Du smartphone au frigo : la grande évasion du mot de passe

Serge Borderes

Du smartphone au frigo : la grande évasion du mot de passe

Serge Bordères
Centre d'Etudes Nucléaires de Bordeaux-Gradignan
Route du Solarium
33175 GRADIGNAN

Résumé

Le mot de passe, un secret bien gardé... par tellement de logiciels et d'appareils connectés que Polichinelle ne le renierait pas ! Tous ces appareils, à commencer par les smartphones, ont débarqué dans nos vies privées et envahissent, silencieusement, nos environnements professionnels. L'offre en ligne des acteurs d'Internet donne également des solutions alternatives aux utilisateurs et oblige nos systèmes d'information à s'aligner. Cette déferlante technologique a des répercussions non négligeables sur la sécurité de nos systèmes d'information car elle démultiplie les risques d'évasion et de compromission des mots de passe qui sont bien souvent la seule défense des services. Bien malin qui peut dire aujourd'hui où sont les mots de passe des utilisateurs. Peut-être dans un frigidaire.

À une époque hyper-technologique et hyper-connectée, le mot de passe trahit de plus en plus son archaïsme. C'est comme si on voulait parler de sécurité routière et que nos voitures, bardées d'électronique et fonçant à 130 à l'heure sur l'autoroute, étaient équipées de roues de diligence en bois ! Le défi est important puisqu'il s'agit de maintenir la protection de nos activités numériques professionnelles tout en étant capable de profiter du meilleur des technologies qui déboulent dans la société.

Dans un premier temps nous verrons comment l'authentification par de simples mots de passe pour des services en ligne favorise l'usage de matériels personnels, la dispersion et la fuite des identifiants. Dans un deuxième temps sera exposée la méthode mise en place expérimentalement au Centre d'Etudes Nucléaires de Bordeaux-Gradignan (CENBG) afin de réduire les risques induits par les sacro-saints mots de passe et de restreindre leur diffusion incontrôlable.

Mots-clefs

mobiles, Android, iOS, mot de passe, authentification, certificats, Oauth, Openvpn, Radius

1 Introduction

Le mot de passe authentifiait déjà nos connexions à une époque où Internet n'existait pas et les ordinateurs ne sortaient pas de nos bureaux. Aujourd'hui il est toujours là et pourtant tout a changé dans l'environnement technologique. Internet, les micro-ordinateurs nomades professionnels et personnels, le Cloud, les smartphones, les tablettes et maintenant des milliards d'objets connectés dont on nous promet la déferlante.

Tous ces objets veulent se connecter facilement et automatiquement à tous nos comptes et embarquent tous nos identifiants. Nous commençons à voir les problèmes par le biais des smartphones mais ils ne sont que la partie visible de l'iceberg qui, peu à peu, efface les limites traditionnelles du système d'information. « Avant », on savait où il commençait et où il se terminait. Mais aujourd'hui, avec cette dispersion, il n'y a même plus de ligne de front. Les pirates ont déjà bien compris la valeur des identifiants et font preuve d'une grande imagination pour y accéder.

2 A qui la faute ?

L'authentification directe par simple mot de passe est très répandue, souvent par simplicité. Les services se retrouvent directement exposés sur Internet telle une vitrine faiblement protégée. Non seulement sept milliards d'individus peuvent tenter leur chance mais il est impossible de savoir avec quel matériel se connecte l'utilisateur, ni son niveau de protection. L'appareil n'apparaît pas sur notre radar, on ne connaît pas son historique et il est certain que les identifiants y sont enregistrés dans des conditions inconnues et, puisque tout cela baigne dans un environnement privé, qui y a accès. On ne sait plus répondre à la question « qui se connecte avec quoi ? »

Les objets connectés sont en train de repousser les frontières du phénomène. Par exemple, pourquoi ne pas configurer ma messagerie dans mon téléviseur ? C'est pratique, mais qui a accès, aux messages et au mot de passe inscrit en clair dans les paramètres ? Et que penser du gars de la maintenance quand elle tombera en panne ?

