Protocoles réseau : grandeur et décadence

Pierre Bétouin[1], Cédric Blancher[2], and Nicolas Fischbach[3]

[1] EADS/CCR - ESIEA
Paris - France
pierre.betouin@security-labs.org
http://unsigned.ath.cx/
[2] EADS/CCR
Paris - France
cedric.blancher@eads.net
http://sid.rstack.org/
[3] Colt Telecom
Zurich - Suisse
nico@securite.org
http ://www.securite.org/

Introduction

Les échanges qui régissent les systèmes d'information ne sont pas le fruit du hasard : ils reposent sur des schémas plus ou moins clairement établis afin de garantir la compatibilité entre les systèmes. Les *protocoles réseau*[4] définissent des standards qui permettent aux ordinateurs qui les respectent de communiquer.

Afin de faciliter la compatibilité des matériels, le *modèle OSI* propose une abstraction en 7 couches des différents services nécessaires au bon fonctionnement d'un réseau. Il existe également d'autres modèles comme le DoD (du nom du *Department of Defense* américain) qui ne contient que 4 couches. Toutefois, ces modèles sont théoriques et aucune implémentation ne les respecte scrupuleusement.

Les protocoles qui régissent les réseaux ont été définis depuis longtemps pour certains, à une époque où la sécurité n'était même pas envisagée. Pour combler cette lacune, une tendance actuelle est de mettre de la cryptographie partout. Mais cela constitue-t-il réellement une solution ? Si les attaques sur les applications, serveurs et clients, sont légion (les fameux *buffer overflows* par exemple) et permettent de prendre le contrôle d'un système, manipuler les communications entre des entités s'avère souvent tout aussi efficace, et pas nécessairement plus compliqué.

Dans cet article, nous présenterons les différents types de communications puis détaillerons les attaques envisageables, qu'elles soient passives ou actives.

En revanche, si nous aborderons la question de l'implémentation des protocoles, nous n'évoquerons pas les failles répertoriées dans les bases de vulnérabilités qui sont d'ordre applicatif et non intrinsèque.

[4] Dans la suite du document, nous utiliserons simplement le terme *protocole*.

Nous analyserons différentes attaques classées par couche du modèle OSI (physique, liaison de données, réseau), applicative comme la corruption DNS puis les attaques génériques de protocoles applicatifs.

1 Typologie des attaques sur les protocoles

Les protocoles ne sont que l'abstraction qui permet une communication, un échange d'informations. Les attaques visant les protocoles sont donc en réalité des attaques qui portent sur ces échanges. Les acteurs d'un échange sont donc 3 :

- la *source* qui initie l'échange ;
- le *flux d'information* constitué des données émises par l'une des parties ;
- la *destination*.

On notera l'importance du sens de l'échange, toujours d'une des parties vers l'autre. Il est par conséquent unidirectionnel. En revanche, une communication étant constituée de plusieurs échanges dans les 2 sens, elle apparaîtra à double sens.

Si la source d'un échange est toujours unique, un même message aura soit une unique destination (échange *one-to-one* appelé *unicast* en terminologie réseau, cf. fig. 1) soit plusieurs (échange *one-to-many*, *broadcast* et *multicast* en terminologie réseau, cf. fig. 2). Les acteurs d'une communication étant maintenant identifiés,

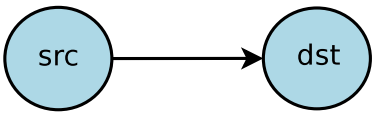

Fig. 1. Échange one to one

il est facile de déterminer où peuvent porter les attaques : partout ! En effet, chaque élément constitue une cible à part entière, par exemple :

- en fournissant de faux renseignements à la source, elle pourrait émettre des informations vers une mauvaise destination sans s'en rendre compte ;
- en bloquant le canal de communication, on pourrait empêcher la source et la destination de discuter ;
- en se faisant passer pour la source, on pourrait transmettre de fausses informations à la destination.

Chaque élément constitue donc une cible à part entière.

Néanmoins, on dégage 5 catégories d'attaques sur une communication :

- l'*interruption* (cf. fig. 3) : cette attaque vise la disponibilité, et se produit lorsqu'un des éléments du système est détruit (ex : virus attaquant le matériel) ou bloqué (ex : saturation de la bande passante) ;

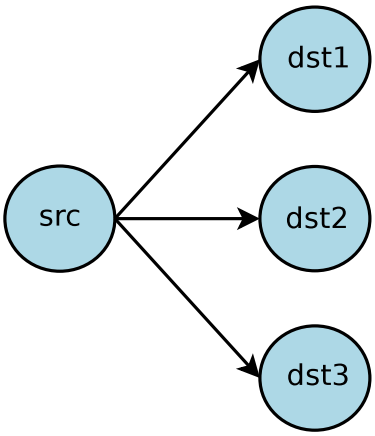

Fig. 2. Échange one to many

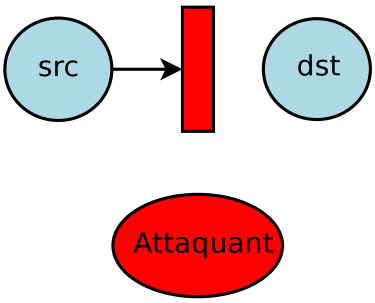

Fig. 3. Interruption : le flux d'information ne peut aller de la source à la destination

– la *capture* (cf. fig. 4) : cette attaque vise la confidentialité, et se produit lorsque l'attaquant parvient à accéder aux informations transférées lors de l'échange, par exemple en écoutant sur le réseau (*sniffing*), en modifiant le routage (RIP, OSPF, etc.) ou l'adressage (ARP ou DNS) ;

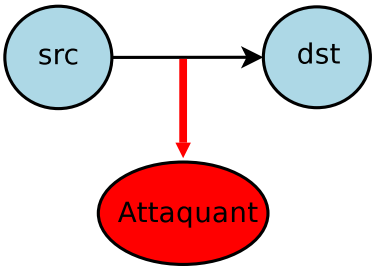

Fig. 4. Capture : l'attaquant obtient un accès illégitime au flux d'informations

– l'*injection* (cf. fig. 5) : cette attaque vise l'intégrité, et se produit lorsque l'attaquant insère des informations dans le flux sans que cela ne soit remarqué par la destination ;

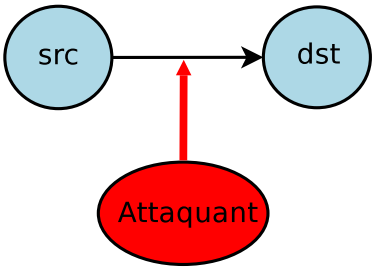

Fig. 5. Injection : l'attaquant injecte ses propres informations dans le flux légitime

– la *modification* (cf. fig. 6) : cette attaque vise la confidentialité et l'intégrité, et se produit lorsque l'attaquant intercepte et éventuellement modifie les informations en jouant le rôle de l'*homme du milieu* ;

Fig. 6. Modification : l'attaquant intercepte le flux émis par la source, et le modifie avant de le renvoyer à la destination

– l'*usurpation* (cf. fig. 7) : l'attaquant construit lui-même le flux d'informations, et l'envoie à la destination en se faisant passer pour la source légitime, à la base de toutes les formes de *spoofing* (usurpation d'identité) ou des attaques de rejeu.

