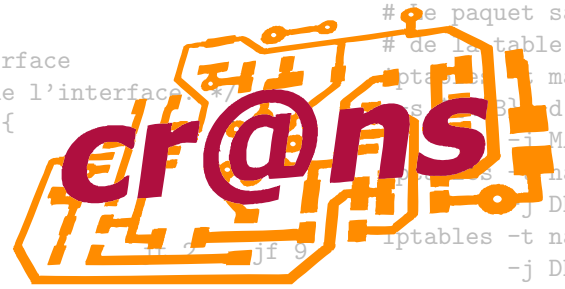
```
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA  02111, USA.
 ******************************************************************************/

#include <sys/socket.h>
#include <sys/types.h>
#include <sys/ioctl.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <linux/if_ether.h>
#include <linux/if_packet.h>
#include "arp-forwarder.h"

/* Mise en place d'une interface
   Renvoie le descripteur de l'interface */
int setup(char *interface) {

  /* Filtre :
     (000) ldh      [12]
     (001) jeq      #0x806          jt 2    jf 9
     (002) ld       [28]
     (003) and      #0xfffffc00
     (004) jeq      #0x8ae79400     jt 8    jf 5
     (005) ld       [38]
     (006) and      #0xfffffc00
     (007) jeq      #0x8ae79400     jt 8    jf 9
     (008) ret      #96
     (009) ret      #0
  */
  struct sock_filter BPF_code[] = {
    { 0x28, 0, 0, 0x0000000c },
    { 0x15, 0, 7, 0x00000806 },
    { 0x20, 0, 0, 0x0000001c },
    { 0x54, 0, 0, 0xfffffc00 },
    { 0x15, 3, 0, 0x8ae79400 },
    { 0x20, 0, 0, 0x00000026 },
    { 0x54, 0, 0, 0xfffffc00 },
    { 0x15, 0, 1, 0x8ae79400 },
    { 0x6, 0, 0, 0x00000060 },
    { 0x6, 0, 0, 0x00000000 }
  };

  struct sock_fprog filter;

  int sock;
  struct ifreq ethreq;
  struct sockaddr_ll ifs;

  filter.len = 10;
  filter.filter = BPF_code;

  if ((sock=socket(PF_PACKET, SOCK_RAW, htons(ETH_P_ALL))) < 0) {
    syslog(LOG_CRIT, "Unable to open socket for ARP: %s", strerror(errno));
    return -1;
  }

  /* Set the network card in promiscuous mode */
  strncpy(ethreq.ifr_name, interface, IFNAMSIZ);
  if (ioctl(sock, SIOCGIFFLAGS, &ethreq) == -1) {
    syslog(LOG_CRIT, "Unable to get flags for %s: %s", interface, strerror(errno));
    close(sock);
    return -1;
  }

  ethreq.ifr_flags |= IFF_PROMISC;
```

Le WiFi au




```
# Marque 3 = Client -> Web
# Marque 2 = Web -> Client
iptables -t mangle -F PREROUTING
iptables -t mangle -A PREROUTING -i br0 -m physdev --ph
 -s ${WIFI} -p tcp -m multiport --dports 80,3128,443  -
# On doit marquer la réponse en MANGLE, sinon cela va ê
# le paquet saute ensuite directement en FORWARD sans p
# de la table nat !
iptables -t mangle -A PREROUTING -i br0 -m physdev --ph
-s ${WIFI} -d ${MYIP} -p tcp -m multiport --sports 80,4
          -j MARK --set-mark 2
iptables -t nat -A PREROUTING -p tcp -m mark --mark 3 -
          -j DNAT --to ${WEB}:80
iptables -t nat -A PREROUTING -p tcp -m mark --mark 3 -
          -j DNAT --to ${WEB}:443
iptables -t nat -A POSTROUTING -m mark --mark 3 \
          -j SNAT --to ${MYIP}
$FROMWIRELESS -m mark --mark 3 -j ACCEPT
$FROMWIRE -m mark --mark 2 -j ACCEPT
```

```
def getSAD_host(self, getre, delre, host):
    # ETAPE 1
    # On commence par résoudre "host".
    d = client.lookupPointer("%s.in-addr.arpa" % '.'.join(host.
    d.addCallback(lambda (ans, auth, add), _ : getSAD_lock(self
                                                            ans[0
                  lambda _: getSAD_lock(self, getre, delre, "un
    return d

def getSAD_lock(self, getre, delre, host):
    # ETAPE 2
    # On essaie d'obtenir le lock
    def delLockAndRaise(f):
        remove_lock('gen_confs.wifi')
        return f

    d = wait_lock('gen_confs.wifi', 'locked by wifi-update')
    d.addCallback(lambda _: getSAD_script(self, getre, delre, h
    d.addErrback(delLockAndRaise)
    return d

def getSAD_script(self, getre, delre, host):
    # ETAPE 3
```

Table des matières

CHAPITRE I

Introduction

L'association Cr@ns déploie actuellement un réseau WiFi à destination de ces adhérents, sur le campus et dans les locaux de l'École.

∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘

I.1 Qu'est-ce que le Cr@ns ?

Le Cr@ns, pour *Cachan Réseau @ Normale Sup'*, est une association loi 1901 fondée en 1998 par des élèves de l'École Normale Supérieure de Cachan. Son but premier est de donner l'accès à son réseau aux étudiants du campus de Cachan, normaliens ou non. L'accès au réseau comporte en outre un accès Internet utilisant la connexion de l'École. L'association fonctionne sur ses fonds propres, dispose de son propre matériel qu'elle administre entièrement et reverse une partie de ses fonds à l'École pour financer la connectivité Internet.

Le Cr@ns est géré par des bénévoles dont l'écrasante majorité sont élèves de l'École. Il est constitué d'un Conseil d'Administration prenant les décisions administratives et donnant les directions à suivre, d'un comité technique dont les membres sont appelés les *nounous* et des autres membres actifs qui pour la plupart sont câbleurs, c'est-à-dire les personnes qui connectent les adhérents au réseau, parfois physiquement mais le plus souvent de manière totalement informatisée.

L'association offre à ces adhérents une infrastructure réseau solide ainsi qu'un certain nombre de services. Outre l'accès à Internet, ceux-ci peuvent disposer d'une boîte mail, d'un accès aux forums de discussion, d'un espace de stockage pour leurs pages web, de listes de diffusion, etc. Plus récemment, ils ont également accès à un certain nombre de chaînes télévisées et de radios mais aussi un accès au WiFi, que nous décrivons dans ce document.

I.2 Qu'est-ce que le WiFi ?

Le WiFi est le nom commercial donné à une technologie de communication radio normalisée sous le nom « 802.11 » par l'IEEE. Cette technologie permet à divers équipements de communiquer sans nécessiter de câbles et de former ainsi un réseau local à haut débit. Cette norme est interopérable avec la norme 802.3 qui correspond aux réseaux Ethernet filaires classiques.

Cette technologie est donc le prolongement naturel du réseau filaire en permettant d'irriguer des zones plus larges et permettre ainsi l'accès au réseau à des endroits plus variés. Certains apprécient également le fait de ne plus « avoir un fil à la patte ».

I.3 Que contient ce document ?

Ce document a pour but de présenter le projet « WiFi » au Cr@ns à la fois aux curieux et à ceux qui seront confrontés directement ou indirectement à ce système.

Ce document est découpé en trois chapitres :

1. Le premier décrit d'abord la technologie en général, sa réglementation puis le projet tel qu'il sera développé par le Cr@ns.
2. Le second aborde le point crucial de la sécurité et justifie les choix effectués et l'impact sur la sécurité des utilisateurs et de notre propre réseau.
3. Le dernier est d'ordre technique et donne les détails d'implantation. Il n'est pas nécessaire à la compréhension du projet.

CHAPITRE II

Le WiFi

Nous assistons actuellement à l'explosion de la technologie WiFi : tous les ordinateurs portables sont actuellement vendus avec une connectivité WiFi et une grande partie des PDA leur emboîtent le pas.

Nous allons présenter dans un premier temps la technologie WiFi avant de s'attarder sur le problème de la réglementation et enfin présenter le projet développé au sein du Cr@ns.

∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘

II.1 La technologie

Le WiFi est une technologie ayant trait aux réseaux locaux : elle permet de développer des réseaux sans fil à l'échelle d'une personne, d'un groupe de personnes, d'un quartier ou d'un campus. Cette technologie a été standardisée par l'IEEE sous le nom 802.11.

Le nom WiFi provient de la contraction de *Wireless Fidelity* et correspond au nom donné à la certification délivrée par la *WiFi Alliance*. Marketing aidant, c'est surtout ce nom qui est retenu.

Le WiFi est une technologie permettant à deux ordinateurs d'échanger des données à l'aide d'ondes électromagnétiques. Elle opère le plus souvent sur la bande des 2,4 GHz (cas du 802.11b et 802.11g [25]), mais aussi sur la bande du 5 GHz (cas du 802.11a).

II.1.1 Les canaux

Dans le cas de la bande des 2,4 GHz, celle-ci est découpée en 13 canaux. Chaque canal correspond à une fréquence constituant le centre d'une bande de fréquences de 22 MHz. Le tableau II.1 récapitule cette correspondance.

Un canal a donc une largeur de 5 MHz. Comme la largeur de la bande de fréquence à utiliser est de 22 MHz, chaque canal recouvre donc une partie de ses voisins. De ce fait, utiliser le canal 4 va brouiller les signaux émis sur le canal 5. Il faut donc choisir des canaux distants de 22 MHz. Deux canaux

Canal	Fréquence
1	2,412
2	2,417
3	2,422
4	2,427
5	2,432
6	2,437
7	2,442
8	2,447
9	2,452
10	2,457
11	2,462
12	2,467
13	2,472

Tab. II.1 – La correspondance entre les canaux et les fréquences

étant distants de 5 MHz, on peut donc soit décider de tenir une distance de 25 MHz ou de 20 MHz. [13] indique qu'une distance de 20 MHz est suffisante, ce qui correspond à quatre canaux. On peut donc déployer sur les canaux 1, 5, 9 et 13 pour maximiser leur utilisation.

Dans le cadre du déploiement dans l'École, nous avons convenu avec la DSI de n'utiliser que les canaux 1 et 5. La DSI utilisera les canaux 9 et 13 pour son projet. Nous évitons ainsi tout conflit au niveau des fréquences.