On sait déjà que des vagues de SPAM ont été causées par des milliers d'objets connectés, y compris par des réfrigérateurs ! En août 2015, la société Pen Test Partners a montré qu'il était possible de pirater un réfrigérateur par une attaque man-in-the-middle. De plus en plus d'annonces du même type sont faites dans les conférences spécialisées.([1] et [2])

Pour la première fois dans l'histoire de l'informatique, des nouvelles technologies ont été introduites dans la sphère privée avant la sphère professionnelle. Chacun se trouve alors exposé à de multiples influences agissant de façons désordonnées, telles que le cadre familial, amical, commercial ou les réseaux sociaux. Fatalement, la voix de l'administrateur systèmes et réseaux porte de moins en moins. Les recommandations, chartes et PSSI sont mises à mal, d'un côté par l'ambiance sociétale, et de l'autre parce que l'accumulation des consignes finit par complètement assommer l'utilisateur au point de lui demander une vigilance hors du commun et un niveau de compétences techniques croissant sur une multitude d'applications et de systèmes d'exploitation. Qui peut vraiment absorber tout ça ? Les conséquences ne sont pas négligeables : mauvaises habitudes, idées reçues, influences sociétales. Toutes ces choses qui se trouvent à l'opposé de ce que doit être un environnement professionnel et de toutes les sensibilisations que nous nous évertuons à dispenser depuis des années.

L'« objetisation » laisse à penser que les anciens problèmes n'existent plus, que les nouveaux comportements sont modernes, que la simplicité l'emporte sur le respect de procédures. Comme l'hameçonnage (ou *phishing*) nous l'a déjà montré, un peu d'astuce permet d'obtenir une bonne pêche. Avec ce nouveau terrain de jeu il faut s'attendre à de nouvelles attaques techniques ou sociales. Il est fort douteux que nos vieux mots de passe sachent y faire face. (Voir aussi [3]).

3 Où est-ce que ça fuit ?

Bien sûr, il n'est pas question ici d'être exhaustif puisque nous sommes dans un domaine où l'imagination est reine. Quelques exemples suffisent à montrer la fragilité de la sécurité lorsque les identifiants deviennent volatiles. Cela va d'exemples simples qui relèvent souvent du non respect de bonnes pratiques à des exemples plus sophistiqués et inquiétants.

3.1 Par les navigateurs

Les fonctions d'enregistrement des mots de passe dans les navigateurs devraient être protégées par un mot de passe principal qui en assure le chiffrement. Malheureusement on peut constater que c'est loin d'être le cas, surtout sur du matériel personnel, voire familial. Sur des mobiles sur lesquels, tout aussi souvent, il n'y a pas de code de verrouillage, avoir l'appareil en main donne ainsi un accès direct à une grande quantité de mots de passe en clair. Ce n'est guère mieux sur des machines « classiques » et même professionnelles. Par exemple, sous Windows, avec Firefox, il suffit qu'un virus exporte quelque part les fichiers *key3.db* et *signon.sqlite* du profil de l'utilisateur. Sans mot de passe principal, le pirate aura juste à les copier dans le profil de son propre navigateur pour lire tous les mots de passe.

3.2 Par les clients de messagerie

Prenons maintenant comme exemple le client mail standard d'Android. Pour se connecter sur un compte IMAP cette application utilise le gestionnaire de compte d'Android. Il s'agit d'une base Sqlite gérée par le système qui peut-être interrogée comme suit, à la condition d'être *root* :

```
cd /data/system/users/0

sqlite3 accounts.db

select * from accounts :

dupont@labo.cnrs.fr | com.android.email | motdepasseenclair
```

On constate que le mot de passe est en clair et récupérer un appareil non chiffré permettrait d'y accéder par un vidage de la mémoire flash. D'autres clients mail, n'utilisent pas le gestionnaire de comptes. Cela signifie que le mot de passe est enregistré dans l'espace de l'application, probablement pas chiffré.