Fig. 7. Usurpation : l'attaquant construit un flux d'informations et l'envoie à la destination en usurpant l'identité de la source

Les attaques menées à l'encontre des communications sont soit *passives*, soit *actives*. Lors d'une attaque passive, l'attaquant n'intervient pas dans la communication et se contente de recueillir les informations qui passent. Par exemple, l'écoute d'un réseau révèle souvent des informations sur les versions des logiciels utilisés ou des données confidentielles quand les échanges ne sont pas chiffrés. Ces attaques, si elles sont difficilement repérables, fournissent souvent de piètres résultats. À l'inverse, les attaques actives sont certes plus risquées, mais le gain potentiel est d'autant plus important. On distinguera essentiellement 3 attaques actives :

- attaque *directe* : l'attaquant se connecte directement à sa cible (ex : exploitation d'une faille sur un serveur vulnérable) ;
- attaque par *rebond* (*bounce*) : l'attaquant passe par un relais pour mener son attaque (ex : relais ouverts ou *zombies*) ;
- attaque *aveugle* (*blind*) : l'attaquant lance son action sans en voir les résultats, directement ou non.

Comme nous le verrons par la suite, ces schémas se combinent à merveille pour donner des attaques souvent efficaces.

2 Modèle OSI et Détournements

2.1 Couche 1

Les attaques visant ce niveau consistent en des opérations physiques visant à détourner le médium de communication de manière à créer la redirection de trafic.

Liens filaires L'attaque d'un lien filaire se fait par insertion physique. Ceci peut se faire de deux manières :

- rupture du câble et insertion par restauration de connectique ;
- insertion sur un point de connectique.

La seconde méthode est la plus simple à mettre en œuvre puisqu'elle ne suppose que la possession de cartes réseau et de connecteurs adaptés au médium à détourner. En revanche, elle suppose un accès physique à des zones particulières des locaux, comme une baie de brassage ou le point de connexion d'une des machines visées, ce qui peut s'avérer difficile en terme d'accessibilité et de discrétion.

La première méthode permet de viser n'importe quel point d'un lien de connexion puisqu'on va en quelque sorte créer de la connectique au milieu d'un câble. L'attaque d'un lien cuivré type RJ45 est simple : elle nécessite peu de matériel (pince à sertir et deux connecteurs mâle) et s'exécute très rapidement. Le dispositif inséré peut également être minuscule : on trouve aujourd'hui sur le marché des machines sous Linux à peine plus grosses qu'un connecteur RJ45 femelle... L'attaque d'un lien optique sera par contre un peu plus compliqué. Couper proprement une fibre et l'aligner sur le connecteur demande de l'habitude et du matériel spécialisé, mais reste tout à fait faisable.

Liens non-filaires La facilité de détournement physique d'un lien sans fil va dépendre de ses caractéristiques, et en particulier de son caractère directif. Un lien comme une communication laser à l'air libre est difficile à intercepter de manière furtive, ne serait-ce que parce que le dispositif réalisant le détournement doit se trouver sur le chemin du faisceau. Au contraire, un système peu directif comme un lien WiFi[5], même mis en place avec des antennes directives, est plus facile à attaquer.

Parmi tous les systèmes disponibles sur le marché, les plus faciles à attaquer sont ceux intégrant la notion de *roaming* (WiFi, GSM). L'interception consistera à réaliser une attaque du type *Man in the Middle*. Il faudra d'une part détourner la station « cliente » de son point d'accès légitime en présentant un signal plus fort et d'autre part se présenter vis-à-vis du point d'accès légitime comme une station. Cette attaque suppose également l'intégration de données de niveau 2 cohérentes pour que la station accepte de se connecter sur notre dispositif. En WiFi, cela se traduit par la présentation d'une configuration adaptée de part et d'autre (SSID, clé WEP, BSSID par exemple).

Actuellement, l'interception physique de liens WiFi est extrêmement simple. L'interception de liens GSM est possible, en particulier dans des zones de couverture GSM faible, par la mise en place d'une fausse station de base (BTS) de puissance suffisante. Un tel équipement est de la taille d'un sac à dos.

2.2 Couche 2

Depuis quelques années les commutateurs (*switches*) remplacent les simples répéteurs (*hubs*) sur les réseaux locaux. Ceux-ci ne permettent plus d'écouter simplement tout le trafic sur la partie locale du réseau et il faut employer différentes attaques pour pouvoir créer un déni de service, injecter des informations ou écouter : attaques ARP, VLAN « hoping », attaques xTP[6], etc.

Cisco Discovery Protocol (CDP) CDP est un protocole d'administration qui permet d'échanger des informations de configuration entre équipements : nom de l'équipement, port physique, adresse IP affectée à l'interface d'administration, version d'IOS ou de CatOS installée, etc. CDP fonctionne sur tout lien

[5] IEEE 802.11

[6] Protocoles de signalisation de niveau 2

supportant HDLC via des envois en multicast et ne se limite pas aux commutateurs (routeurs, points d'accès ou encore téléphones IP par exemple).

CDP ne permet pas directement de détourner du trafic mais donne des informations intéressantes sur la topologie du réseau et sur les équipements qui le composent, et permet dans certains cas d'influer sur la configuration dynamique de ports en fonction des équipements détectés (détection des téléphones IP par exemple), permettant l'accès à un attaquant à des réseaux normalement inaccessibles (VLAN VoIP par exemple). Un attaquant capable de générer des messages CDP peut donc influer sur sa vision de l'infrastructure en se présentant, selon ses besoins, comme un commutateur, un routeur ou encore un téléphone IP. Enfin, l'envoi en masse de messages CDP sur des versions logicielles anciennes entraine l'utilisation de toutes les ressources mémoire de l'équipement pout créer un déni de service. Un outil comme *cdp* de la suite IRPAS[8] permet de réaliser de telles attaques. Pour ces raisons, il est recommandé de désactiver CDP (sur tout l'équipement ou interface par interface). Cependant, l'expérience prouve que la documentation réseau n'est pas toujours aussi complète et à jour qu'il n'y paraît et que, parfois, CDP apparaît comme un sauveur...

