de messages, l'ensemble des noeuds du réseau peut se retrouver infecté par cette mauvaise combinaison et incapable de décoder le message final. Pour palier à ce type d'attaque il faut donc prévoir un mécanisme d'authentification des paquets qui permette aux noeuds intermédiaires de créer la signature d'une combinaison de messages à partir de la signature de chacun des messages reçus. Ce type de signature présente donc un caractère d'homomorphisme, puisque les signatures sont compatibles avec l'opération de combinaison linéaire. Les besoins de ces signatures sont donc très spécifiques et contraires aux besoins classiques des signatures (où la compatibilité avec l'opération de combinaison linéaire serait considérée comme un défaut, et nous avons proposé une solution à ce problème en modifiant un schéma de signature basé sur les couplages bilinéaires dans les courbes elliptiques [11,13].

3.3 Confidentialité et respect de la vie privée

Garantir la confidentialité des messages est un autre aspect fondamental de la sécurité des réseaux. Les solutions classiques pour ce problème passent par le chiffrement des données de bout-en-bout. On distingue deux grandes familles de chiffrements :

- les méthodes de chiffrements symétriques dans lesquels l'émetteur et le récepteur doivent partager une clef secrète,
- les méthodes de chiffrements asymétriques où chaque noeud possède généralement une clef publique (qui est accessible à tous) et une clef privée (que seul le noeud connaît).

Le cas des chiffrements symétriques ne peut être employé pour assurer la confidentialité bout-en-bout, car il suppose que l'émetteur et le récepteur entrent dans une phase interactive d'établissement de la clef secrète, ce qui entre en conflit avec l'absence de connectivité bout-en-bout.

Les méthodes de chiffrement asymétriques sont plus adaptées a priori, mais soulèvent tout de même un problème : contrairement au cas de la signature où l'émetteur du message peut envoyer son certificat en même temps que le message qu'il a signé, le chiffrement requiert de l'émetteur la connaissance de la clef publique du récepteur avant l'envoi du message. Or ce certificat est en général disponible soit directement auprès du récepteur, soit auprès d'une entité tierce de confiance (l'autorité de certification par exemple). Bien que le chiffrement asymétrique ne soit donc pas interactif à proprement parler, la phase d'obtention du certificat du récepteur entre en conflit avec les contraintes des communications opportunistes. Pour contourner ce problème, il est possible d'utiliser le chiffrement à base d'identité [3] comme proposé par Asokan et al. [1]. Le chiffrement basé sur l'identité permet en effet de dériver la clef publique de la destination à partir de l'identité de cette dernière, et permet donc de se passer de certificats. Le chiffrement basé sur l'identité offre donc une solution crédible au problème de la confidentialité lorsque l'identité de la destination est connue comme c'est le cas pour les protocoles de transmission aveugles.

Le défi reste entier en revanche pour les protocoles plus riches comme par exemple les protocoles basés sur le contexte où l'identité de la destination n'est pas connue mais peut être déduite implicitement. Cette catégorie de protocoles requiert donc de nouvelles méthodes de chiffrement qui permettent de dériver une clef de chiffrement à partir de la définition implicite de la destination et qui fassent en sorte que seule la destination puisse dériver la clef de déchiffrement associée.

De façon plus générale, le problème de respect de la vie privée couvre plusieurs aspects dont la confidentialité du contenu mais aussi la confidentialité de la communication, c'est-à-dire empêcher un noeud d'analyser le trafic pour savoir quel émetteur communique avec quel récepteur. Les besoins de protection de la vie

privée peuvent être considérés à différents niveaux, et le niveau requis dépend de l'application, du point de vue adopté (émetteur, récepteur, noeud intermédiaire, noeud extérieur) and du niveau de confiance entre les entités. En nous basant sur nos travaux [15], nous définissons un cadre général de modèles de respect de la vie privée, en considérant une donnée privée D_1 appartenant au noeud N_1 et qui doit être traitée par le noeud N_2 (pour prendre une décision de transmission par exemple) comme suit :

- **modèle 1, absence de secret** : ce modèle correspond au cas où N_1 ne requiert pas de protection pour D_1 du tout, N_2 (ou n'importe quel autre noeud) a accès à D_1 en clair dans le processus.
- **modèle, secret binaire** : dans ce modèle N_1 fait entièrement confiance à certains noeuds et pas du tout aux autres. Ainsi, si N_2 appartient au groupe de confiance de N_1, N_2 a accès à l'intégralité de D_1 en clair sinon N_2 n'a pas accès à D_1.
- **modèle 3, secret adaptable** : dans ce modèle, le niveau de protection dépend de la relation entre les noeuds : N_1 fait partiellement confiance à N_2. Le niveau de confiance peut être basé sur l'appartenance à une communauté. N_2 doit alors être en mesure d'accéder à une partie de D_1 variable selon le niveau de confiance.
- **modèle 4, secret complet** : contrairement aux modèles précédents, ce modèle fait référence au cas où les noeuds n'ont aucune confiance les uns envers les autres, et dans ce cas N_2 doit être en mesure de manipuler les données D_1 sans y avoir accès en clair.