De toute façon, en clair ou pas, il y a d'autres manières de récupérer des mots de passe, sans vraiment pirater le système d'exploitation ou une application, à partir du moment où le propriétaire n'applique pas la protection minimum que procure le code de verrouillage d'écran. Dans cette situation très fréquente, un attaquant ayant accès à l'appareil peut configurer l'outil de messagerie sans problème. Dans le paramétrage, sans toucher au champ du mot de passe, il lui suffira de modifier le serveur de mail en le faisant pointer sur une machine sous son contrôle et ainsi très facilement intercepter le mot de passe. Cette méthode a été testée avec succès avec des clients de messagerie sous Android et iOS, sans que cela ne nécessite une grande expertise sur ces systèmes.

Combien d'applications gèrent correctement les mots de passe que nous enregistrons, y compris sur des machines classiques ? Par exemple, dans ses premières versions, le client de synchronisation d'Owncloud sous Windows ou Linux enregistrait le mot de passe en clair dans un fichier de l'espace utilisateur. Depuis, la situation a été corrigée, Owncloud utilise désormais Qtkeychain. Très bien, mais ces exemples démontrent toutefois à quel point la sécurité des mots de passe est dépendante d'une multitude d'applications dont le niveau de sécurité peut fluctuer dans le temps. Même une application de coffre-fort à mots de passe a été prise en flagrant délit de transmission vers Internet des mots de passe qu'elle contenait ![4]

3.3 Et pourquoi pas par Eduroam

La plupart du temps, la configuration pour se connecter à Eduroam est faite au moyen du login principal de l'utilisateur et de son mot de passe qui sont en général enregistrés sur la machine cliente. La bonne procédure, consiste à récupérer les fichiers de configuration sur le site Eduroam pour chaque type de système. Ainsi le certificat de l'autorité de certification du serveur Radius sera chargé et le dialogue d'authentification sécurisé. On est sûr qu'on parle avec le bon serveur. Malheureusement il est aussi possible de faire cette configuration manuellement et de façon erronée. Ce que font nombre de personnes. Ainsi, sur Android on peut configurer le SSID Eduroam comme suit :

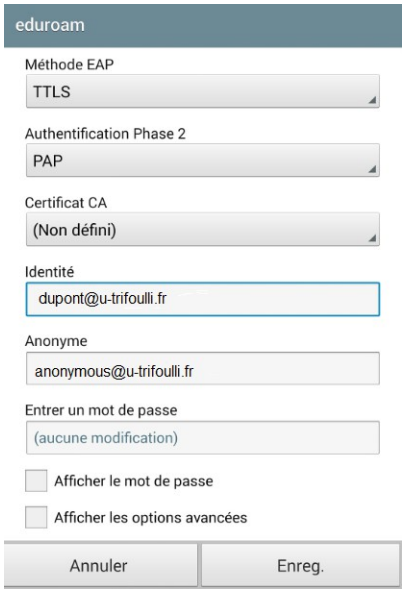

Figure 1 - Configuration Eduroam (erronée) sous Android

Ici le certificat CA n'est pas renseigné. L'authentification marche, l'utilisateur est content, mais la connexion n'est pas du tout sécurisée et le mot de passe pourra être intercepté. Le pirate a juste à diffuser un SSID Eduroam sur lequel quelqu'un viendra très vite se raccrocher. Ce faux SSID pointe sur son propre serveur Radius qui n'a besoin que d'une configuration très minimaliste. Il suffit de lancer Radius en mode Debug pour obtenir le mot de passe en clair dans le log. Dans ce type d'attaque il n'est pas utile d'avoir l'appareil en main et il est possible de piéger quelqu'un n'importe où, d'autant plus si la connexion Eduroam s'établit automatiquement quand le signal est détecté.

3.4 Merci Google

Google est formidable ! Il fournit à ses utilisateurs Gmail la possibilité de centraliser toutes leurs boites à lettres, quel que soit le fournisseur. C'est pratique, en se connectant sur une seule interface, on peut consulter tous ses mails. Pour cela, dans l'interface de paramétrage de Gmail il suffit de rentrer les coordonnées du serveur de mail et d'entrer son identifiant et son mot de passe qui, désormais trônera quelque part dans les serveurs de Google. A partir du moment où IMAP est directement ouvert sur Internet, qu'est-ce qui peut empêcher cela ? Nombre d'administrateurs de messagerie ont pu constater en lisant leurs logs que désormais Google fait parti de leurs utilisateurs.