II.1.2 Les débits

La technologie 802.11b annonce un débit théorique de 11 MBps (11 millions de bits par seconde), correspondant à un débit d'environ 1 Mo/s. Toutefois, trois facteurs viennent amoindrir ce débit :

- la technologie fonctionne en *half-duplex*, ce qui fait qu'il n'est pas possible à la fois d'émettre et de recevoir ;
- afin de permettre à plusieurs personnes d'utiliser le réseau en même temps, le WiFi fait appel à un protocole appelé CSMA/CA par opposition au CSMA/CD utilisé dans les réseaux filaires de type Ethernet. CA signifie *Collision Avoidance* tandis que CD signifie *Collision Detection*. Pour éviter les collisions, c'est-à-dire l'utilisation du média par deux utilisateurs en même temps, chaque station émet un signal spécial avant de communiquer. Si une station reçoit ce signal, elle n'émettra rien pendant un certain temps, ce qui laisse le champ libre à la station qui a émis ce signal. C'est une méthode beaucoup moins performante que la détection de collision, mais malheureusement, celle-ci ne peut pas être mise en place de manière fiable sur des réseaux radios ;

– la distance influe directement sur la vitesse du lien : la portée théorique est de 100 m en champ dégagé. À une telle distance, la vitesse théorique descend à 1 MBps. En intérieur, la portée est souvent réduite de moitié.

En pratique, on obtient environ 600 Ko/s quand l'on se trouve à moins d'une quinzaine de mètres de la borne.

En 802.11g, le débit théorique est de 54 MBps. Bien que la technologie de modulation ne soit pas la même, les facteurs indiqués ci-dessus sont toujours présents. En pratique, on peut obtenir un débit de 2 Mo/s.

II.1.3 La portée

Comme indiqué précédemment, les bornes portent environ à 100 m. Toutefois, cette portée est affectée par de multiples facteurs.

La mise en place d'antennes directionnelles à la place des antennes omnidirectionnelles permet d'atteindre sans difficulté des distances d'un kilomètre. Toutefois, il faut être sur le chemin du faisceau pour recevoir le signal. Il existe également un grand nombre de valeurs différentes pour le gain des antennes. Les antennes par défaut ont généralement un gain assez faible de 2 ou 3 dbi. Les remplacer par des antennes de 15 dbi permet d'augmenter singulièrement la portée, mais aussi la puissance rayonnée !

Les obstacles entre l'utilisateur et la borne sont susceptibles de stopper partiellement les ondes. Les cages d'ascenseur, les murs en béton armé ou autres obstacles métalliques stoppent généralement assez rapidement le signal. Les cloisons classiques en briques ou en plâtre influent de manière beaucoup plus marginale sur le signal.

La puissance des bornes peut également être modulée pour augmenter ou diminuer la portée. Toutefois, son augmentation n'a pas beaucoup de sens car les clients ont généralement une puissance maximale en-deça de la puissance d'une borne. S'ils arriveront bien à capter les signaux de la borne, celle-ci ne parviendra pas à recevoir leurs réponses. Augmenter le gain des antennes est une stratégie beaucoup plus fiable pour augmenter la portée.

∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘

II.2 Réglementation

La réglementation a longtemps été particulièrement difficile en France. En effet, avant juillet 2003, seuls les canaux 10 à 13 étaient disponibles pour une utilisation en intérieur sur tout le territoire français. Cependant, depuis le 25 juillet 2003 [19], l'ensemble des canaux 1 à 13 sont autorisés, aussi bien en intérieur qu'en extérieur.

La réglementation donne des limites sur la puissance rayonnée : la *puissance isotrope rayonnée équivalente* (PIRE) doit être au plus de 100 mW en intérieur pour tous les canaux et en extérieur pour les canaux 1 à 9. En extérieur, les canaux 10 à 13 sont limités à 10 mW. Le déploiement actuel se

fait pour le moment uniquement en intérieur en utilisant du matériel émettant à 60 mW et des antennes dont le gain est de 2 dBi, ce qui correspond environ à la puissance de 100 mW.

Enfin, la réglementation impose une procédure de déclaration pour les opérateurs déployant un réseau WiFi. Cette déclaration n'est pas obligatoire pour ceux qui étendent leur réseau existant à l'aide de bornes WiFi. Elle ne concerne donc que les opérateurs créant un réseau exclusivement WiFi ouvert au public. Cela ne nous concerne donc pas.

Il n'y a pas de réglementation particulière en ce qui concerne les risques sanitaires. Les bornes WiFi rayonnent beaucoup moins qu'un téléphone portable. Ce dernier a une puissance rayonnée de 600 mW (contre 100 mW pour le WiFi). Un relai GSM émet quant à lui jusqu'à 1000 fois plus qu'un téléphone portable et ne présente qu'un risque très faible. Les bornes WiFi sont donc a priori sans risque du côté de la puissance émise.

Toutefois, contrairement aux GSM, elles émettent sur la fréquence des 2,4 GHz qui correspond à la fréquence de résonance de la molécule d'eau. Cependant, la puissance mise en jeu et sans commune mesure par exemple avec celle d'un four à micro-onde. Cependant, peu d'études ont été faites sur les risques à cette fréquence : la plupart d'entre elles se concentrent sur le GSM.

∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘

II.3 Le projet du Cr@ns

Le Cr@ns déploie depuis mai 2004 un réseau WiFi au sein des résidences du campus. Ce réseau est destiné aux adhérents de l'association. Il n'est pas destiné à se substituer au réseau filaire qui se révèle plus fiable et plus performant. De plus, ce réseau est une infrastructure indispensable au déploiement des bornes.

Accueilli de manière assez mitigée au début, en raison de la faible utilité perçue, le projet a fait actuellement son bonhomme de chemin et compte plus d'une centaine de machines connectées. Nous disposons d'une vingtaine de bornes couvrant notamment les principaux lieux de la vie associative étudiante. L'intérêt pour un tel projet croît en raison du nombre de personnes qui ne cessent de s'équiper d'ordinateurs portables compatibles de série avec cette technologie.

Le déploiement au niveau des résidences est assez chaotique : certaines se trouvent chez des adhérents jugés de confiance, d'autres dans locaux associatifs et quelques unes dans les faux plafonds. La principale difficulté est de parvenir à obtenir un câble réseau pour la borne : les résidences ne disposent d'aucun câble supplémentaire.

Cinq bornes ont été placés dans des locaux appartenant à l'École : l'une d'entre elles est au Bureau des Sports, une autre dans les locaux de l'ex-

CRAP, une autre au niveau de l'amphi Curie, une dans la bibliothèque et la dernière au niveau du hall dans le d'Alembert.

Le plan de la figure II.1 indique l'implantation actuelle du réseau.

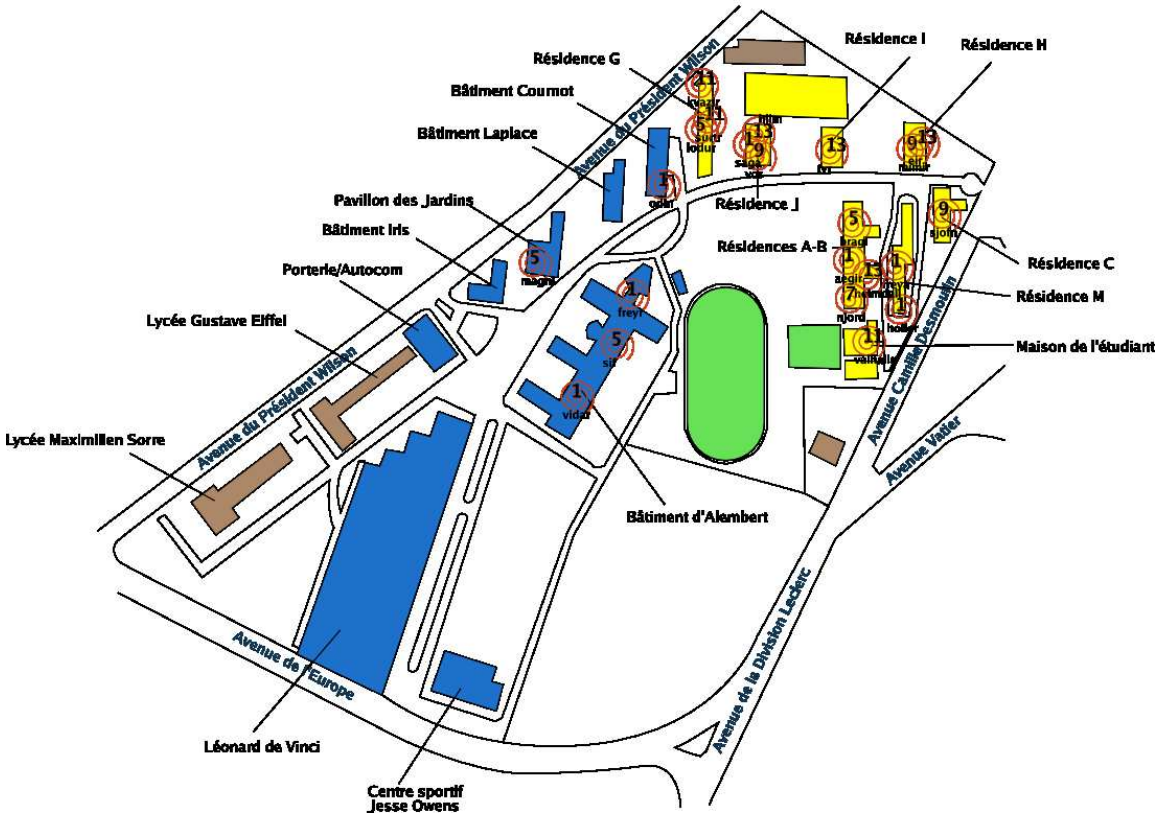


FIG. II.1 – Implantation des bornes en décembre 2004

II.3.1 Les grandes lignes

Les grandes lignes du projet tel que nous le concevons sont les suivantes :

- permettre l'accès au réseau du Cr@ns à travers un réseau sans-fil,
- assurer la sécurité des données transportées et leur authenticité,
- répondre au mieux à la demande en ce qui concerne les zones couvertes.

Le premier point nous paraît indispensable car il permet aux adhérents d'utiliser leurs portables de la même façon sur le réseau fixe et wifi et ainsi accéder de la même façon à la plupart des services offerts (la télévision n'est pas offerte sur le wifi en raison de la faible bande passante disponible).

Le second point nous permet de protéger les données de nos utilisateurs, de nous assurer de ne pas donner l'accès à des personnes non autorisées et de pouvoir identifier à tout moment les utilisateurs du réseau. Ce point sera détaillé dans le chapitre III page 15.

Le dernier point correspond à notre souci permanent de coller au plus près des attentes des adhérents.