Spanning Tree Protocol (STP) STP[7] et ses variantes comme RSTP (Rapid Spanning Tree Protocol) sont des mécanismes de détection de boucle dans le réseau (boucle physique ou boucle logique, par exemple par VLAN). Le port détecté et élu comme étant sur le chemin redondant passe en mode bloqué jusqu'à la détection d'un problème sur le pseudo-anneau. Comme d'habitude, STP est bien souvent activé par défaut sur tous les ports d'un commutateur et, autre caractéristique intéressante, durant le processus d'élection STP, les ports concernés sont inactif : il est impossible d'envoyer ou de recevoir des trames Ethernet. Différents dénis de service qui consistent donc à forcer des ré-elections à l'infini (racine éphémère, faux lien redondant, etc.) sont donc possibles. L'attaque la plus intéressante est l'interception mais, dans la majorité des cas, celle-ci requiert d'avoir une connexion physique à deux commutateurs. L'intrus va injecter des trames de manière à se faire passer pour la racine STP du réseau et ainsi déclencher des modifications de topologie pouvant conduire à des dénis de service (autres ports de la boucle en mode bloquant) ou des redirections de trafic. L'outil Yersinia[10] permet simplement injecter du trafic STP.

Pour éviter ce genre d'attaques, il faut désactiver l'acceptation des messages de type *Bridge PDU* sur les ports qui ne font pas partie de la boucle de couche 2 (liens entre commutateurs). Au niveau de l'architecture réseau il est souvent plus intéressant d'éviter autant que possible STP, car comprendre le comportement du réseau et localiser les fautes est un véritable casse-tête, et de favoriser la gestion de la redondance par des protocoles de routage (OSPF ou IS-IS par exemple).

Sur un réseau local d'entreprise, le détournement reposant sur STP se détecte assez rapidement : le réseau ne fonctionne plus ou devient très lent dans la mesure

[7] IEEE 802.1D

où la majorité des attaquants oublient que leur matériel n'est souvent pas capable de traiter le trafic détourné pour maintenir la bande passante nécessaire.

VLAN Un VLAN[8], pour *Virtual LAN*, est un domaine de *broadcast* Ethernet (i.e. branche Ethernet) logique. L'utilisation des VLANs permet de rationaliser l'utilisation des ports physiques des commutateurs et de rendre la gestion du réseau d'entreprise plus simple, puisque pour placer un utilisateur dans un réseau particulier, il suffit de placer le port physique auquel il est connecté dans le bon VLAN. Cette notion de VLAN peut-être exportée d'un commutateur à un autre en utilisant des liens spécialisés appelés *trunks*. Ces derniers véhiculent des trames Ethernet au format 802.1q : elles possèdent un champ supplémentaire indiquant à quel VLAN elles appartiennent.

Pour un attaquant, la possibilité, depuis un VLAN donné, d'envoyer du trafic dans un autre VLAN, de changer de VLAN ou, mieux, d'accéder à plusieurs VLANs choisis, présente un intérêt tout particulier, puisque cela lui permet *in fine* d'atteindre pratiquement toutes les machines du réseau sans passer par les points de routage et de filtrage mis en place entre les différents réseaux. En dehors des erreurs de configuration, différentes attaques existent. La plus efficace consiste à modifier la configuration de son port physique soit en attaquant le commutateur directement, soit en utilisant les protocoles DTP et VTP décrits par la suite. Une autre attaque couramment évoquée est la double encapsulation 802.1q qui consiste à positionner deux fois les champs relatifs aux VLANs dans la trame Ethernet. L'outil *Scapy*[11] permet d'injecter du trafic dans de telles trames et, lorsque l'attaque fonctionne, d'établir une voie de communication à sens unique vers le VLAN cible. On pourra s'en servir pour déclencher une attaque en corruption de cache ARP par exemple. Cependant, des conditions très particulières doivent être réunies, ce qui est très rarement le cas, en particulier si on suit les conseils de déploiement des équipementiers.

Dynamic Trunk Protocol (DTP) DTP permet de transformer dynamiquement un port en mode *hôte* en un port en mode *trunk* : le port physique ne fait donc plus partie du VLAN mais permet de transporter un ensemble de VLANs. Tous les ports d'un commutateur sont en mode DTP par défaut et le changement de l'état du port se fait par le simple envoi d'un message SNAP HDLC 0x2004 avec un outil comme [10] par exemple. Il est alors facile d'injecter des trames dans n'importe quel VLAN.

En général les ports en mode *trunk* sont clairements identifiés et il est donc inutile de laisser tous les ports en mode automatique : la meilleure solution consiste à désactiver DTP sur l'ensemble du commutateur et configurer les ports « trunks » de manière statique. Des versions logicielles récentes disposent d'une option qui permet de forcer un port en mode « hôte », ce qui présente l'avantage de désactiver toutes les fonctionnalités inutiles (VTP, DTP, STP, etc.).

[8] IEEE 802.1Q

VLAN Trunking Protocol (VTP) VTP permet une gestion centralisée des VLANs quand un ensemble de commutateurs sont interconnectés (via un *trunk*). Celui-ci facilite l'administration, car il n'est plus nécessaire de se connecter sur chacun d'eux pour créer un nouveau VLAN ou lui affecter des ports physiques, mais présente plusieurs failles de sécurité. La plus simple à exploiter est le déni de service : l'injection aléatoire de messages VTP (SNAP HDLC 0x2003) avec Yersinia[10] peut conduire au remplissage des ressources mémoire (TCAM par exemple) ou au placement des hôtes dans de mauvais VLANs. S'il est possible de passer un port physique en *trunk* (si DTP n'est pas désactivé sur ce port) il devient alors facile d'injecter un message VTP pour reconfigurer les VLANs et les ports qui leur sont affectés : l'attaquant peut aisément se débrouiller pour changer de VLAN et de créer de fausses boucles.

Pour se protéger, la meilleure solution est de désactiver VTP (en activant le mode transparent) et se connecter sur les différents commutateurs pour modifier la configuration. Un mot de passe permet de protéger son domaine VTP mais encore faut-il que celui choisi ne soit pas faible.

Hot Standby Routing Protocol (HSRP) et Virtual Router Redundancy Protocol (VRRP) HSRP et VRRP[9] apportent un mécanisme de redondance du prochain saut : un ensemble d'équipements (deux routeurs ou deux pare-feux par exemple) partagent une adresse IP et une adresse MAC virtuelles, l'un d'eux étant maître, les autres esclaves.

Pour HSRP, par exemple, cette adresse MAC, si elle n'est pas modifiée, est de la forme 00-00-0c-07-ac-¡ID¿, où ID représente l'identifiant du groupe HSRP. Elle est donc facilement identifiable, ainsi que l'adresse IP associée et en définitive, on parvient rapidement à identifier les routeurs membres. Une attaque naïve consiste d'abord à intégrer le groupe, puis à bloquer successivement les autres routeurs jusqu'à ce que l'on devienne le maître. Or si l'attaquant est capable d'intégrer le groupe, il lui est possible d'injecter des messages pour devenir le maître et placer les autres équipements en standby simplement en utilisant le protocole. C'est ce que permet par exemple outil comme *hsrp* de la suite *IRPAS*[8]. Dès lors, l'ensemble du trafic destiné à l'adresse IP du groupe HSRP est redirigé vers le pirate.