3.5 Merci aussi Apple (et tous les autres)

La recherche des failles est devenue une vraie science très efficace. Même si à un moment un système peut être considéré comme sain, la mise à jour suivante peut tout remettre en cause. Régulièrement des chercheurs publient des études révélant des failles dans tel ou tel système. Il en est justement une intéressante puisqu'elle touche au cœur même de la gestion des mots de passe dans les systèmes iOS et OS X. En juin 2015, des chercheurs universitaires ont démontré que, grâce à une application malicieuse, il est possible de contourner le mécanisme de cloisonnement des applications (*sandboxing*) et aussi d'atteindre le Keychain et d'ainsi exposer des données et des mots de passe. 99 % des applications qu'ils ont testées au travers d'un échantillon statistique dans l'Apple store se sont avérées vulnérables. [5]

Ce cas est assez représentatif de l'écart entre le discours de l'éditeur et la réalité, même si bien sûr les vulnérabilités ne touchent pas que le milieu Apple. Le nombre de vulnérabilités annuellement découvertes montre bien qu'il est illusoire d'imaginer maintenir la sécurité d'un système d'information en sécurisant les clients, surtout s'ils sont un frigo ou une télé. Les failles XARA, Stagefright, XcodeGhost et bien d'autres

sont là pour nous le rappeler tous les jours.

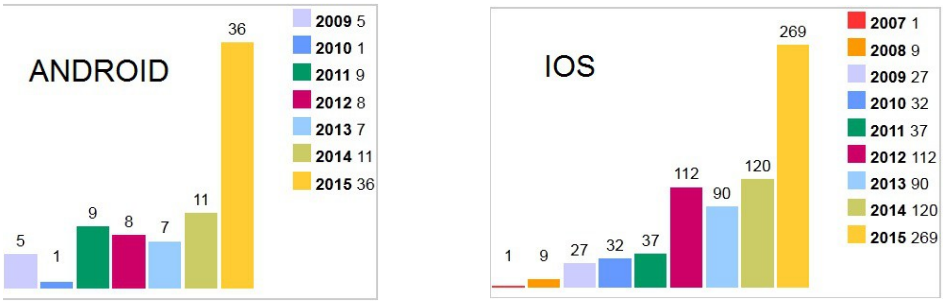

Figure 2 - Nombre de vulnérabilités annuelles dans les systèmes Android et iOS au 1er octobre 2015 (source http://www.cvedetails.com)

4 Mais que fait Google (et d'autres) ?

Contrairement à ce que l'on pourrait croire, sur mobile Android, dans des applications telles que Google Play ou Gmail, les identifiants de l'utilisateur ne sont pas enregistrés dans la machine. En effet, Google utilise le protocole *Oauth 2.0* dont le but est justement d'éviter ce type de problème.

Oauth implique deux parties. D'un côté une application cliente et de l'autre un serveur d'authentification. En résumé, le but du protocole consiste à déclarer une application cliente comme étant autorisée à agir au nom de l'utilisateur. Lors de la première connexion le client doit fournir ses identifiants Google. Le serveur renvoi alors un *access token*. C'est lui qui est enregistré. Lors des futures connexions c'est l'*access token* qui est présenté au serveur en guise d'authentification. Ce token ne peut pas être utilisé avec un navigateur pour se connecter sur une application Google et il est renouvelé automatiquement périodiquement. En cas de compromission l'impact est réduit.

D'autres acteurs d'Internet utilise aussi ce protocole. Par exemple Facebook ou Twitter pour leurs applications respectives.

Cette méthode ne pourrait pas résoudre tous nos problèmes. Google n'a pas besoin de savoir la nature du terminal et ne se préoccupe pas de qui peut accéder aux informations qu'il contient. *Oauth* a le mérite de montrer qu'il est possible grâce à un protocole d'éviter d'enregistrer les « vrais » mots de passe des utilisateurs, à la condition que les applications clientes le supportent.