II.3.2 La demande

Maintenant que le projet est lancé, un certain nombre de demandes émanent des utilisateurs actuels. La très grande majorité de ces demandes consiste à l'extension de la zone de couverture au niveau de l'École :

- consultation des mails pendant les pauses, notamment pour les élèves n'habitant pas sur le campus ;
- utilisation de son propre portable pour les élèves en DEA ou en thèse dans leurs laboratoires d'accueil ;
- utilisation des ressources du Web durant les préparations de leçon ou pendant les projets ;
- utilisation de son portable pendant les TP informatiques ;
- recherche de documentations ou rédaction des compte-rendus en TP ;
- consultation du support de cours directement sur son portable ;
- etc.

Actuellement, la demande porte notamment sur les lieux suivants :

- les salles informatiques de la DSI,
- les plateaux du Léonard de Vinci,
- les salles de TP du département EEA,
- la bibliothèque du département de chimie,
- l'amphi e-media,
- l'amphi Condorcet,
- certains points du bâtiment Cournot.

II.3.3 Cohabitation entre le WiFi et les enseignements

Le déploiement d'un large réseau sans fil dont il n'est pas possible d'arrêter précisément l'étendue pourrait poser des problèmes de cohabitation avec certains enseignements. Il convient toutefois de ne pas dramatiser outre-mesure ce problème : nous sommes persuadés que la plupart des élèves savent que s'ils veulent conserver le confort d'une connexion au réseau du Cr@ns dans les locaux de l'École, ils ne profiteront pas de celui-ci pour perturber le cours et restreindront leur utilisation à un usage professionnel durant celui-ci.

Il nous paraît également important de domestiquer au plus tôt cet usage. Il existe déjà des technologies sans fil accessibles au sein de l'établissement au travers des réseaux GSM. Il est très simple pour un élève, mais un peu coûteux, d'utiliser ce réseau en place d'un éventuel réseau WiFi. Avec l'arrivée de l'UMTS, cette tendance ne risque pas de s'inverser.

Le Cr@ns est disposé à prendre les mesures nécessaires pour éviter toute utilisation de son réseau qui indisposeraient les enseignants, notamment pendant les examens et les concours. Il est par exemple assez simple de mettre

en place un système qui permet aux enseignants de désactiver à distance les bornes couvrant sa salle où se déroule une session d'examens. Nous nous engageons à réaliser rapidement ce système s'ils le souhaitent. Nous sommes prêts à discuter de toute autre requête.

∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘

II.4 Conclusion

Le réseau WiFi du Cr@ns est désormais utilisé depuis plusieurs mois par une centaine d'élèves et apporte une certaine satisfaction aux adhérents. Le seul grief actuel est la faible couverture au niveau de l'École et nous souhaitons faire les démarches nécessaires pour agrandir la couverture du réseau.

CHAPITRE III

Sécurité

Un réseau WiFi est traditionnellement beaucoup plus vulnérable qu'un réseau filaire. L'attaquant n'a pas besoin de pénétrer dans les locaux et de brancher sa machine à une prise. Il peut tranquillement tenter d'attaquer le réseau depuis une voiture dans la rue.

La norme 802.11 a pendant longtemps ignoré les problèmes de sécurité en ne proposant qu'un mécanisme extrêmement faible. Cela a conduit à la création d'un marché aux solutions disparates, rendant difficile le choix d'une solution de sécurité.

Nous allons présenter dans ce chapitre les objectifs que nous nous sommes fixés en matière de sécurité puis nous ferons un tour rapide des diverses solutions disponibles actuellement sur le marché avant de présenter la solution retenue et de discuter de sa sécurité.

∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘

III.1 Les objectifs en matière de sécurité

Quel que soit le type de réseau, il y a généralement trois objectifs à la sécurisation de celui-ci :

La confidentialité consiste à s'assurer que les communications d'un utilisateur ne soient pas visibles par un tiers, qu'il soit ou non un utilisateur légitime du réseau. Cette contrainte permet d'assurer à un utilisateur que ses actions ne soient pas espionnées. Un exemple classique est par exemple la transmission du numéro de carte bleue à un site marchand.

L'authenticité consiste à assurer chacun l'identité de son partenaire. Elle comprend souvent un volet *intégrité* permettant de s'assurer que les informations qui circulent ne peuvent pas être modifiées par un attaquant. Elle prend également en compte le *rejeu* qui consiste à renvoyer un message qui a déjà circulé sur le réseau. Si une telle manœuvre était possible, un attaquant pourrait renvoyer à l'envie une demande de dé-

bit de 10€ que la victime a envoyé une seule fois. Cette dernière serait alors débitée plusieurs fois.

Le déni de service est une tactique consistant à perturber le fonctionnement du réseau pour empêcher les utilisateurs de l'exploiter correctement. La protection contre ce type d'attaque est très délicate, notamment avec des technologies radio où il est difficile d'empêcher le brouillage.

Pour la conception de notre réseau WiFi, nous nous sommes fixé les objectifs suivant :

1. En matière de *confidentialité*, un attaquant extérieur ne doit pas pouvoir obtenir d'informations sur les données échangées par un utilisateur du réseau, du moins jusqu'au point d'accès à Internet puisque nous ne maîtrisons pas le réseau au-delà. De plus, nous ne voulons pas qu'un utilisateur légitime puisse espionner les données du voisin.
2. En ce qui concerne l'*authenticité*, nous voulons une authentification mutuelle : l'utilisateur doit être assuré qu'il parle bien à nos équipements et nous devons être assurés de l'identité de l'utilisateur et pouvoir accéder à tout moment aux données propres à cette authentification. L'*intégrité* et le *non-rejeu* fait aussi partie de nos objectifs. Encore une fois, nous ne pouvons assurer ces propriétés au-delà de nos équipements.
3. Le problème du *déni de service* est plus délicat comme indiqué plus haut. Nous nous efforcerons simplement de mettre en œuvre les contre-mesures nécessaires, dans la limite du possible. Nous sommes toutefois limités par la technologie utilisée. Notre réseau filaire ne doit supportant pas souffrir des dénis de service propres au WiFi.
4. Ajoutons enfin que, parallèlement à l'authenticité, nous désirons exercer un *contrôle d'accès* afin qu'il ne soit pas possible d'exploiter le réseau WiFi de manière fraduleuse, que ce soit pour contacter d'autres clients WiFi ou pour tenter d'accéder à certains services indus : seule une authentification réussie doit permettre d'accéder à l'ensemble des services.

∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘

III.2 L'existant

À l'origine, la norme 802.11 ne comprenait qu'un mécanisme relativement faible pour assurer la sécurité des données. Ce mécanisme est appelé WEP pour *Wired Equivalent Privacy*. Conscients de la faiblesse de cette protection, les fabriquants de matériel WiFi ont développé une seconde solution appelée WPA pour *WiFi Protected Access*. Cette solution n'étant qu'une rustine de la première, l'IEEE s'est mise à plancher sur la norme 802.11i

pour développer un mécanisme de sécurité beaucoup plus fiable. À ce jour, cette norme n'est toujours pas ratifiée.

Ces trois technologies sont propres aux réseaux WiFi. Il existe d'autres solutions plus génériques applicables aux réseaux WiFi. Ces solutions sont connues sous le nom *Virtual Private Network* (VPN).

III.2.1 Le WEP

Le sigle WEP signifie *Wired Equivalent Privacy*. Il était donc, dès l'origine, peu ambitieux. Aucun mécanisme de distribution de clefs n'a été prévu, ce qui fait que habituellement tout le monde partage la même clef. Cela signifie que deux utilisateurs légitimes du réseau peuvent s'observer mutuellement sans aucune difficulté. Il est cependant possible de mettre en place l'infrastructure nécessaire pour que chaque utilisateur dispose de sa propre clef. Sa mise en place est toutefois très rare.

Il n'y a de plus aucun mécanisme d'authentification en dehors de cette clef. Il n'est donc pas possible d'identifier précisément l'utilisateur qui peut donc se faire passer pour un autre utilisateur légitime. De plus, le mécanisme d'authentification ne fonctionne que dans un seul sens : il est tout à fait possible de mettre en œuvre un point d'accès pirate et ainsi de détourner les communications des utilisateurs. Il n'y a de plus aucune protection contre le rejeu.

WEP utilise un moteur de chiffrement par flux RC4 pour la confidentialité. L'intégrité est assuré par une simple somme de contrôle CRC32. Cette dernière n'a jamais été prévue pour résister à un attaquant : elle sert à l'origine à détecter des erreurs. Il est donc très simple de modifier les trames envoyées.

De nombreux papiers font état des faiblesses structurelles extrêmement importantes de WEP : [11, 33, 21, 24, 10]. En août 2001, [21] présente une méthode de cryptanalyse sur le WEP qui permet de récupérer de manière totalement passive la clef utilisée pour le chiffrement en quelques heures. De nombreux outils implémentent cette attaque. Elle se base sur certaines faiblesses qui ont été depuis évitées par les fabriquants qui filtrent les clefs permettant à cette attaque de réussir : ce mécanisme est connu sous le nom de *weak key avoidance*.

Cependant, en août 2003, un hacker connu sous le nom de Korek publie un outil qui rend de nouveau les attaques contre WEP très faciles. De nombreux outils implantent désormais cette attaque qui permet de récupérer une clef WEP en quelques minutes ou quelques heures. Airsnort [2] et Aircrack [1] sont deux de ces outils.

Il apparaît donc clairement que le WEP n'est pas du tout adapté à assurer la sécurité d'un réseau : de nombreuses attaques pratiques existent actuellement.

III.2.2 Le WPA

Devant le manque flagrant de sécurité du WEP, les fabriquants devaient trouver une parade sous peine de voir leurs ventes affectées. L'IEEE préparait alors la norme 802.11i qui était basée sur le chiffrement AES [17]. Cependant, cela nécessitait de remettre à jour tout le parc : les cartes wifi actuelles n'ont pas la puissance nécessaire pour faire du chiffrement AES.

III.2.2.a Un WEP amélioré

WPA est donc un protocole qui repose sur WEP et permet ainsi de conserver le matériel existant en nécessitant uniquement une mise à jour du *firmware*, qui est le logiciel embarqué dans le matériel.

Pour résoudre le problème de confidentialité du WEP qui permettait de récupérer la clef utilisée pour chiffrer, WPA met en œuvre TKIP (*Temporal Key Integrity Protocol* qui permet de dériver d'une clef d'autres clefs temporaires. À partir d'une clef maître, ce mécanisme est donc capable de générer une nouvelle clef pour chaque paquet. Cette clef est passée au moteur WEP qui sert donc toujours à effectuer le chiffrement mais comme chaque paquet est chiffré avec une clef différente, l'attaque décrite pour le WEP n'est plus possible en pratique.