L'accès aux groupes peut être sécurisé. Pour HSRP, il s'agit d'un simple mot de passe. Pour VRRP, on pourra utiliser un mot de passe ou une somme de type HMAC MD5.

2.3 Address Resolution Protocol (ARP)

Sur les supports multicast, on se trouve confronté à deux espaces d'adressage. On a d'une part l'adressage MAC de niveau 2 (Ethernet, Token Ring, etc.) et d'autre part l'adressage de niveau 3, à savoir IPv4 ou IPv6. Si deux machines veulent communiquer entre elles, elles doivent disposer d'un mécanisme

[9] RFC 2338

de résolution d'adresses permettant d'associer à une adresse IP l'adresse MAC correspondante. Le protocole permettant de réaliser cette liaison entre la couche 2 et la couche 3 est ARP[10].

Tel qu'implémenté dans les systèmes d'exploitation, ARP repose sur deux sous-systèmes :

- un sous-système de gestion du trafic ARP, permettant l'envoi et la réception de messages, ainsi que l'interprétation de ces derniers ;
- un sous-système de cache alimenté par les informations retournées par le sous-système précédent.

Le cache ARP constitue ainsi une base d'association MAC/IP pour les réseaux physiquement attachés à la machine. La corruption de cette base de données conduirait donc à de fausses associations MAC/IP. Si nous prenons une IP donnée, son association à une mauvaise adresse MAC conduit à une redirection de tout le trafic qui lui est adressé. Pour parvenir à compromettre le cache ARP d'une machine, nous lui envoyons des messages ARP corrompus contenant les associations que nous désirons injecter.

Les informations mises en cache sont tirées de certains champs de l'entête ARP. Pour une réponse, ces champs contiennent l'adresse IP demandée et l'adresse MAC correspondante. Pour une requête, il s'agit de l'adresse MAC et de l'adresse IP de la machine demandeuse. Dans la mesure où les flux de communication impliquent très souvent des réponses, ces informations sont mises en cache pour éviter la génération d'une requête ARP nécessaire à l'envoi de paquets IP vers cette machine. La RFC 826 définit un comportement très opportuniste de ce cache, ce qui a pour conséquence directe de rendre sa corruption d'autant plus simple.

Pour corrompre le cache ARP, l'attaquant va donc soit émettre des réponses, comme la plupart des outils classiques de corruption de cache ARP le permettent, soit des requêtes, avec des outils plus avancés. Une étude du fonctionnement d'un cache ARP montre en effet que la corruption par génération de requêtes est plus efficace et plus furtive. On pourra utiliser le générateur de paquets ARP *arp-sk*[7] pour générer ce type de messages :

```
root@joker:~# arp-sk -w -d batman -S robin -D batman
```

Cette commande permet à la machine locale (Joker) de forger une requête ARP en provenance d'une machine qui aurait l'adresse MAC de Joker et l'adresse IP de Robin à destination de Batman. À réception d'une telle requête, Batman va d'abord mettre en cache l'association présentée (ou la mettre à jour si elle existe déjà) et répondre. Il s'en suivra que dorénavant, à chaque fois que Batman voudra envoyer un paquet IP à destination de l'IP de Robin, celui-ci sera encapsulé dans une trame Ethernet à destination de Joker. Tout le trafic émis par Batman à destination de Robin se trouve donc détourné. En répétant l'opération vis-à-vis de Robin, Joker détourne l'ensemble des communications entre Robin et Batman :

[10] RFC 826

```
root@joker:~# arp-sk -w -d robin -S batman -D robin
```

Il ne lui reste plus qu'à mettre en place un simple routage de paquets pour que la redirection soit transparente. L'attaque complète est décrite en figure 8.

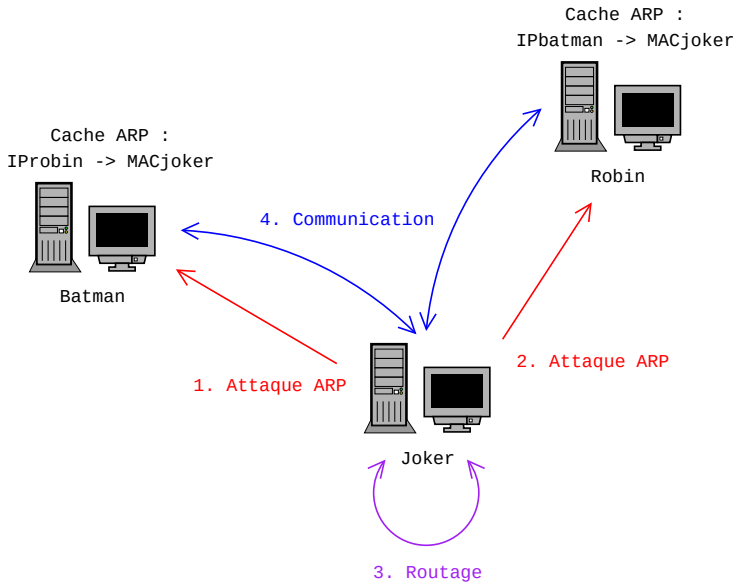

Fig. 8. Corruption de cache ARP (*ARP cache poisoning*)

La corruption de cache ARP[1] est une attaque facile à mettre en œuvre et extrêmement efficace. La meilleure méthode pour s'en protéger reste la segmentation en sous-réseaux.

2.4 Couche 3

Configuration automatique : Dynamic Host Cconfiguration Protocol (DHCP) Abuser un protocole de configuration automatique comme DHCP[11] permet à un attaquant d'imposer une configuration IP arbitraire à une ou plusieures cibles. Ainsi, il peut choisir de rediriger le trafic externe vers lui en fournissant son IP comme passerelle par défaut ou fausser la résolution de noms (DNS) en se plaçant comme serveur DNS. Pour réaliser cette attaque, l'intrus mettra en place un faux serveur DHCP sur le réseau qui répondra aux requêtes émises avant le serveur légitime. Il pourra en outre émettre des messages factices d'expiration de bail pour forcer des hôtes à redemander une configuration.

[11] RFC 2131

Pour gérer la concurrence vis-à-vis du serveur DHCP légitime, on pariera sur notre vitesse de réaction, ou on mettra en place un déni de service. La méthode la plus efficace contre les serveurs DHCP est une attaque en inondation qui consiste à compléter des demandes de configuration jusqu'à ce qu'il ne puisse plus répondre, soit parce qu'il ne dispose plus d'adresse libre dans les pools qu'il contrôle, soit parce qu'il a dépassé sa capacité de gestion.

Le routage dynamique Les protocoles de routage sont intéressants du point de vue de la sécurité. En effet, le chemin que va suivre un paquet dans le réseau, à l'échelle de l'entreprise comme à l'échelle mondiale, est directement lié à des informations de routage qui non seulement ne sont pas gérées par les parties qui communiquent, mais leur sont aussi le plus souvent invisibles. Ces protocoles sont donc des éléments clés qu'un attaquant voudra perturber de manière à rediriger des communications.