5 Recréer la ligne de front

Après ce tour d'horizon la question est comment rétablir la situation ?

On peut imaginer une réponse purement disciplinaire et répressive en cas d'utilisation de matériels non autorisés. Mais il s'agira toujours d'une méthode à posteriori et probablement peu appliquée et donc peu efficace. En considérant le paragraphe 3.4 chacun pourrait aussi poser la question : « *Pourquoi mon employeur permet à Google de se connecter sur ses serveurs de mails et relever mon courrier* ? ». Face à la pression des usages de la société et aux capacités d'autonomie que procure la technologie, la sensibilisation, telle que nous la pratiquons, a probablement atteint ses limites. Il faudra certainement inventer une nouvelle forme de sensibilisation.

Durcir les terminaux clients n'aura pas d'efficacité tant qu'il sera possible d'utiliser n'importe quoi pour se connecter. La présence de quelques terminaux sécurisés ne contrecarre pas les effets d'une multitude invisible et ingérable d'appareils, de systèmes et de comportements qui positionneraient le niveau de sécurité du système d'information au plus bas.

Pour (re-)créer une ligne de front il sera certainement indispensable que le système d'information se protège globalement lui-même par des moyens techniques qui lui permettront de s'assurer que les utilisateurs ET leurs terminaux sont bien référencés ET qu'ils utilisent obligatoirement des protocoles qui conservent le secret des identifiants.

6 Expérimentation d'une méthode

Comme partout, la question de l'utilisation des mobiles s'est posée au CENBG. C'est à partir de l'analyse précédente qu'un dispositif expérimental a été mis en place pour intégrer les mobiles. Bien sûr, il n'était pas question de réduire le niveau de sécurité général. Deux principes de base pré-existaient et devaient toujours s'appliquer dans le cas des mobiles. D'une part, quel que soit le terminal classique ou mobile, personnel ou professionnel, être capable de répondre à trois questions : Qui se connecte, avec quoi et comment ? D'autre part, obliger le passage dans un mode sécurisé avant de pouvoir accéder aux services du SI depuis l'extérieur. Par l'addition de ces deux conditions, frigos, téléviseurs et autres engins restent à distance.

Ce mode sécurisé existait déjà grâce à l'utilisation d'un accès VPN (OpenVpn) qui, depuis Internet, permettait d'assurer une forte authentification, à double facteur, du couple utilisateur/machine. Mais ce dispositif n'était pas suffisant pour contrecarrer la diffusion des mots de passe dans tous les logiciels. Il fallait donc utiliser une méthode, ressemblant à *Oauth*, utilisable quelle que soit l'application.

Cette expérimentation devait prendre en compte iOS et Android, se faire sans coût supplémentaire et avec une réutilisation des moyens techniques déjà existants, à savoir : OpenVpn, Radius et une Infrastructure de Gestion de Certificats (IGC).

6.1 Connexion par VPN (Android et iOS)

Déjà utilisé pour les machines « classiques », déployer OpenVpn sur les systèmes Android et iOS ne fut pas un problème puisqu'ils disposent tous deux de l'application cliente. La configuration et l'utilisation sont strictement les mêmes dans tous les environnements.

L'authentification sur le serveur VPN est réalisée au moyen de deux facteurs. Le premier est un certificat, émis par une IGC locale, importé dans le magasin de certificats du système d'où il n'est pas exportable et non pas dans une application. Il est spécifique à la machine et il est installé par le service informatique qui s'assure ainsi que le fichier PKCS12 intermédiaire est détruit. La présence de ce certificat contribue aussi à la protection générale de l'appareil, car il oblige l'utilisateur à positionner un code de verrouillage de l'écran.

Le second facteur est un secret, requis par OpenVpn et indépendant du certificat, donc un mot de passe. Pour ne pas retomber dans le problème initial des mots de passe il doit être simple, non enregistrable et être la seule chose que doit taper l'utilisateur pour se connecter jusqu'à son application.