Pour le problème de l'intégrité, un nouveau mécanisme a été ajouté. Celui-ci s'appelle MIC pour *Michael Integrity Check*. Il s'agit cette fois-ci d'un mécanisme cryptographique et il est donc beaucoup plus difficile de l'attaquer.

WPA peut fonctionner avec une clef partagée par tous les utilisateurs ou avec une clef par utilisateur. La première méthode est connue sous le nom *WPA Personal*. Tous les utilisateurs partageant le même secret, ils peuvent voir les données échangées par le voisin. La seconde méthode nécessite la mise en place d'une PKI (*Public Key Infrastructure*). C'est cette dernière qui offre le plus de sécurité.

On peut se référer à [10] pour une description plus précise de WPA, notamment à propos des différents mécanismes d'authentification qu'il propose.

III.2.2.b Et les défauts ?

Actuellement, WPA est ce qui semble s'imposer dans le monde du WiFi. Il est pour le moment en pratique assez sûr et il est supporté par une grande partie du matériel existant.

Toutefois, plusieurs défauts persistent. Conceptuellement, il apporte les sécurités que nous avons énoncé, mais il est condamné sur le long terme en raison des faiblesses de chacun de ses éléments. Il est encore un peu trop jeune pour avoir été étudié extensivement, mais des failles importantes ont déjà été trouvées. On pourra par exemple se reporter à [32] qui décrit comment l'espace de recherche peut être réduit de 128 bits à 105 bits. Cela ne permet toujours pas directement une attaque, mais c'est une faille très importante

pour un protocole cryptographique. Il y a donc un risque pour que dans un an, d'autres techniques soient découvertes et permettent de casser WPA assez facilement. Il sera certes toujours possible d'adopter 802.11i, mais les données actuelles pourront alors être déchiffrées[1].

WPA dispose d'autres défauts comme la nécessité de mettre en place une PKI et de distribuer des certificats. C'est un mécanisme assez lourd à mettre en place malgré le gain non négligeable de sécurité d'une telle procédure.

À noter également que le support de WPA est inégal. Si le matériel actuel le supporte sans problème, des cartes wifi plus anciennes peuvent ne pas le supporter, même après une mise à jour. Au niveau logiciel, certains systèmes ont un support inégal du WPA. Par exemple, pour Mac OS X, la version 10.3 est nécessaire. Pour Linux, la procédure est assez ardue et seule certaines cartes sont supportées.

Ces défauts sont cependant loin d'être rédhibitoires.

III.2.3 Le 802.11i

802.11i [14] est une norme en cours de développement par l'IEEE. Elle est parfois appelée WPA2 bien qu'elle en soit très éloignée. Elle repose cette fois sur le chiffrement AES au lieu de WEP. AES, pour *Advanced Encryption Standard* [17], est un mécanisme de chiffrement adopté comme un standard par le gouvernement américain en remplacement du vieillissant *Data Encryption System* (DES). AES a été développé par deux cryptographes belges et a été sélectionné face à d'autres algorithmes proposés. Il a été étudié par de très nombreux cryptographes et est donc, a priori, très solide. AES assure à lui seul une bonne partie des propriétés de sécurité.

802.11i s'annonce donc particulièrement solide. Il existe déjà du matériel supportant ce protocole, mais étant donné que celui-ci n'est pas encore ratifié, il est possible que ce matériel ne soit plus conforme à la spécification lorsque celle-ci sera ratifiée.

Son grand défaut est donc sa faible diffusion, pour le moment.

III.2.4 Les VPN

Il existe des solutions de chiffrement et d'authentification qui ne sont pas propres au WiFi. Initialement, ces solutions ont été développées pour permettre à un utilisateur nomade de se connecter en toute sécurité depuis son domicile vers son entreprise en passant par Internet. Il est toutefois possible de les mettre en œuvre pour assurer la sécurité d'un réseau WiFi.

[1] Il est important de bien comprendre que les données confidentielles actuellement peuvent l'être demain. Un schéma de chiffrement doit donc assurer une protection suffisante dans le temps : un attaquant peut enregistrer un flux chiffré et attendre deux ou trois ans que la technologie nécessaire pour le décrypter soit disponible. DES a assuré la sécurité des données qu'il protégeait pendant une trentaine d'année. La même performance est attendue d'AES.

Il existe de nombreuses technologies de VPN. La question de l'interopérabilité est alors assez épineuse. Deux technologies ont émergé :

PPTP , pour *Point-to-Point Tunneling Protocol*, est un protocole publié par Microsoft. Il est donc disponible sur les versions les plus courantes de Windows et mis en avant par Microsoft. Il repose toutefois sur des mécanismes d'authentification relativement faibles et n'est donc pas considéré comme sûr. D'ailleurs, Microsoft a cessé son développement en faveur de L2TP par-dessus IPsec. On pourra se référer à [35] pour connaître l'ensemble des problèmes de PPTP.

IPsec , pour *IP security*, est un protocole ratifié par l'IETF mettant en œuvre un ensemble de mécanismes destinés à assurer la confidentialité et l'authenticité des paquets qu'il véhicule. Il est le fruit de nombreuses années de travail et dispose de très nombreuses implantations et du support de nombreux industriels. Il est capable d'utiliser AES comme algorithme de chiffrement, DH (*Diffie-Hellman*) et DSA (*Digital Signature Algorithm*) pour l'échange de clefs via le protocole IKE [34] et HMAC-SHA1 pour l'intégrité. Il reste un protocole assez ardu à mettre en œuvre. Pour une évaluation de IPsec, on pourra se reporter à [26, 28, 27, 23, 20, 22]. Nous retiendrons essentiellement qu'IPsec a été décortiqué pendant plusieurs années par des cryptographes et qui en ont déduit sa solidité.

◦ ◦ ◦ ◦ ◦ ◦ ◦ ◦ ◦ ◦ ◦

III.3 La solution retenue par le Cr@ns

Les deux solutions qui nous semblaient répondre aux critères de sécurité énoncés ci-dessus et qui étaient exploitables à l'époque du choix étaient donc WPA et IPsec. Cependant, le support WPA était (et est toujours, dans une moindre mesure) extrêmement spartiate sous Linux ce qui a motivé le choix d'IPsec.

Adopter un protocole de classe militaire ne suffit cependant pas pour assurer la sécurité de l'ensemble. Nous allons donc détailler ici en quoi cette sécurité est assurée. IPsec n'est alors qu'un des composants de celle-ci.

III.3.1 Le modèle réseau en couches

Pour bien comprendre les problèmes de sécurité réseau, il est nécessaire de comprendre dans un premier temps le modèle en couches qui décrit les protocoles réseau. Il existe généralement deux modèles. Le premier est celui de l'OSI [18]. Il constitue le modèle académique. Pour illustrer notre propos, nous allons plutôt utiliser le second modèle, dit DoD [15] pour *Department of Defense*, qui en est une version simplifiée en quatre couches.

L'idée du modèle en couches est que chaque couche constitue un protocole qui n'utilise que les fonctions offertes par le modèle qui lui est immédiatement

inférieur et n'offre ces fonctions qu'au modèle qui lui est immédiatement supérieur. On peut agencer les couches comme on le désire pourvu qu'il y ait compatibilité des interfaces entre les couches. Quand on a réussi à empiler un certain nombre de couches, on obtient ce que l'on appelle une *pile de protocoles*. On l'illustre comme dans la figure III.1.

Processes
Transport
Internet
Network access

FIG. III.1 – Le modèle en couches DoD

Le modèle DoD dispose donc de quatre couches :

Network access est la couche qui permet d'échanger des informations sur un réseau local. Sur ce type de réseau, chaque machine peut directement contacter une autre machine. 802.11 est une telle couche, au même titre que Ethernet. Cet couche offre donc comme service la possibilité pour une machine de contacter et d'échanger des informations avec une autre machine sur le même réseau physique.

Internet est la couche qui permet à des machines de réseaux différents de communiquer entre elles. Cela correspond au protocole IP. Chaque machine est identifiée par un numéro IP comme 138.231.136.6 et connaît le moyen de contacter directement ou indirectement les autres machines. Le fait de contacter indirectement une autre machine induit le processus de routage : un paquet à destination d'une machine qui n'est pas sur le même réseau est passée à une autre machine qui transmet, directement ou indirectement, le paquet. Cette couche utilise la couche d'accès au réseau qui lui est inférieure pour donner le paquet à une machine du même réseau (qui est soit le destinataire final, soit l'intermédiaire choisi pour relayer le message).

Transport est la couche qui va réellement permettre de transmettre des données entre plusieurs machines. Cela correspond à TCP [9] ou UDP. Cette couche assure le multiplexage des paquets entre les différents services à l'aide de la notion de ports : un service est associé à un port sur une machine donnée. Elle utilise le service de transport entre machines n'appartenant pas au même réseau de la couche Internet pour envoyer les paquets d'une machine à l'autre. Un paquet a alors comme source une IP et un port et comme destination une IP et un port. Le port permet de savoir à quelle application est destinée le paquet. TCP assure en plus un service d'intégrité, de fiabilité, d'ordonnancement et de non duplication des données. Il offre de plus la notion de connexion permettant à deux applications de communiquer entre elles dans le cadre d'une session unique.

Process layer est la dernière couche et correspond aux applications. À l'aide des services offerts par TCP et UDP dans la couche inférieure, les applications vont mettre en œuvre un protocole qui leur est propre pour communiquer entre elles, sur des machines distantes.

Un serveur de mail va par exemple pouvoir envoyer un mail à un autre serveur de mail via le protocole SMTP (de niveau 4 donc). Pour cela, il va utiliser les services de TCP (de niveau 3) pour ne pas avoir à gérer les problèmes de pertes de paquets et s'adresser directement au serveur de mail (port 25) distant. TCP à son tour va utiliser IP pour faire voyager le paquet et sur les réseaux locaux, IP va utiliser Ethernet (ou autre) pour transmettre les données physiquement.

Si l'application n'est pas intéressée par les services offerts par TCP, elle peut utiliser UDP qui lui assure toujours le multiplexage mais ne lui garantit pas de recevoir tous les paquets.

Ajoutons enfin que tous les équipements ne travaillent pas au même niveau. Un switch est un équipement de niveau 1, il ne connaît donc pas les autres couches. Un routeur est un équipement de niveau 2. Il existe également des équipements de niveau 4. Un équipement de niveau 1 ne prend pas en compte les informations contenues dans le niveau 2. Ainsi, un switch ne prend pas en compte les informations comme l'adresse IP, sauf cas particuliers...