- Les redirections ICMP[12] ne constituent pas, à proprement parlé, un protocole de routage, mais permettent de modifier dynamiquement la table de routage d'un hôte qui les accepterait. Au sein d'un même sous-réseau IP, ils servent à indiquer à une machine qu'elle n'utilise pas le bon routeur et, évidemment, lui fournir l'adresse du meilleur routeur. L'injection de messages ICMP de type Redirect permettra donc à un attaquant de créer une redirection de trafic. Même si beaucoup de systèmes acceptent ces messages par défaut, il est sain de les ignorer. En effet, ce n'est pas un mécanisme normal de routage : de tels messages ne devraient donc pas exister sur un réseau bien conçu.
- Interior Gateway Routing Protocol (IGRP) est un protocole de routage propriétaire défini par Cisco que l'on trouve encore sur certains réseaux et qui ne présente aucun élément de sécurité. Il peut donc très simplement être contourné, la seul véritable contrainte pour un attaquant étant de donner aux routes qu'il injecte un poids (métrique) inférieur à celui des routes en place. Il est donc trivial pour un attaquant de créer des boucles (déni de service) ou de détourner du trafic, avec un outil comme *igrp* de la suite *IRPAS*[8].
- Routing Information Protocol[13] (RIP) est un protocole de routage qui n'est pratiquement plus utilisé aujourd'hui, ce qui est heureux du point de vue de la sécurité. En effet, aucune des deux versions de ce protocole ne présente d'authentification sûre des messages : RIPv1 n'est pas authentifié et RIPv2[14] transporte le mot de passe en clair. Son fonctionnement étant proche de celui de IGRP, il est vulnérable aux mêmes attaques, même si les boucles sont plus difficiles à créer grâce à un algorithme appelé *split horizon with poisoned reverse* destiné à éliminer les routes concurrentes.

[12] RFC 792
[13] RFC 1058
[14] RFC 2453

- Open Shortest Path First[15] (OSPF) est un protocole de routage interne qui opère en multicast. S'il est relativement simple d'injecter des annonces (Link State Announcement ou LSAs) sur un LAN, parvenir à conserver une route ainsi créée est complexe. En effet, comme cette information est reçue par tous les routeurs de l'aire OSPF, un routeur qui verrait passer un LSA contradictoire avec sa vision de l'état du réseau répondrait immédiatement avec un LSA correct, contestant et annulant la route injectée. Une technique efficace consisterait à détourner les annonces, par corruption de cache ARP par exemple, et les modifier à la volée entre deux routeurs, ce qui reste relativement difficile. Cependant, réaliser un déni de service en multipliant les messages injectés est relativement facile. On notera qu'il est possible d'authentifier les échanges par somme MD5 et de restreindre la diffusion des annonces à ses seuls voisins OSPF déclarés, mais ces mesures sont rarement déployées.
- Border Gateway Protocol[16] (BGP-4) est le protocole de routage déployé entre systèmes autonomes (AS) sur Internet. Il est également déployé sur des réseaux d'entreprises lorsque ceux-ci atteignent une taille conséquente. Une session BGP est une connexion TCP entre deux routeurs qui servira à échanger les informations de routage (préfixes réseaux, numéros d'AS et caractéristiques associées). Les attaques les plus connues sont des dénis de service : envoi de masse de TCP SYN, envoi de messages ICMP de type « unreachable », de TCP RST, etc. L'injection de messages BGP UPDATE est très complexe, voire quasi-impossible si la session TCP est protégée par un condensat MD5 commun aux deux routeurs, à moins d'être en position de capturer le trafic BGP visé, ce qui est évidemment assez difficile. Finalement, le plus simple consistera à s'attaquer directement aux routeurs composant l'infrastructure (login/password, faille, mauvais filtrage, etc.) et s'en servir pour injecter des informations.
- Multi-Protocol Label Switching[17] (MPLS) est plus un protocole de couche 2 que de couche 3, mais il est essentiellement utilisé dans un contexte WAN, en parallèle des protocoles de routage. Dans un réseau MPLS un datagramme porte un label. Il n'est pas routé à chaque saut, mais commuté en fonction de son label par examen d'une table des labels. Ainsi, un paquet traversant un nuage MPLS en sortira comme s'il n'avait traversé qu'un seul routeur. L'injection de trafic MPLS (paquet marqué avec un label) n'est possible que depuis un élément réseau appartenant au nuage MPLS. En effet, les interfaces non-MPLS n'acceptent pas de paquets marqués. Comme pour BGP, il est plus intéressant de s'attaquer à l'infrastructure que de tenter d'injecter des informations en aveugle. Il peut également être pertinent de s'attaquer aux protocoles de routage pour influencer le marquage des paquets par le nuage que de vouloir directement jouer avec

[15] RFC 2328
[16] RFC 1771
[17] RFC 3031

les labels et le protocole de distribution de label LDP (Label Distribution Protocol).

– Optimized State Link Routing[18] (OSLR) est un protocole de routage utilisé dans les réseaux maillés ou *mesh networks*. Sur ce type de réseau, chaque participant est un nœud diffusant des informations de routage et est donc susceptible de provoquer des changements de topologie. OSLR est particulièrement déployé sur les réseaux sans-fil mobiles ad-hoc (*Mobile Ad-hoc NETwork* ou MANET) pour assurer la bonne gestion du routage et sa diffusion au sein du nuage. En général communautaires, ces réseaux ne disposent souvent d'aucune mesure visant à sécuriser l'infrastructure qui est alors vulnérable à de nombreuses attaques en redirection de trafic par reconfiguration depuis un noeud du réseau, voire même un réseau externe connecté. Dans la mesure où la simple insertion d'un nouveau nœud va entraîner une reconfiguration, il n'est même pas toujours nécessaire d'émettre des messages corrompus pour obtenir les effets désirés, en particulier si cette insertion est combinée à une forte puissance et une bonne sensibilité permettant de couvrir une grosse partie du réseau et d'en devenir naturellement un nœud central, comme illustré en figure 9.