Avec OpenVpn il est possible d'interdire l'enregistrement de ce mot de passe, mais cette option pourrait être contournée. L'idéal serait que ce mot de passe doivent changer à chaque connexion, mais cela nécessite une infrastructure spécifique et probablement des contraintes peu compatibles avec l'ergonomie des mobiles. En réalité, il suffit que le format du mot de passe rende l'enregistrement plus fastidieux et inutile. Par exemple s'il est composé d'une partie fixe, secrète, et d'une partie variable aléatoire qui doit changer à chaque connexion. Un code pin de 4 chiffres suivi de deux caractères aléatoires que l'utilisateur doit modifier d'une fois sur l'autre constitue un mot de passe facile à retenir et à taper. Du côté serveur, avant de valider la partie secrète, il suffit de s'assurer que la partie variable a changé depuis la dernière connexion.

6.1.1 Fonctionnement du processus d'authentification à deux facteurs

Le schéma ci-dessous montre le fonctionnement du processus d'authentification d'un client par OpenVpn et Radius :

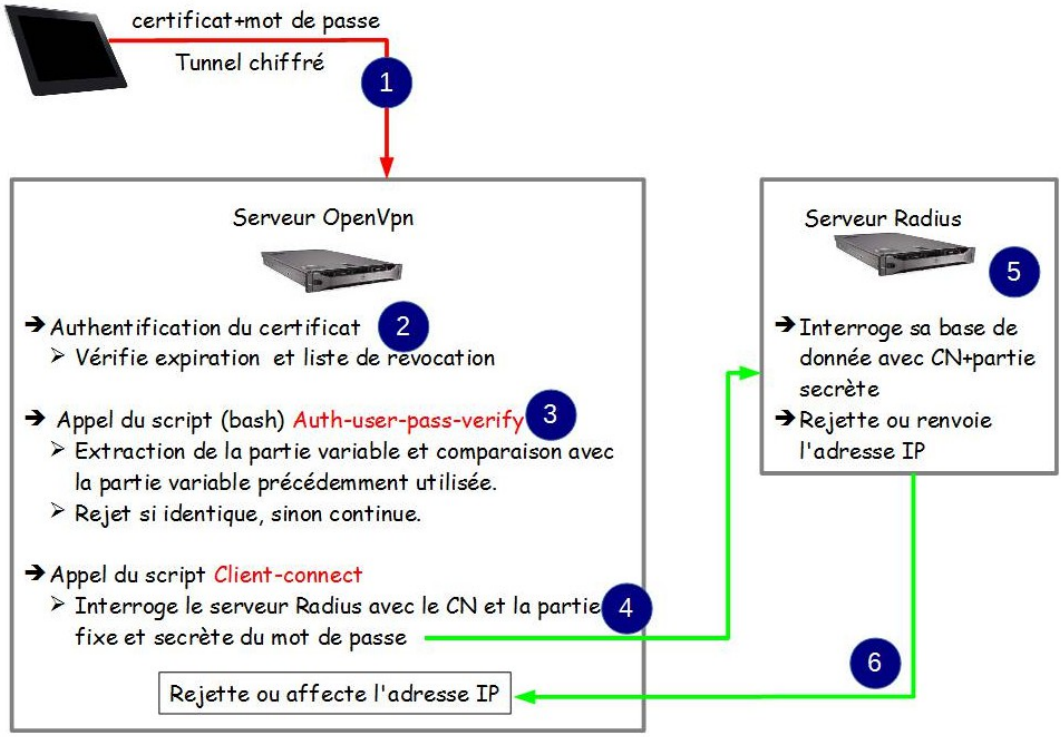

Figure 3 - Processus d'authentification VPN à double facteur

Le processus d'authentification est le suivant :

1. le client présente son certificat et son mot de passe (pin code+ caractères aléatoires)
2. le serveur OpenVpn authentifie ce certificat ou rejette la connexion
3. puis il extrait la partie secrète du mot de passe
4. et interroge le serveur Radius en lui présentant le CN du certificat et la partie secrète.
5. le serveur Radius valide (ou pas) la correspondance entre le CN et le mot de passe
6. et renvoie au serveur OpenVpn l'adresse IP à allouer au client ou bien un refus de connexion