Une autre façon d'illustrer le modèle en couches est présenté dans la figure III.2 page ci-contre. Chaque niveau émet des trames ou des paquets d'un format donné et encapsulent dans leurs corps les niveaux supérieurs. Quand un paquet doit être examiné au niveau n, les entêtes au niveau $n-1$ sont détruites tandis que quand il doit être transmis pour être envoyé sur le réseau, on lui rajoute les entêtes du niveau n.

III.3.2 La sécurité dans le modèle en couches

TCP/IP n'a pas été prévu dans une optique de sécurité. Il n'offre aucun mécanisme de défense contre un adversaire capable d'observer les trames réseau. Cela n'a longtemps pas été un problème vu qu'il était difficile d'effectuer cette observation passive. Ce n'est plus du tout le cas avec des réseaux WiFi vu que tout le monde est à même d'effectuer cette observation.

Toutefois, la sécurité n'est qu'un service comme un autre dans le modèle de couches. Si une des couches assure par exemple la confidentialité, toutes les couches supérieures en bénéficient automatiquement. Il convient d'éviter les conclusions hâtives : si par exemple, on chiffre le niveau 1, on assure la confidentialité sur le segment chiffré uniquement. Si le segment se trouvant derrière est lui aussi chiffré, cela n'assure pas la sécurité de toute la chaîne si l'élément intermédiaire travaille au niveau 2. En effet, celui-ci va déchiffrer ce qui arrive d'un côté et le rechiffrer pour le renvoyer. Un attaquant peut alors agir sur cet équipement pour récupérer le clair.

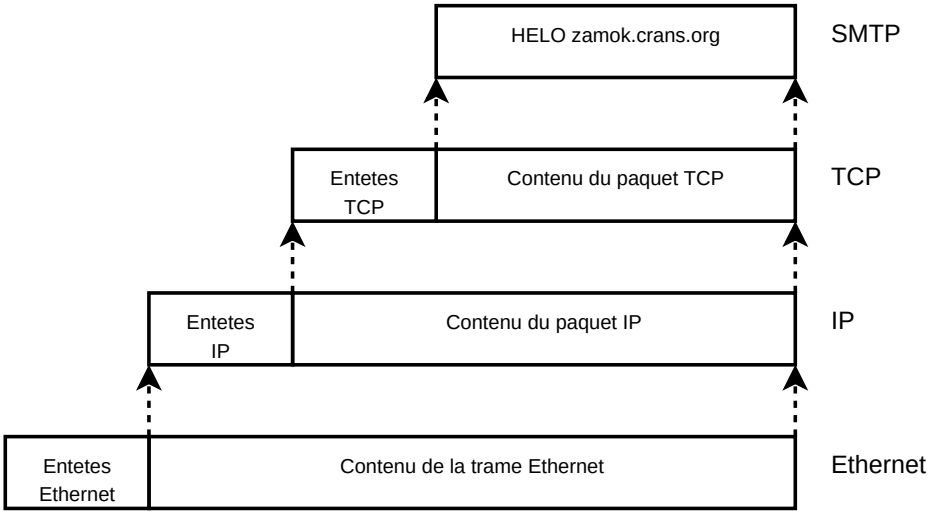


FIG. III.2 – Une autre illustration du modèle en couches

C'est un scénario que l'on peut retrouver très facilement en pratique. Imaginons que nous utilisions des bornes wifi avec WPA (qui chiffre la couche 1). Malgré le bon niveau de chiffrement, il suffira pour un attaquant de pirater la borne wifi pour obtenir tout ce qui y transite, en clair. Ce n'est pas improbable vu que celles-ci se trouvent généralement dans des endroits non protégés. C'est en pratique encore plus simple vu que le chiffrement se fait uniquement sur les ondes radio. La partie filaire ne l'est pas. Il suffit donc de se placer derrière la borne pour obtenir tout le trafic en clair ! Cette attaque ne faisant appel à aucun exploit technique est décrit dans la figure III.3.

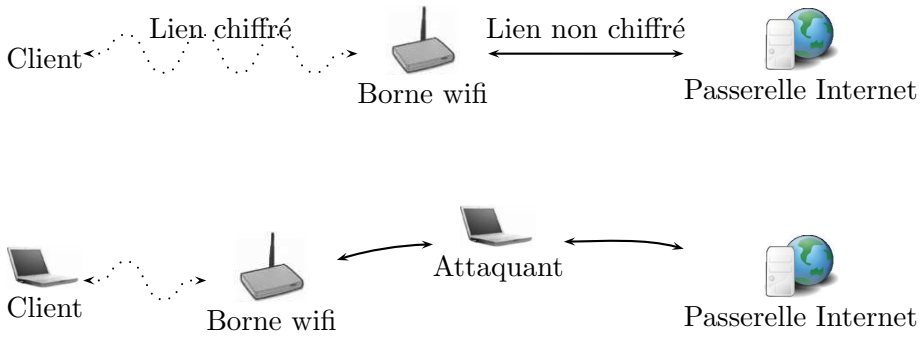


FIG. III.3 – Attaque simple dans le cas de la non sécurisation physique des bornes

Un autre piège concerne l'authentification. Si la couche de niveau 1 assure

l'authentification, les couches supérieures en bénéficient automatiquement. Cependant, l'authentification se fait avec les éléments disponibles au niveau 1, c'est-à-dire les adresses des cartes réseau. Les éléments de niveau 2, c'est-à-dire les adresses IP, ne sont pas authentifiés. Il est ainsi possible de prendre une fausse IP. Il faut donc penser à lier l'adresse physique et l'adresse IP pour s'assurer de l'authenticité des paquets.

Enfin, le déni de service est également un service. Si une couche est sensible au déni de service, toutes les couches supérieures en « bénéficient ». C'est pour cette raison qu'il est vain de tenter de contrer le déni de service avec le WiFi : il est quasiment impossible de protéger le niveau 1 de ce type d'attaque. On veillera par contre à ce que celle-ci ne se propage pas au-delà des segments touchés : il ne faudrait pas que le réseau filaire en souffre.

III.3.3 Description de la mise en œuvre

Nous avons déjà expliqué que nous mettons en œuvre IPsec. Il s'agit d'un protocole de niveau 2. En fait, ce protocole est un protocole de type IP dans IP : il utilise les services de IP (donc serait plutôt de niveau 3), mais il fournit les mêmes services qu'IP (donc serait plutôt de niveau 2). Pour des raisons de simplicité, on va donc considérer qu'il s'agit d'une altération d'IP et qu'il travaille au niveau 2.

Toutefois, IPsec n'est qu'une partie de la solution. Détaillons donc la solution au complet. Nous disposons d'un routeur, appelé ici `Nectaris`, qui est une passerelle IPsec. Les clients communiqueront en IPsec avec cette machine. La communication sera donc chiffrée de bout en bout entre cette machine et les clients. Nous disposons également d'un certain nombre de bornes wifi. Celles-ci vont agir comme des switchs un peu spéciaux. Il aurait été possible à peu de coût de transformer ces bornes en routeur et ainsi s'abstraire d'un certain nombre de problèmes qui vont apparaître par la suite. Toutefois, nous désirions qu'il soit possible de conserver la même IP quelle que soit la borne sur laquelle nous nous connectons et rendre possible la migration d'une borne à une autre. Ceci est beaucoup plus facile avec un équipement de niveau 1 qui dispose d'un protocole (ARP) pour « localiser » les clients qu'avec un équipement de niveau 2 qui doit savoir par avance à quelle borne s'adresser pour savoir où envoyer un paquet.

Pour résumer, un client wifi se connecte à une borne (au niveau 1). Il communique, à travers cette borne, avec `Nectaris` en IPsec (au niveau 2), de bout en bout. La figure III.4 représente le schéma du réseau au niveau 1 tandis que la figure III.5 page suivante celui au niveau 2.

III.3.4 Étude de la sécurité

Nous allons donc étudier plus en détail la sécurité de la solution en se basant sur le modèle en couches. Nous ne nous intéressons qu'à la sécurité

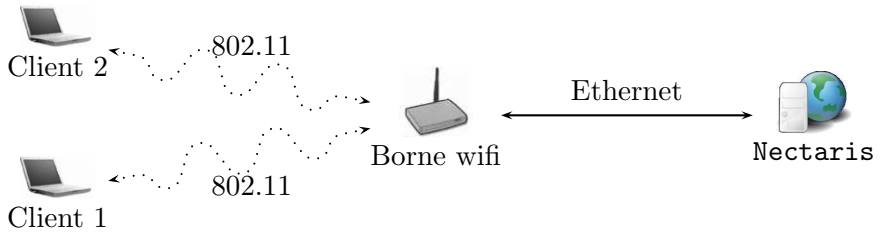


FIG. III.4 – Schéma du réseau au niveau 1

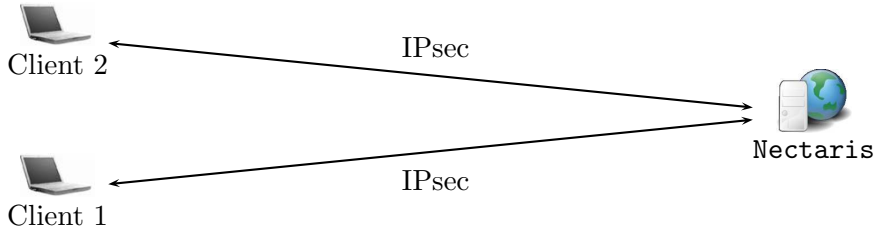


FIG. III.5 – Schéma du réseau au niveau 2

entre le client et `Nectaris`. La sécurité sur le reste du réseau est hors de propos ici.

III.3.4.a Confidentialité et authenticité

La *confidentialité* est assurée à partir du niveau 2 grâce à IPsec qui utilise l'algorithme de chiffrement AES [17] dans ce but. Le chiffrement est assuré de bout en bout : lorsqu'un client émet des données, les équipements intermédiaires ne voient passer qu'un flux chiffré. Seul `Nectaris` est capable de déchiffrer celui-ci. Les autres clients ne partagent pas la même clef, ils sont donc également incapables de déchiffrer le flux. La confidentialité est donc assurée pour le niveau 2 et les niveaux supérieurs. Le niveau 1 n'est donc pas sécurisé. Cela permet d'obtenir comme information les adresses MAC et les adresses IP sans aucun effort. Nous considérons ces informations comme publiques. Leur divulgation n'est donc pas notre soucis.