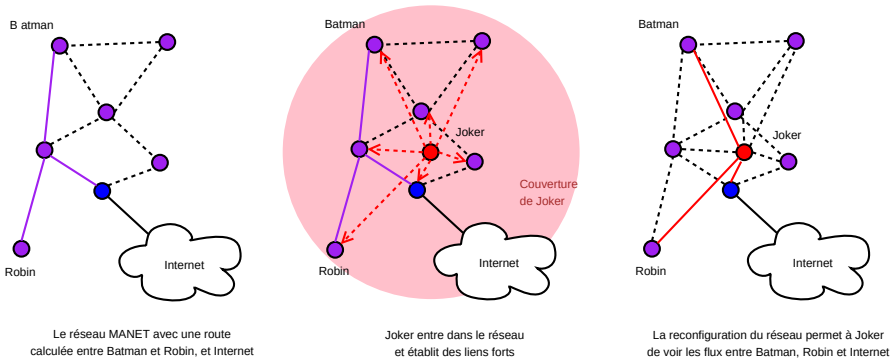

Fig. 9. Reconfiguration de nuage OSLR par insertion

L'injection de fauses annonces est également possible pour détourner des liens. Une méthode relativement efficace sur un MANET consiste à modifier la qualité des liens dans les annonces injectées pour favoriser certains chemins par rapport à d'autres de même métrique dans le calcul de l'arbre de routage. Les attaques en déni de service sont également possibles, soit par création de boucles par reconfiguration, soit par écroulement par surcharge des agents OSLR. Cette dernière attaque se fait en rejouant de nombreux messages OSLR vers plusieurs nœuds pour entraîner l'écroulement du réseau par consommation des ressources dédiées à la gestion du rou-

[18] RFC 3626

tage. Les participants à un réseau MANET communautaire doivent donc mettre eux-mêmes en œuvre les moyens de protection de leurs flux IP (IPSEC, SSL, etc.). La section 20, *Security considerations*, de la RFC 3626[5] qui définit ce protocole détaille ces problèmes et propose des solutions. Au demeurant, même sécurisé, un réseau OSLR reste très sensible aux dénis de service.

3 Protocoles applicatifs

Les détournements applicatifs consistent à agir sur un protocole applicatif de manière à entrainer une perturbation du trafic. La corruption de la résolution de noms en est l'exemple le plus parlant.

3.1 Domain Name System (DNS)

DNS[19] est, avec BGP-4 et la caféine, un élément clé de l'infrastructure Internet. Il représente donc une cible de choix dans les attaques de détournement de flux de haut niveau et peut facilement devenir le maillon faible d'un échange. Son rôle est d'ailleurs au centre de notre problématique car une réponse illégitime à une requête entraîne directement la redirection du trafic.

La majeure partie des échanges transite sur UDP/53. TCP/53 est utilisé pour les réponses de plus de 512 octets, comme les transferts de zones par exemple. L'utilisation d'UDP, bien que justifiée par des raisons évidentes de performances, rend ce protocole vulnérable au *spoofing*. De plus, l'authentification et la correspondance requête/réponse est exclusivement basée sur le DNS ID, champ de 2 octets contenu dans les paquets DNS : la sécurité du protocole repose donc uniquement sur une clé de 16 bits ! Cette seule considération laisse perplexe...

Le détournement et l'altération de ce trafic peut conduire à :

- *Une exploitation directe* dans le but d'entreprendre un détournement immédiat ;
- *Une exploitation indirecte*, généralement effectuée après un détournement passif, afin d'élaborer des attaques plus évoluées (*phishing* par ex.) et généralement ciblées.

Nous aborderons les attaques[2] par prédiction des DNS ID puis les attaques en corruption de cache côté serveur (*DNS cache poisoning*).

Prédiction de DNS ID Cette attaque vise essentiellement les clients DNS (cf. fig. 10). Mais comme les serveurs DNS se comportent comme des clients vis-à-vis d'autres serveurs quand ils ne possèdent pas l'information demandée en cache, nous pouvons entrevoir que les faiblesses du protocole côté client peuvent directement se répercuter côté serveur. Il sera donc possible d'envisager des attaques de *DNS ID Spoofing* sur une communication serveur à serveur.

[19] RFC 1035

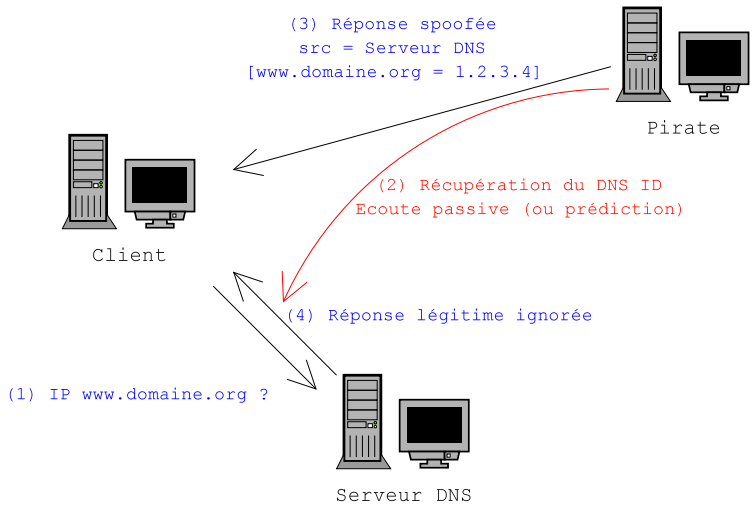

Fig. 10. DNS ID spoofing

Dans le cas général, cette attaque peut être menée de 2 façons différentes, selon les informations dont nous disposons :

- **DNS ID connu**
  S'il est possible d'écouter le trafic réseau et ainsi être en mesure de récupérer les DNS IDs, l'attaquant forgera un paquet DNS avec comme IP source celle du serveur, et comme IP de destination, celle du client ayant fait la requête.Le DNS ID sera évidemment identique à celui récupéré sur le réseau dans la requête.
  Le client recevra la réponse forgée et la traitera sans pouvoir s'apercevoir de la supercherie. La principale contrainte est de répondre avant le serveur DNS légitime puisque seule la première réponse sera considérée par le client, ce qui peut se révéler aléatoire si l'attaquant et la victime ne sont pas dans le même sous-réseau.
  dnsa peut être utilisé dans ce cas de figure avec les arguments suivants :
  ```
  ./dnsa -1 -i eth0                                    \
            -D www.mabaqueonline.com                   \
            -S 100.102.103.104                         \
            -s victime
  ```
- **DNS ID inconnu**
  Plusieurs solutions sont alors envisageables :
  - Les attaques par force brute sont presque utopiques. Les chances de réussite sont très réduites, la valeur de 16 bits devant être trouvée avant que la réponse légitime ne parvienne au client...
  - Lorsque l'on fait face à un générateur d'aléa faible, la fenêtre de valeurs peut devenir suffisamment petite pour pouvoir, en considérant

le paradoxe des anniversaires (cf. `http://www.x5.net/faqs/crypto/q95.html`), trouver le DNS ID en très peu de temps. Cependant, les générateurs aujourd'hui utilisés tendent à rendre obsolête ce type d'attaques...
- Certains OS incrémentent simplement le DNS ID à chaque requête d'une valeur fixe (souvenez-vous l'attaque de Mitnick sur les numéros de séquence !). La manipulation est alors triviale : par des techniques de *phishing*, on amène la victime à effectuer quelques requêtes sur un serveur DNS contrôlé, permettant la récupération du DNS ID et de l'incrément, de manière à pouvoir prévoir la valeur des prochains ID utilisés.