6.1.2 Configuration Radius

L'autorisation associée à un certificat est enregistrée dans la base du serveur Radius comme suit :

```
common-name-du-certificat  Crypt-Password:= b9127MbUQKty,
expiration:= "11 Dec 2015 00:00:00"
       framed-ip-address = 192.168.30.12
```

Dans cette ligne il est dit que le certificat possédant ce Common Name (nom de la machine) doit être accompagné du code pin dont la forme chiffrée est dans ***Crypt-Password***. Il s'agit ici uniquement de la partie secrète. En cas de succès, *FRAMED-IP-ADDRESS* renvoi l'adresse IP au serveur VPN.

6.1.3 Bénéfices

Grâce à ce dispositif, l'administrateur est en mesure d'avoir une vue complète du parc de machines autorisées à interagir avec le système d'information et l'utilisateur n'a pas à se soucier du certificat qui est complètement transparent. Il reçoit des alertes lorsqu'il approche des dates d'expiration et l'administrateur dispose d'un bilan permanent lui permettant, par exemple, d'anticiper le blocage d'un utilisateur distrait.

Pour que la connexion soit authentifiée il faut posséder un certificat et le code pin secret. Pour réaliser une attaque force brute il faut donc déjà posséder ce certificat. De plus, un mécanisme permet de bloquer les connexions après quelques tentatives infructueuses. Le serveur VPN agit comme un firewall qui filtre les protocoles utilisables par le client.

En cas d'incident sur un terminal, il est possible de révoquer le certificat et/ou l'autorisation dans Radius. Seul l'appareil concerné est bloqué laissant à l'utilisateur la possibilité d'utiliser ses autres matériels non compromis.

Une trentaine de lignes de code en Bash a suffit pour implémenter la gestion du mot de passe variable (auth-user-pass-verify).

6.1.4 Configuration du client VPN

Deux étapes sont nécessaires :

- Installer l'application cliente ***Openvpn connect*** sur l'appareil. Cette opération peut être réalisée par l'utilisateur.

- Installer le certificat et la configuration OpenVpn et enregistrer le code pin utilisateur dans Radius (réalisé par l'administrateur)

Le certificat et la configuration OpenVpn sont téléchargés depuis un serveur web (interne) dédié. Une fois ces étapes réalisées, l'utilisateur n'a plus qu'à se connecter en fournissant son code pin.

6.2 Configuration d'une application

Une connexion VPN est une chose intéressante pour sécuriser la liaison mais cela ne permet pas en soi d'empêcher le stockage des mots de passe pour les applications utilisées ensuite. La méthode ici va consister à enregistrer un token, un peu comme dans *Oauth*. Ce token ne pourra être utilisé qu'après établissement de la connexion VPN, pour une application donnée et uniquement pour ce terminal. Nous prendrons ici l'exemple d'une application de mail et supposerons que IMAP n'est pas accessible autrement que via la connexion VPN, sans quoi tout cela ne servirait à rien.