L'*authentification* entre le client et `Nectaris` est mutuelle : le client ne parlera qu'avec `Nectaris` et celui-ci n'acceptera de parler qu'avec le client. Les deux entités partagent une clef[2] connue de eux seuls. Les paramètres de l'authentification sont les adresses IP des deux parties et la clef partagée. Ainsi, le client est assuré de parler avec une machine ayant l'IP de `Nectaris` et partageant le même secret qu'elle. À moins de révéler le secret, il n'est donc pas possible qu'il parle en fait à un intrus. Inversement, `Nectaris` est sûre de parler avec le client. L'IP est donc liée fortement au client et le client ne peut pas falsifier celle-ci. L'authentification fournie par IPsec donne gratuitement l'*intégrité* des données et le *non-rejeu*. Le niveau 2 remplit bien nos conditions sur l'authenticité des paquets. Les niveaux supérieurs également mais les paramètres restent les IP. Généralement, on ne demande pas d'autres données : on cherche principalement à identifier les utilisateurs et les IP sont suffisantes pour cette fonction.

III.3.4.b Usages frauduleux

Laissons le déni de service de côté et regardons quels sont les usages non autorisés du réseau qu'un intrus pourrait effectuer. Au niveau de la couche 2, il ne pourra pas discuter avec `Nectaris`, par contre, il pourrait accéder aux autres services du réseau et discuter avec un autre client ou avec une autre machine de notre réseau. Il existe deux parades possibles :

1. S'arranger pour qu'il n'existe sur le réseau comme seul équipement de niveau 2 (le niveau 1 ne donne accès à aucun service) que `Nectaris`. C'est possible physiquement, mais coûteux au niveau matériel. Une solution intermédiaire est d'utiliser les VLAN. Il s'agit d'une technologie permettant de compartimenter un réseau local en plusieurs réseaux lo-

[2] Une architecture utilisant des certificats serait encore plus sûre. Il est possible de la mettre en place assez facilement au niveau technique tout en conservant tout ce qui est dit ici. Cependant, la gestion de certificats est assez lourde administrativement. Nous ne nous sommes donc pas lancé là-dedans.

caux. Il serait ainsi possible de faire un réseau virtuel ne contenant que `Nectaris` et les clients WiFi. Cela ne suffit pas puisqu'il faut encore protéger les autres clients. De plus, nous n'avons pas encore les moyens de mettre en place un tel réseau car certaines bornes sont hébergées par des adhérents qui branchent ensuite leur PC dessus.

2. Tricher un peu avec le modèle en couches et rendre les bornes plus intelligentes. Tout client passe automatiquement par une borne pour toutes ses communications, y compris avec les autres clients. Nous avons donc configuré les bornes pour qu'elles empêchent au client de se voir entre eux (au niveau 2) et pour qu'ils ne puissent parler qu'avec `Nectaris` (au niveau 2). Nous avons donc des équipements de niveau 1 qui prennent en compte le niveau supérieur. Nous avons également interdit tout autre protocole que IPsec[3] au niveau 2.

Nous vérifions de plus que lorsqu'un client prétend parler avec `Nectaris` au niveau 2, il parle bien avec `Nectaris` au niveau 1 ; plus en détail, nous vérifions que l'adresse MAC correspond à l'adresse IP. Ainsi, chaque client ne peut parler qu'avec `Nectaris`, garante de la politique de sécurité.

III.3.4.c Dénis de service

Comme nous l'avons indiqué précédemment, il n'est pas possible de se prémunir entièrement du déni de service. Un attaquant peut brouiller les signaux radios et ainsi empêcher les clients de se connecter. Nous ne tenterons rien de ce côté.

L'attaquant dispose de deux autres sources de déni de service qui peuvent nuire au réseau. Il peut envoyer ce que bon lui semble au niveau 1. Cependant, les bornes assurent que seuls les paquets IPsec peuvent circuler au niveau 2. Elles laissent également passer des paquets ARP. ARP, pour *Address Resolution Protocol*, est un protocole permettant de faire la correspondance entre une IP et une adresse MAC. Fausser cette correspondance pourrait conduire une machine à s'adresser au mauvais destinataire. Une attaque appelée *ARP poisoning* consiste à exploiter la faiblesse de ce protocole pour intercepter toutes les données destinées à une machine. On pourra se référer à [12] pour une description précise de cette attaque et des solutions possibles. Dans notre cas, cette attaque peut uniquement aboutir à un déni de service (l'attaquant ne peut rien faire des données récupérées, il s'agit d'IPsec).

Cependant, pour minimiser cet aspect, les bornes effectuent un filtrage des requêtes ARP et vérifient la correspondance MAC/IP à l'aide d'une table. Il n'est donc pas possible d'émettre de fausses requêtes ARP.

L'attaquant peut encore tenter de falsifier le niveau 1 pour des paquets contenant de l'IPsec. S'il modifie l'adresse MAC cible, il sera arrêté par les bornes qui vérifient que l'adresse MAC est celle de `Nectaris`. Il lui reste l'adresse MAC source qui lui permettrait de fausser la correspondance

[3] On verra un peu plus loin que ce n'est pas tout à fait le cas

MAC/port des switchs et ainsi faire perdre à certaines machines des paquets (les switchs croyant que la machine a changé de port). Pour éviter cette attaque, les bornes vérifient sur toutes les trames Ethernet la correspondance entre la MAC source et l'IP source. Cette attaque n'est donc pas possible.

III.3.4.d Les services en clair

Nous avons dit précédemment que le niveau 2 ne voyait circuler que de l'IPsec. En fait, nous permettons aussi la circulation de trames IP non chiffrées contenant au niveau 4 des trames HTTP, HTTPS (qui correspondent au Web), des trames DHCP (qui est un protocole de configuration automatique) et des trames DNS (qui permettent de faire la conversion de noms en IP).

Le cas du DHCP Le *DHCP* [16] nous permet de configurer automatiquement les clients. Ceux-ci, quand ils se connectent, émettent une requête à destination des serveurs DHCP qui leur donnent en retour leur adresse IP, le réseau sur lequel ils se trouvent et le routeur par défaut. Tout le monde peut se faire passer pour un serveur DHCP. Il est donc possible pour un attaquant de répondre de fausses informations. Selon la vigilance de l'utilisateur, cela peut conduire à un déni de service ou à un vol de données. En effet, l'utilisateur ne parviendra pas à établir la connectivité IPsec. Il peut donc décider de ne pas l'activer et émettre les données en clair à destination de l'attaquant. Il est donc important qu'un utilisateur s'assure du succès de la connexion IPsec avant d'envoyer des données. Sous Windows, Mac OS X et GNU/Linux, il n'est pas possible d'émettre des données si l'établissement de la connexion IPsec échoue.

Afin d'éviter qu'un serveur DHCP pirate ne perturbe tout le réseau, les bornes ne laissent en réalité pas passer les trames DHCP. Elles embarquent un relai DHCP. Les perturbations seront alors localisées à une seule borne.

Enfin, les informations véhiculées par le DHCP ne sont pas confidentielles.

Le cas de l'HTTP et de l'HTTPS Nous laissons également passer l'HTTP (correspondant au Web) afin que les nouveaux clients puissent télécharger les programmes et les instructions nécessaires à la configuration de leur connexion WiFi. Ces données ne sont pas confidentielles. Par contre, il est important que le client s'assure de l'identité du site Web distant : il ne faut pas qu'il télécharge un virus ou un autre programme malicieux. Le second problème est de s'assurer que cette entrée ne permet pas d'accéder à autre chose que le site web en question. Ce dernier est localisé sur `Nectaris`.

Au niveau 2, les bornes s'assurent que seules les trames à destination de `Nectaris` passent (et leurs réponses). Au niveau 1, les adresses MAC sont vérifiées. Au niveau 3, seules les trames à destination du serveur Web

passent. En bonus, toutes les requêtes qui ne sont pas à destination de `Nectaris` mais qui sont des requêtes Web sont redirigées sur `Nectaris`. Ainsi, quelqu'un qui tente d'accéder à un site extérieur tombera sur les instructions nécessaires pour configurer son accès WiFi.

`Nectaris` dispose d'un serveur Web capable de répondre en SSL ou non. S'il ne répond pas en SSL, il renvoie alors le client sur le serveur qui répond en SSL. SSL, pour *Secure Socket Layer*, est un protocole de niveau 4 assurant la confidentialité et l'authenticité. Une analyse de ce protocole est disponible dans [30]. Dans notre cas, l'authentification ne se fait que dans un seul sens : le client peut s'assurer de l'identité du serveur à l'aide du certificat que celui-ci lui présente. Il ne nous paraît pas utile d'avoir à authentifier le client. Si un attaquant monte un faux serveur Web et redirige le client sur celui-ci, celui-ci obtiendra un avertissement.

En pratique, cette approche dispose d'une faiblesse : peu d'utilisateurs vérifient que le certificat correspond bien au site web et ignorent les avertissements éventuels. Si l'attaquant renvoie sur un site en clair, il n'y aura même pas d'avertissement et le client devra se rendre compte de lui-même que le site n'est pas sécurisé, ce qui n'est pas évident vu que l'utilisateur ne va pas entrer ou consulter des données confidentielles comme il le ferait sur un site bancaire. Enfin, notre certificat n'est pas signé par une autorité reconnue. Nous l'utilisons en interne pour tous les services et encourageons les utilisateurs à l'enregistrer dans leur navigateur, mais un nouveau venu ne pourra pas savoir s'il s'agit vraiment du nôtre. Il serait nécessaire qu'il le télécharge d'une source sûre.

Le cas du DNS Le DNS, *Domain Name Service* [31], est un protocole qui permet de convertir un nom comme `www.crans.org` en une IP comme `138.231.136.6`. Nous devions laisser l'accès à ce service pour permettre aux connectés d'accéder au site Web. Dans le cas contraire, ils auraient été contraints de taper l'IP. Les bornes redirigent toutes les requêtes DNS vers leur propre serveur qui répond invariablement l'adresse IP de `Nectaris`. Nous ne sommes donc pas concernés par les problèmes de confidentialité ou d'authenticité à ce niveau : cette adresse IP n'est pas secrète et fournir une fausse IP ne mènera nulle part vu qu'au niveau IP, seule `Nectaris` est accessible.

Une attaque courante sur les hotspots est de mettre en place un tunnel par-dessus le protocole DNS afin d'accéder de manière non autorisée à Internet. Imaginons que nous contrôlions le domaine `machin.com`. Cela signifie que nous pouvons contrôler les réponses du serveur de DNS qui sert ce domaine. Nous allons utiliser cette possibilité pour encapsuler des données arbitraires dans le flux DNS et ainsi utiliser de manière frauduleuse le réseau. Par exemple, si on veut faire passer l'information `zorglub` à la machine distante, on peut demander à résoudre le nom `zorglub.machin.com`.