Corruption de cache DNS Les attaques par corruption de cache DNS étaient de moins en moins fréquentes car elles n'exploitent pas une faille intrinsèque au protocole mais généralement des failles d'implémentation : *BIND*, le serveur DNS *Microsoft*, ou plus récemment des produits *Symantec* en ont fait les frais. Plusieurs FAI et entreprises ont également été victimes de ces attaques, leur rappelant le rôle crucial de la résolution de noms au sein de leurs infrastructures. La diversification des serveurs a également permis de limiter les dommages occasionnés par de telles attaques. Les *root servers* étaient encore tous identiques il n'y a pas si longtemps ! Il est cependant intéressant de remarquer un récent regain d'intérêt pour ces attaques à des fins de *phishing*.

La corruption de cache DNS peut être menée de différentes façons :
- Les paquets DNS contiennent un champ *additional record* qui permet de renvoyer des informations complémentaires aux clients, comme les adresses IP associés aux noms fournis dans le champ NS.
  Prenons l'exemple d'une requête de type A pour l'adresse IP de la machine *www.domaine.com*. Le serveur DNS du domaine *domaine.com* va renvoyer l'adresse de la machine *www*, et peut, s'il est configuré pour le faire, donner plus d'informations sur son domaine. Ainsi, si ce champ permet de compléter une requête concernant le même domaine, il ne doit pas (en théorie !) être pris en compte pour des domaines différents. Dans le cas contraire, des informations relatives à un autre domaine ajoutées à une réponse légitime se verront mises dans le cache, permettant à un attaquant capable de fournir des informations inexactes de le corrompre.
  Cette attaque triviale est généralement mise en œuvre en utilisant un serveur DNS malicieux, tel que *dnsa*, sur un domaine contrôlé. Ses réponses comporteront alors toutes un champ *additional record* associant le nom à détourner à l'IP vers laquelle doivent être redirigées les requêtes. La figure 11 détaille cette attaque.
  Typiquement, dans ce cas de figure, Pirate lancera *dnsa*[6] sur la machine *dns.fakeone.org* avec les arguments suivants :

```
./dnsa -3 -i eth0                                    \
          -D www.fakeone.org                         \
          -S 100.102.103.104                         \
          -s dns.victime.org                         \
```

```
-a www.mabanqueonline.com                    \
-b 1.2.3.4
```

Ceci permet à l'attaquant, en interrogeant le serveur dns.victime.org sur une entrée du domaine fakeone.org qu'il contrôle, de placer dans le cache une fausse IP (1.2.3.4) pour www.mabanqueonline.com.

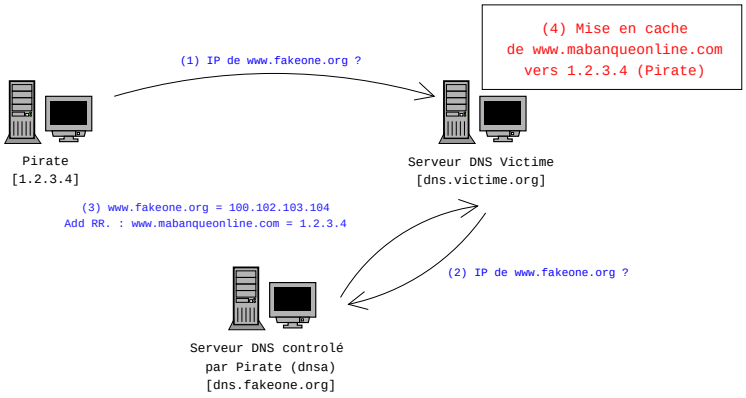

Fig. 11. Corruption de cache DNS (*DNS cache poisoning*)

Ainsi, un client qui demanderait l'IP de www.mabanqueonline.com au serveur dns.victime.org se verrait fournir l'IP du pirate, à savoir 1.2.3.4, et se connecterait donc à un faux site.

– Une attaque par *DNS ID Spoofing* décrite précédemment est également possible quand le serveur ne possède pas en cache l'adresse IP correspondant à l'adresse demandée. Non seulement l'attaquant parviendra à fournir une résolution corrompue au client, mais amènera le serveur à placer celle-ci dans son cache.

On notera que les systèmes Windows 2000 et XP intègrent un cache client activé par défaut susceptible d'être corrumpu.

3.2 Autres exemples

Les protocoles les plus intéressants à étudier de ce point de vue sont ceux qui mettent en œuvre plusieurs flux de données dont les paramètres sont négociés dans une connexion de contrôle, parfois avec un hôte tiers. C'est le cas en particulier de la voix sur IP[3] (VoIP). Lorsqu'un poste veut en joindre un autre, il se connecte à un serveur central appelé Call Manager[4] qui va se charger d'interpréter sa requête, localiser son correspondant et négocier avec les deux parties l'établissement de flux de voix directement entre les deux postes. Le détournement de ces connexions de contrôle à destination du Call Manager permet de parvenir à la redirection des flux de voix.

4 Attaques post-redirection

4.1 Manipulation de trafic

Lorsqu'un attaquant parvient à détourner des flux entre deux hôtes, il va pouvoir intercepter le trafic échangé pour écouter, rediriger, filtrer, bloquer ou modifier les flux au niveau réseau en fonction du protocole cible. La démarche est similaire quels que soient les protocoles concernés (HTTP, SMTP, etc.).

En voici quelques utilisations avec *Netfilter/iptables* :

- **Écoute passive de trafic**
  ```
  iptables -I FORWARD -i eth0 -o eth0 -j ACCEPT
  echo 0 > /proc/sys/net/ipv4/conf/eth0/send_redirects
  iptables -t mangle -A FORWARD -j TTL --ttl-inc 1
  ```
  Ces commandes permettent à l'attaquant de router le trafic détourner. Pour gagner en furtivité, il interdit l'envoi d'erreurs de type ICMP Redirect et repositionne la valeur de TTL.
- **Redirection de trafic HTTP**
  ```
  iptables -t nat -A PREROUTING -p tcp -s bob -d \
           alice --destination-port 80 \
          -j REDIRECT --to-ports 80
  ```
  Cette commande permet à l'intrus de rediriger les flux HTTP émis par Bob vers Alice vers le port TCP/80 local. Ainsi, s'il dispose d'un proxy ou d'un serveur HTTP, il pourra fournir un contenu détourné qui paraîtra issu du site d'Alice.
- **Déni de service** entre Bob et Alice :
  ```
  iptables -A FORWARD -s bob -d alice -j DROP
  iptables -A FORWARD -s alice -d bob -j DROP
  ```
  Avec ces commandes, on interdit toute communication entre Alice et Bob. Les dénis de service peuvent cependant être plus sélectifs et participer au déroulement d'une attaque de plus grande envergure. Un exemple concret d'attaque de ce type est l'exploitation d'un *fall back* IPSec présent par défaut sur d'anciens *service packs* Windows 2000 : si la machine n'arrive pas à communiquer sur le port UDP/500 (ISAKMP), alors la communication est établie en clair sans se soucier d'avantage d'IPSec pour maintenir le service !
  ```
  iptables -A FORWARD -s bob --destination-port 500 -j DROP
  iptables -A FORWARD -s alice --destination-port 500 -j DROP
  ```
  Ces deux commandes permettent la neutralisation des flux ISAKMP.