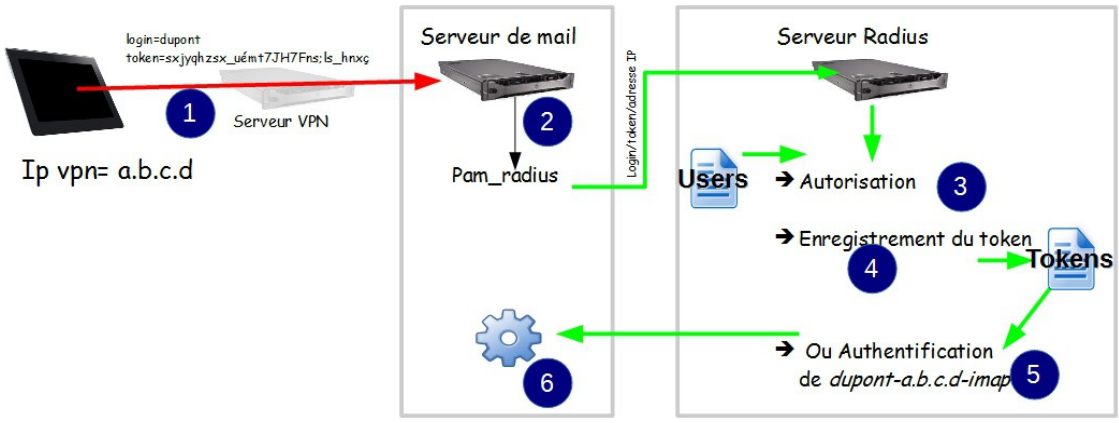

Figure 4 - Méthode d'authentification applicative par token

1. Le client ouvre sa connexion VPN qui lui attribue une adresse IP spécifique et authentifiée. L'application mail est configurée avec le login de l'utilisateur et, en guise de mot de passe, une chaîne de caractères tapés aléatoirement, qui devient le token. La demande d'authentification est envoyée au serveur de mail via la liaison VPN.
2. Le serveur de mail entame une authentification PAM au travers du module *pam_radius* qui pointe sur le serveur Radius. PAM lui transmet le login, l'adresse IP et le token.
3. Le serveur Radius vérifie que le couple login/adresse IP est bien autorisé à s'authentifier ainsi.
4. S'il s'agit de la première tentative le token reçu est enregistré dans un fichier de tokens.
5. Pour les tentatives suivantes, le token reçu est vérifié dans le fichier des tokens.
6. Le serveur de mail accepte ou refuse la connexion suivant la réponse du serveur Radius.

Du côté client le token est enregistré dans l'application et l'utilisateur n'a plus à s'en préoccuper. Pour se connecter sur sa messagerie il devra simplement lancer sa connexion VPN.

6.2.1 Bénéfices

Le vrai mot de passe de l'utilisateur n'est jamais enregistré. Chaque application sur chaque appareil utilise un token spécifique. S'il est compromis, à lui seul il ne peut pas être utilisé puisque qu'il faut aussi acquérir le certificat et le code bin. L'implémentation de la gestion du token dans Radius correspond à un code de moins de 100 lignes en Bash.

7 Conclusion

Tolérer aujourd'hui cette dissémination des identifiants revient à dire qu'enregistrer son mot de passe dans tout un tas d'appareils est moins dangereux que de le scotcher sous son clavier !

L'Internet des Objets change l'ambiance numérique et ce qui nous paraissait saugrenu et éloigné de nos préoccupations pourrait se retrouver au beau milieu de nos moyens informatiques. Dans les années 80 nous n'imaginions pas ce que nous ferions des micro-ordinateurs d'alors. Nous avons actuellement les mêmes limitations imaginatives avec les nouvelles technologies. Pourtant, nous devrons savoir les intégrer tout en repoussant les débordements risqués.

Pour cela il faudra mettre en œuvre des méthodes qui garantissent les limites du système d'information. L'expérience du CENBG montre qu'il est possible de considérablement réduire le risque sans pour autant compliquer l'usage et avec des moyens déjà connus.

Bibliographie

[1] Proofpoint Uncovers Internet of Things (IoT) Cyberattack - http://investors.proofpoint.com/releasedetail.cfm?ReleaseID=819799

[2] Hacking DefCon 23's IoT Village Samsung fridge - https://www.pentestpartners.com/blog/hacking-defcon-23s-iot-village-samsung-fridge/

[3] Etude 2015 Symnantec/Norton sur la mobilité - http://www.youscribe.com/catalogue/tous/etude-de-symantec-norton-sur-la-mobilite-2015-2583578

[4] Unencrypted storage of confidential information in Keeper Password & Data Vault v5.3 for iOS – Avril 2013 - http://blog.fox-it.com/2013/04/05/security-advisory-unencrypted-storage-of-confidential-information-in-keeper-password-data-vault-v5-3-for-ios/

[5] Luyi Xing, Xiaolong Bai, Tongxin Li, XiaoFeng Wang, Kai Chen, Xiaojing Liao : Unauthorized Cross-App Resource Access on MAC OS X and iOS https://drive.google.com/file/d/0BxxXk1d3yyuZOFlsdkNMSGswSGs/view?pli=1