Cette requête va parvenir au serveur de DNS que nous contrôlons et celui-ci saura donc que nous lui avons passé l'information `zorglub`. En retour, il peut par exemple répondre qu'il s'agit d'un alias sur `compris.machin.com`. Nous recevrons alors cette réponse.

Cette attaque n'est pas possible ici car le serveur DNS des bornes n'interrogent aucun DNS et se contentent de répondre l'adresse IP de `Nectaris`.

∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘

III.4 Conclusion

Au regard du modèle en couches, nous pouvons donc affirmer que la solution retenue est sûre du point de vue de la confidentialité et de l'authenticité, à hauteur de la sécurité fournie par IPsec. L'attaquant ne dispose pas de moyens techniques simples d'accéder aux données de l'utilisateur. Tant bien même il mettrait la main sur les bornes, pour les remplacer par des versions modifiées, cela ne l'avancerait guère : elles se contentent de laisser passer le signal chiffré et ne contiennent absolument aucun secret.

Du point de vue déni de service, un certain nombre de contre-mesures sont en place et les risques sont relativement faibles et resteront localisés.

Enfin, l'attaquant n'ayant pas d'accès au réseau ne dispose que de services extrêmement limités : il n'a guère accès qu'au serveur DHCP et au site Web. Cependant, s'il a un accès physique aux bornes, il dispose d'un accès plus complet aux services du réseau car ce sont les bornes qui le maintiennent en dehors de celui-ci. Cette faiblesse sera grandement amoindrie quand nous mettrons en place les VLAN.

CHAPITRE IV

Technique

Ce chapitre est une description technique de la solution mise en œuvre au Cr@ns. Nous disposons de deux types d'éléments. Le premier est le serveur `Nectaris` qui sert de passerelle. Le second est constitué de l'ensembles des bornes WiFi que nous pouvons disséminer partout.

∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘

IV.1 Le serveur `Nectaris`

Ce serveur est hébergé dans nos locaux et agit comme un routeur. Il est équipé du système d'exploitation OpenBSD [6] dans sa version 3.6 avec les derniers *patchs* appliqués. L'ensemble des services dédiés au WiFi tournent sur cette machine. Les services inclus dans le système de base bénéficient d'un audit de sécurité de la part de l'équipe des développeurs. La sécurité est le *credo* du projet OpenBSD qui met en œuvre un certain nombre de mécanismes pour éviter les attaques les plus courantes, comme par exemple la séparation systématique des privilèges ou la mise en place d'un bac à sable (*chroot*) pour chaque application dès que c'est possible. Leur politique s'avère payante au fil des ans car peu de failles sont finalement découvertes et très peu d'entre elles sont exploitables. On pourra consulter la page `http://www.openbsd.org/errata.html` pour consulter les failles concernant OpenBSD.

Au niveau matériel, ce serveur dispose d'une carte d'accélération crypto, permettant de décharger le processeur central des opérations cryptographiques, et d'une carte gigabit.

IV.1.1 Le démon ISAKMP et la pile IPsec

L'élément principal de `Nectaris` est le démon ISAKMP, ISAKMPd, et la pile IPsec. Ces deux éléments constituent le fer de guerre de la solution WiFi. Le démon ISAKMP [29] est chargé de la distribution des clefs à l'aide du protocole IKE et la gestion des associations de sécurité. Il est fourni et

installé en standard dans OpenBSD. Sa configuration est générée à partir de la base LDAP, à chaque changement de celle-ci. Les éléments d'authentification sont les adresses IP et les secrets partagés : chaque client dispose de sa propre adresse IP et de son propre mot de passe. Celui-ci est généré de manière aléatoire (et non choisi) pour des raisons de sécurité.

Le démon ISAKMP négocie les différents paramètres de sécurité avec le client. Celui-ci peut choisir entre 3DES et AES pour le chiffrement. Ce dernier est préféré car il est à la fois plus rapide et plus sûr. Cependant, les clients Windows ne disposent que de 3DES. Le mécanisme d'intégrité est au choix HMAC-MD5 et HMAC-SHA1 ; ce dernier étant préféré. Si le client le désire, il peut également activer *Perfect Forward Secrecy* (PFS) qui permet d'éviter la compromission des clefs temporaires de session suite à la compromission de l'une d'entre elles.

Les fichiers de configuration pour ISAKMPd sont :

- `/etc/isakmp/isakmpd.conf`
- `/etc/isakmp/isakmpd.policy`

Une fois les paramètres de sécurité négociés, le démon IKE fournit à la pile IPsec les paramètres nécessaires pour construire l'association de sécurité et ainsi chiffrer les communications entre le client et le serveur. OpenBSD contient également en standard une pile IPsec basée sur l'implantation de Kame [4]. Cette implantation sait tirer partie de l'accélération matérielle fournie par les cartes cryptos. `Nectaris` dispose d'une de ces cartes, ce qui lui permet de chiffrer 200 MBps de flux.

IV.1.2 Le serveur DHCP

`Nectaris` contient également un serveur DHCP chargé de la configuration automatique des clients. Sa configuration est regénérée automatiquement à partir du contenu de la base LDAP. Ce serveur fait également partie du système de base et sert de relai aux bornes WiFi.

IV.1.3 Le serveur Web

Une version modifiée de Apache 1.3 est également disponible. Elle ne sert que des pages statiques extraites de notre wiki. L'extraction se fait manuellement pour éviter qu'une personne mal intentionnée ne modifie les pages avec des instructions erronées. Les connexions non sécurisées sont redirigées vers le serveur sécurisé. Celui-ci est capable de détecter si le client a été redirigé de force ou de plein gré pour lui afficher la page d'explication adéquate.

Le serveur Web fait aussi partie de l'installation de base.

IV.1.4 Le démon wifi-update

Afin de permettre aux bornes de recevoir les instructions de configuration, un démon maison tourne sur `Nectaris`. Celui-ci permet de servir des

scripts de configuration pour les bornes. Celles-ci se connectent au boot puis à intervalles réguliers à `Nectaris` qui leur envoie des scripts de configuration que celles-ci exécutent.

Le démon est écrit en Python [7] en utilisant le framework Twisted [8]. Ce dernier permet d'écrire des serveurs très performants et peu gourmands en ressource. Toutes les connexions sont servies à partir d'un processus unique.

Toutes les communications sont protégées avec le protocole SSL et l'authentification est assurée à l'aide de certificats.

IV.1.4.a Fonctionement

Celui-ci détermine dans un premier temps le nom de la borne, cherche les fichiers de configuration qui lui correspondent puis envoie l'ensemble de ceux-ci. Une fois que la borne acquitte de la bonne réception des scripts, le serveur efface ceux-ci.

Plus précisément, le démon reconnaît trois commandes : `BOOT`, `UPDATE` et `RESET`. La première permet d'obtenir les scripts correspondants à la séquence de boot tandis que la seconde permet d'obtenir les scripts permettant de mettre à jour la configuration de la borne. La dernière commande permet d'acquiescer la bonne réception des scripts précédents.

En pratique, si l'on prend l'exemple de la borne `valhalla`, ces scripts se trouvent réunis dans le répertoire `/etc/wifi/wifi-update/valhalla.wifi.crans.org`. Ceux commençant par 0 sont considérés comme des scripts de boot et ceux commençant par un autre chiffre sont considérés comme des scripts de mise à jour. Quand l'un ou l'autre de ces ensembles est demandé, les scripts correspondants sont triés par nom, concaténés ensemble et envoyés à la borne. Les scripts de mise à jour sont alors déplacés dans un répertoire spécifique et seront renvoyés pour chaque demande sauf si la borne émet une commande `RESET`, ce qui signifie qu'elle acquiesce la bonne réception du script précédent. Une session type est :

```
RESET
BOOT
RESET
UPDATE
RESET
UPDATE
```

En pratique, les bornes partageant un certain nombre de scripts en commun, ceux-ci sont placés dans `/etc/wifi/wifi-update/shared` et des liens symboliques sont utilisés.

IV.1.4.b Contenu des scripts

Au boot, un certain nombre d'actions sont entrepris par les scripts :

1. Le script `constants` initialise un certain nombre de constantes, comme le nom des interfaces, les IP de certaines machines, etc.

2. Le script `get_config` initialise une fonction permettant d'obtenir la configuration de la borne : son adresse IP, son adresse MAC, sa puissance et le canal qu'elle doit utiliser. Ce script est généré à partir de la base LDAP.
3. Le script `get_clients` initialise une fonction qui renvoie l'ensemble des clients WiFi avec leur nom, leur adresse IP et leur adresse MAC. Cette fonction est utilisée à plusieurs reprises plus loin.
4. Le script `update-config` permet de mettre à jour la configuration de la borne. Il permet par exemple de changer si besoin le canal de celle-ci.
5. Les scripts dans le répertoire `/etc/wifi/wifi-update/shared/cron` permettent de configurer le démon cron pour effectuer diverses actions. Par exemple, il fait remonter le nombre de clients connectés à la borne.
6. Le répertoire `/etc/wifi/wifi-update/shared/firewall` contient des scripts quit permettent de mettre en place le firewall de chacune des bornes. Celui-ci sera décrit dans la section IV.2.2 page 36.

Divers autres scripts sont disponibles. À la mise à jour, seulement les scripts nécessaires sont appelés. Par exemple, le script `update-config`. Il est à noter que certains scripts dépendent d'autres scripts. Ainsi, `update-config` dépend de `get_config`, il faut donc les ordonner correctement en les liants sous le nom `101get_config` et `102update-config` (par exemple).

La plupart des scripts sont prévus pour être lancés à la fois au boot et à la mise à jour.

∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘

IV.2 Les bornes

Les bornes WiFi sont des Linksys WRT54G. Ce sont des bornes très utilisées car Linksys a accepté d'en libérer le firmware (logiciel embarqué) sous GPL. Il est donc possible de modifier entièrement le fonctionnement de celles-ci. Le système d'exploitation utilisé est une version embarquée de Linux. Les possibilités sont donc infinies.

Chaque borne dispose d'une interface WiFi et de cinq interfaces filaires. Quatre d'entre elles sont configurées comme un switch. Dans notre configuration, la cinquième interface est désactivée.

Nous avons grandement modifié le firmware d'origine pour inclure un certain nombre de fonctionnalités supplémentaires et supprimer d'autres fonctionnalités qui nous étaient inutiles. Notamment :

– Un serveur SSH a été ajouté et nous sert à nous connecter sur la borne pour effectuer certaines opérations de maintenance. L'identification se fait uniquement à l'aide de clefs. La borne ne contient alors aucune donnée secrète et un vol n'apportera donc aucune information à l'attaquant.