4.2 Interception de protocoles chiffrés

Une autre application du détournement de trafic est l'attaque des flux chiffrés par insertion, dite *Man in the Middle*. Une session chiffrée est négociée par l'attaquant avec chacun des participants qui croit s'adresser à l'autre. Il est alors capable de lire le contenu du flux, chaque intervenant chiffrant ses données pour lui. Ce type d'attaques suppose cependant que les parties visées ne soient pas

capables de s'authentifier de manière sûre. C'est particulièrement le cas lorsque l'on s'attaque à un flux HTTPS établi par une personne peu attentive qui ne tiendra pas compte de l'avertissement de sécurité émis par son navigateur (sans parler des failles qui permettent de ne pas déclencher d'avertissement)...

4.3 Generic Routing Encapsulation (GRE) et autres tunnels

GRE[20] et les différents tunnels (IP sur ICMP, IP sur DNS, PPP sur HTTP[s], etc.) sont souvent utilisés comme mécanismes de transport et d'encapsulation dans le cadre de détournements de trafic. Des outils (comme *tunnelx*[9]) acceptent le trafic encapsulé avec GRE, permettent un traitement local ainsi qu'une réinjection du trafic soit directement, soit via un second tunnel GRE. L'export de flux détournés devient alors un jeu d'enfant. Les différents mécanismes de trans-

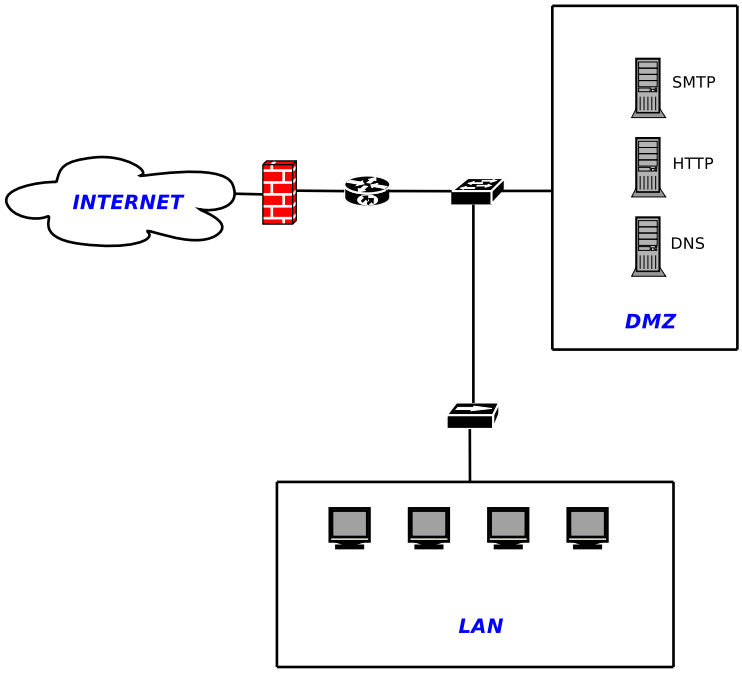

Fig. 12. Architecture attaquée

port disponibles s'apparentent pour la plupart à des canaux cachés et peuvent poser des problèmes : MTU, performances (en particulier dans le cas de transport de protocoles non-connectés sur des protocoles connectés, comme UDP sur TCP), etc.

[20] RFC 2784

5 Exemple de scénario

Nous avons pu voir de nombreuses méthodes pour détourner un flux réseau et agir dessus. Or ces techniques peuvent être combinées dans le cadre d'attaques ciblées de plus grande ampleur. C'est, en guise de conclusion, ce que ce scénario vise à illustrer, sur l'architecture décrite en figure 12.

1. Un serveur HTTP se fait compromettre dans la DMZ. Ce type d'évènement est relativement courant aujourd'hui avec l'utilisation de sites dynamiques prêts à l'emploi présentant des failles permettant l'exécution de commandes arbitraitres et/ou l'upload de fichiers. Il ne reste plus à l'attaquant qu'à trouver une faille locale qui lui permettra d'obtenir les droits super-utilisateur.
2. Une attaque en corruption de cache ARP entre le pare-feu et le serveur DNS situé dans la même DMZ lui permet de répondre aussi bien aux requêtes émises depuis le LAN qu'aux requêtes émises par le serveur DNS lui-même à destination d'Internet, ce qui le met en position de corrompre tout le flux DNS.
3. L'attaquant met en place la corruption des réponses DNS, soit par ID Spoofing via *dnsa*, ce qui lui permet de fournir directement des réponses falsifiées, soit par DNS cache poisoning, toujours avec *dnsa*.
4. Grâce aux informations fournies aux clients, il redirige certaines requêtes HTTP vers un site compromis permettant l'exécution d'un programme malicieux de type cheval de Troie sur un navigateur vulnérable.
5. Une fois le cheval de Troie activé, l'attaquant dispose d'un moyen d'attaque directement sur le réseau interne.

Références

1. C. Blancher, E. Detoisien, F. Raynal, *Jouer avec le protocole ARP*, MISC - Le journal de la sécurité informatique, Numéro 3, juillet 2002, `http://www.miscmag.com/`
2. P. Bétouin, *Failles intrinsèques du protocole DNS*, juillet 2004, `http://securitech.homeunix.org/dnsa/`
3. N. Bareil, *Projet ILTY : I'm listening to You (via VoIP) !*, Actes de la conférences SSTIC 2005, pp. 121–132, juin 2005.
4. P. Bétouin, P. Chambet, Nicolas Fischbach, *Sécurité de la Voix sur IP*, MISC - Le journal de la sécurité informatique, Numéro 16, novembre 2004, `http://www.miscmag.com/`
5. T. H. Clausen, P. Jacquet, *RFC 3626 - Optimized Link State Routing Protocol*, octobre 2003.
6. P. Bétouin, *DNSA*, juillet 2004, `http://securitech.homeunix.org/dnsa/dnsa-current.tar.gz`
7. F. Raynal, *Arp-sk*, juillet 2002, `http://www.arp-sk.org/`
8. Phenoelit, *IRPAS*, mars 2001, `http://www.phenoelit.de/irpas/`

9. Gaius, *Tunnelx*, Journal Phrack Numéro 56, mai 2000, `http://www.phrack.org/show.php?p=56&a=10`
10. D. B. Berrueta, A. A. Omella, *Yersinia*, avril 2005, `http://yersinia.sourceforge.net/`
11. P. Biondi, *Scapy*, octobre 2002, `http://www.cartel-securite.fr/pbiondi/scapy.html`