La nature de la donnée secrète D_1 (qui peut correspondre notamment au contexte ou au contenu) ainsi que le niveau de confiance et de protection requis dépendent du scenario considéré. De façon générale, le défi est de permettre la prise de décision de transmission dans les différents modèles. En particulier, les modèles de respect de la vie privée les plus exigeants (3 et 4) demandent des solutions sophistiquées de calcul sur des données chiffrées, comme celles que nous avons présentés dans les travaux [16,15].

4 Conclusion

Les réseaux opportunistes reposent sur un paradigme de communication très prometteur et plein de défis. Jusqu'à présent l'immense majorité des efforts de recherche dans ce domaine s'est focalisée sur les aspects purement réseaux du problème. Nous sommes les premiers à avoir analysé dans le détail les problématiques de sécurité liées aux communications opportunistes et à avoir proposé un ensemble de solutions couvrant une large gamme de besoins de sécurité.

Références

1. N. Asokan, Kari Kostiainen, Philip Ginzboorg, Jörg Ott, and Cheng Luo. Applicability of identity-based cryptography for disruption-tolerant networking. In *MobiOpp '07 : Proceedings of the 1st international MobiSys workshop on Mobile opportunistic networking*, pages 52–56. ACM, 2007.
2. N. Asokan, Kari Kostiainen, Kari Kostiainen, Philip Ginzboorg, Philip Ginzboorg, Jörg Ott, Cheng Luo, and Cheng Luo. Towards Securing Disruption-Tolerant Networking. Technical Report NRC-TR-2007-007, Nokia Research Center, march 2007. `http://research.nokia.com/files/NRC-TR-2007-007.pdf`.
3. Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In *CRYPTO '01 : Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology*, pages 213–229. Springer-Verlag, 2001.
4. Sonja Buchegger and Jean-Yves Le Boudec. Nodes Bearing Grudges : Towards Routing Security, Fairness, and Robustness in Mobile Ad Hoc Networks. In *Proceedings of the 10th Euromicro Euromicro Workshop on Parallel, Distributed and Network-based Processing*, pages 403 – 410. IEEE Computer Society, 2002.
5. Sonja Buchegger and Jean-Yves Le Boudec. Performance analysis of the confidant protocol. In *MobiHoc '02 : Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, pages 226–236. ACM, 2002.
6. Levente Buttyán and Jean-Pierre Hubaux. Stimulating cooperation in self-organizing mobile ad hoc networks. *Mobile Networks and Applications*, 8(5) :579–592, 2003.
7. Anargyros Garyfalos and Kevin C. Almeroth. Coupon Based Incentive Systems and the Implications of Equilibrium Theory. In *CEC '04 : Proceedings of the IEEE International Conference on E-Commerce Technology*, pages 213–220. IEEE Computer Society, 2004.
8. The Haggle Project, 2006. `http://www.haggleproject.org/index.php`.
9. Andreas Heinemann, Jussi Kangasharju, O Lyardet, and Max MÄuhlhÄauser. iClouds – Peer-to-Peer Information Sharing in Mobile Environments. In *Proceedings of the 9th International Euro-Par Conference, (Euro-Par 2003), volume 2790 of Lecture Notes in Computer Science*, pages 1038–1045. Springer, 2003.
10. Pietro Michiardi and Refik Molva. CORE : a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. In *Proceedings of the IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security*, pages 107–121. Kluwer, B.V., 2002.
11. Melek Önen, Abdullatif Shikfa, and Refik Molva. Haggle deliverable 4.1 : Preliminary design of trust and security mechanisms, June 2007. `http://www.haggleproject.org/deliverables/D4.1\_final.pdf`.
12. Melek Önen, Abdullatif Shikfa, and Refik Molva. Optimistic fair exchange for secure forwarding. In *MOBIQUITOUS '07 : Proceedings of the 2007 Fourth Annual International Conference on Mobile and Ubiquitous Systems : Networking&Services (MobiQuitous)*. IEEE Computer Society, 2007.
13. Melek Önen, Abdullatif Shikfa, and Refik Molva. SigNCode : A provably secure homomorphic signature scheme for network coding. Technical Report RR-07-202, Department Network and Security, EURECOM, September 2007. `http://www.eurecom.fr/util/publidownload.fr.htm?id=2337`.
14. Olga Ratsimor, Tim Finin, Anupam Joshi, and Yelena Yesha. eNcentive : a framework for intelligent marketing in mobile peer-to-peer environments. In *ICEC '03 : Proceedings of the 5th international conference on Electronic commerce*, pages 87–94. ACM, 2003.
15. Abdullatif Shikfa, Melek Önen, and Refik Molva. Privacy in Content-Based Opportunistic Networks. In *WAINA '09 : Proceedings of the 2009 International Conference on Advanced Information Networking and Applications Workshops*, pages 832–837. IEEE Computer Society, 2009.
16. Abdullatif Shikfa, Melek Önen, and Refik Molva. Privacy and confidentiality in context-based and epidemic forwarding. *Computer Communications*, 33(13) :1493–1504, 2010.

17. Sheng Zhong, Jiang Chen, and Yang Richard Yang. Sprite : a simple, cheat-proof, credit-based system for mobile ad-hoc networks. In *INFOCOM 2003 : Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 3, pages 1987–1997, March-April 2003.