- La partie cliente de l'application wifi-update décrite dans la section IV.1 a été ajoutée, ainsi qu'un certain nombre de démons que l'on décrira en section IV.2.1 : un proxy ARP, un proxy DHCP et un proxy DNS.
- Le serveur Web embarqué à l'origine n'est plus présent ainsi que la plupart des autres services, comme par exemple le serveur TFTP qui permettait de mettre à jour la borne à distance. Cela signifie que la seule interface d'administration restante est le serveur SSH et l'application wifi-update.
- Le noyau a été modifié pour inclure les possibilités de bridge filtrant à la borne (avec les patchs `bridge-nf`). De nombreux moteurs et cibles ont été rajoutés à netfilter. Ceci sera discuté plus en détail en section IV.2.2 page suivante.

IV.2.1 Les applications supplémentaires

Des applications spécifiques au Cr@ns ont été développées pour les bornes. Nous allons les décrire brièvement.

IV.2.1.a Wifi-update

Les bornes disposent de la partie cliente de l'application wifi-update tournant sur `Nectaris`. Cette partie cliente est donc chargée de demander au boot puis régulièrement les scripts de configuration. Cette application fait appel à la librairie MatrixSSL [5] qui est une implantation embarquée du protocole SSL. Les bornes disposent d'un certificat permettant de vérifier qu'elles discutent bien avec Nectaris.

Cette application est programmée en C et n'effectue aucune allocation dynamique : elle ne dispose donc pas de fuites mémoires qui risqueraient de la faire planter. Tous les cas d'erreur sont de plus testés et une erreur de communication entraîne rupture puis tentative de reconnexion. Les scripts qui prennent trop de temps à s'exécuter sont de plus arrêtés sauvagement. La sortie des scripts sont remontés via syslog à `Nectaris`.

La connexion est établie de manière permanente et le serveur est interrogé toutes les 30 secondes pour obtenir les mises à jour éventuelles.

IV.2.1.b Le proxy ARP

Afin de répondre aux exigences de sécurité concernant le déni de service, présentées en section III.3.4.c page 27, une application permettant de filtrer les requêtes ARP a été développée. Nous verrons en section IV.2.2 que le firewall ne laisse pas passer les requêtes ARP. Nous disposons à la place d'une application qui écoute sur chacune des interfaces et relaie les requêtes ARP d'une interface à l'autre après les avoir vérifiées. Elle dispose également d'une protection contre le flood.

Cette application lit la correspondance MAC/IP à partir d'un fichier envoyé par wifi-update à la borne. Elle est programmée en C et les allocations

dynamiques sont strictement contrôlées. Toutes les erreurs sont traitées au mieux. Cette application utilise directement les *Linux Socket Filter* (LSF) ce qui lui permet d'être particulièrement efficace et économe en mémoire.

IV.2.1.c Le proxy DHCP et DNS

L'intérêt de ces proxies est décrit en section III.3.4.d page 28.

Le proxy DHCP est en réalité d'une version modifiée du programme dhcrelay de la distribution ISC DHCP [3] en version 2.0pl5. Elle comprend les patchs de sécurité nécessaires. Ce proxy DHCP interroge directement `Nectaris`.

Le proxy DNS est le programme DNSmasq configuré pour toujours renvoyer l'adresse IP de `Nectaris`.

IV.2.2 Le firewall

Les bornes disposent en interne d'un firewall. Elles agissent comme un pont filtrant, c'est à dire comme un équipement de niveau 1 mais capable de comprendre les niveaux supérieurs. C'est une fonctionnalité rendue possible par le patch `bridge-nf` sur les noyaux Linux.

Par défaut, aucune requête n'est autorisée à traverser le pont. Au niveau 1, les actions suivantes sont menées :

1. tous les paquets multicast sont jetés,
2. tous les paquets non IP sont jetés,
3. pour les paquets IP restant, si la cible est `Nectaris`, l'adresse MAC est vérifiée.

Le fait de ne laisser passer que les paquets IP nous assurent que ceux-ci circuleront dans la table FORWARD de Netfilter et pourront donc être filtrés par celui-ci.

Au niveau 2, l'action par défaut est de tout jeter. Nous effectuons ensuite les actions suivantes :

1. la correspondance MAC-IP est vérifiée pour la source[1],
2. les paquets ESTABLISHED ou RELATED sont acceptés,
3. les communications IPsec entre un client et `Nectaris` sont acceptées,
4. les communications avec le démon ISAKMP de `Nectaris` sont acceptées,
5. toutes les communications web sont redirigées sur `Nectaris`,
6. toutes les communications DNS sont redirigées sur le serveur local.

Le firewall fait appel à certaines fonctions avancées comme le marquage des paquets avec l'extension `MARK`, la possibilité de choisir l'interface du

[1] Cette fonctionnalité utilise `ipset` qui permet de faire de manière très rapide cette correspondance MAC-IP.

bridge avec l'extension `physdev`, la possibilité de gérer des ensembles avec l'extension `ipset`.

Le DHCP est également filtré pour empêcher les serveurs DHCP pirates de répondre à la place de `Nectaris`. Le relai interroge `Nectaris` directement.

∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘ ∘

IV.3 Conclusion

La mise en œuvre technique s'appuie sur des logiciels éprouvés, largement diffusés et en partie audités. Elle s'appuie également sur des logiciels maison écrits avec la sécurité, la fiabilité et l'économie mémoire en tête. Les possibilités d'évolution sont vastes et il est possible d'inclure des services supplémentaires sans problèmes puisque nous maîtrisons intégralement la chaîne de développement.

Bibliographie

[1] Aircrack, a 802.11 sniffer and wep key cracker. http://www.cr0.net:8040/code/network/.

[2] Airsnort, a wireless lan encryption keys recovery. http://airsnort.shmoo.com/.

[3] Isc dynamic host configuration protocol. http://www.isc.org/sw/dhcp/.

[4] The kame project. http://www.kame.net.

[5] Matrixssl, an open source embedded ssl. http://twistedmatrix.com.

[6] Openbsd. free, functional and secure since 1995. http://www.openbsd.org.

[7] Python, an interpreted, interactive, object-oriented programming language. http://www.python.org.

[8] Twisted, an event-driven networking framework written in python. http://twistedmatrix.com.

[9] Dod standard : Transmission control protocol, 1980. http://www.faqs.org/rfcs/rfc761.html.

[10] Cédric Blancher. La sécurité des réseaux 802.11 : quoi de neuf depuis un an ? *MISC 12*, Mars 2004.

[11] Nikita Borisov, Ian Goldberg, and David Wagner. Intercepting mobile communications : The insecurity of 802.11. http://www.isaac.cs.berkeley.edu/isaac/wep-faq.html, 2001.

[12] D. Bruschi, A. Ornaghi, and E. Rosti. S-arp : a secure address resolution protocol. In *ACSAC '03 : Proceedings of the 19th Annual Computer Security Applications Conference*, page 66. IEEE Computer Society, 2003.

[13] Mitchell Burton. Channel overlap calculations for 802.11b networks. Technical report, Cirond Technologies Inc, Nov 2002.

[14] Nancy Cam-Winget, Tim Moore, Dorothy Stanley, and Jesse Walker. IEEE 802.11i overview. Technical report, IEEE, 2004.

[15] Vinton Cerf and Ed Cain. The dod internet architecture model. *Computer Networks*, pages 307–318, July 1983.

[16] B. Croft and J. Gilmore. Bootstrap protocol (bootp). IETF Networking Group RFC 951, September 1985.

[17] Joan Daemen and Vincent Rijmen. Aes proposal : Rijndael. `http://www.esat.kuleuven.ac.be/~rijmen/rijndael/`.

[18] John D. Day and Hubert Zimmermann. *The OSI reference model*, pages 38–44. IEEE Computer Society Press, 1995.

[19] Autorité de Régulation des Télécommunications. Évolution du régime d'autorisation pour les rlan à partir du 25 juillet 2003. `http://www.art-telecom.fr/publications/lignedir/evol-rlan-250703.pdf`.

[20] Niels Ferguson and Bruce Schneier. A cryptographic evaluation of IPsec. Technical report, Counterpane, 3031 Tisch Way, Suite 100PE, San Jose, CA 95128, USA, 2000.

[21] Scott R. Fluhrer, Itsik Mantin, and Adi Shamir. Weaknesses in the key scheduling algorithm of RC4. *Lecture Notes in Computer Science*, 2259 :1–24, 2001.

[22] Joshua Guttman, Amy Herzog, and Javier Thayer. Authentication and confidentiality via ipsec. In *ESORICS 2000*. The MITRE Corporation, Springer-Verlag, 2000.

[23] D. Harkins and D. Carrel. The internet key exchange (ike). IETF Networking Group RFC 2409, November 1998.

[24] Russ Housley and William Arbaugh. Security problems in 802.11-based networks. *Commun. ACM*, 46(5) :31–34, 2003.

[25] IEEE. *Part 11 : Wireless LAN Medium Access Control and Physical Layer specifications : Higher-Speed Physical Layer Extension in the 2.4 GHz Band*, supplement to ieee standard for information technology – telecommunications and information exchange between systems – local and metropolitan area networks – specific requirements edition, 2002. `http://standards.ieee.org/reading/ieee/std/lanman/802.11b-1999`.

[26] S. Kent and R. Atkinson. Ip authentication header (ah). IETF Networking Group RFC 2402, November 1998.

[27] S. Kent and R. Atkinson. Ip encapsulating security payload (esp). IETF Networking Group RFC 2406, November 1998.

[28] S. Kent and R. Atkinson. Security architecture for the internet protocol. IETF Networking Group RFC 2401, November 1998.

[29] D. Maughan, M. Schertler, M. Schneider, and J. Turner. Internet security association and key management protocol (isakmp). IETF Networking Group RFC 2408, November 1998.

[30] John C. Mitchell. Finite-state analysis of security protocols. In *Computer Aided Verification*, pages 71–76, 1998.

[31] P. Mockapetris. Domain names – concepts and facilities. IETF Networking Group RFC 1034, November 1987.

[32] Vebjørn Moen, Håvard Raddum, and Kjell J. Hole. Weaknesses in the temporal key hash of wpa. *SIGMOBILE Mob. Comput. Commun. Rev.*, 8(2) :76–83, 2004.

[33] Michael Ossmann. Wep : Dead again. In Security Focus, December 2004. `http://www.securityfocus.com/infocus/1814`.

[34] Radia Perlman and Charlie Kaufman. Key exchange in ipsec : Analysis of ike. *IEEE Internet Computing*, 4(6) :50–56, 2000.

[35] Bruce Schneier. Cryptanalysis of microsoft's point-to-point tunneling protocol (PPTP). In *ACM Conference on Computer and Communications Security*, pages 132–141, 1998.