Source address validation improvements (SAVI): mécanismes de prévention contre l'usurpation d'adresses IP source

Jean-Michel Combes, Maryline Laurent

Abstract. Ten years ago, with the huge increase of attacks using spoofed source IP addresses, IETF standardized the "Network Ingress Filtering" method as well as its dynamical variant, well-known as uRPF, to mitigate these attacks. Unfortunately, from the deployment feedback, this method showed its limits. Then, IETF decided to standardize, inside the Source Address Validation Improvements (SAVI) working group, complementary mechanisms increasing the precision of spoofed source IP addresses detection inside IP flows. These mechanisms are mainly based on the monitoring of IP addresses allocation/assignment signalling (e.g., DHCP, IPv6 autoconfiguration). In this paper, first, we recall attacks using spoofed source IP addresses. Then, we present the "Network Ingress Filtering" method, and uRPF, and the limits of this method. In the next part, we describe the different currently standardized SAVI solutions. Then, we present the integration of SAVI in the ADSL/Fiber IPv6 access network, as concrete example, and we give the current SAVI implementations/deployments. Finally, we conclude with the SAVI limits and the potential future works regarding this topic.

Résumé Il y a une dizaine d'années, suite à une forte augmentation d'attaques utilisant des adresses IP source falsifiées, l'IETF standardisait la technique de "Network Ingress Filtering", ainsi que sa variante dynamique connue sous le terme "uRPF", afin de limiter ces attaques. Malheureusement, suite aux déploiements de cette technique, celle-ci a montré ses limites. Aussi, l'IETF a décidé de standardiser, au sein du groupe de travail "Source Address Validation Improvements" (SAVI), des mécanismes complémentaires affinant la précision de détection d'adresses IP source falsifiées au sein de flux IP. Ces mécanismes reposent principalement sur l'observation de la signalisation d'attribution/assignement d'adresse IP (e.g., DHCP, autoconfiguration IPv6). Dans cet article, nous rappelons tout d'abord les attaques utilisant des adresses IP source falsifiées. Ensuite, nous présentons la technique de "Network Ingress Filtering", et sa variante uRPF, ainsi que les limites de ce mécanisme de protection. Nous décrivons ensuite les différentes solutions SAVI standardisées actuellement. Après cela, nous décrivons une intégration de SAVI dans le cas concret du réseau d'accès IPv6 ADSL/Fibre et nous donnons les implémentations/déploiements existants à ce jour. Enfin, nous concluons avec les limites propres à SAVI et les potentiels futurs travaux sur ce sujet.

Source Address Validation Improvements (SAVI)

Mécanismes de prévention contre l'usurpation d'adresses IP source

Jean-Michel Combes[1], Maryline Laurent[2]

[1] France Telecom - Orange, Orange Labs
38 rue du General Leclerc
92130 Issy-Les-Moulineaux, France
jeanmichel.combes@orange.com
[2] Institut TELECOM, TELECOM SudParis
CNRS Samovar UMR 5157
9 rue Charles Fourier
91011 Evry, France
Maryline.Laurent@it-sudparis.eu

1 Introduction

L'adresse IP dans l'Internet sert à la fois à identifier et à localiser un noeud IP. Ainsi, afin de ne pas pouvoir remonter à la source d'une attaque, des noeuds IP malveillants peuvent falsifier leur adresse IP source, en usurpant soit une adresse appartenant à un autre noeud IP, soit une adresse non attribuée. Afin de lutter contre cette falsification, l'*Internet Engineering Task Force* (**IETF**) a standardisé et encouragé le déploiement de la technique appelée "Network Ingress Filtering", dont la déclinaison la plus connue est *unicast Reverse Path Forwarding* (**uRPF**). Malheureusement, celle-ci a certaines limites dont la principale est sa précision. Aussi, l'IETF a lancé des travaux, au sein du groupe de travail *Source Address Validation Improvements* (**SAVI**), pour standardiser des mécanismes complémentaires permettant une meilleure lutte contre l'usurpation d'adresses IP source.

Dans un premier temps, nous allons rappeler les diverses attaques pouvant s'appuyer sur l'usurpation d'adresse IP source. Ensuite, nous présenterons la technique de "Network Ingress Filtering" et expliquerons ses limites. Nous décrirons alors les différentes solutions standardisées dans le groupe de travail SAVI à l'IETF. Après cela, nous décrivons une intégration de SAVI dans le cas concret du réseau d'accès IPv6 ADSL/Fibre et nous donnons les implémentations/déploiements existants à ce jour. Enfin, nous concluons avec les limites propres à SAVI et les potentiels futurs travaux sur le sujet.

2 Menaces liées à l'usurpation d'adresses IP source

Les attaques pouvant utiliser l'usurpation d'adresses IP source peuvent être classifiées selon leurs conséquences [1]. Tout d'abord, il y a celles dont l'objectif est

de corrompre des bases de données, connues sous le terme technique de "Poisoning", afin généralement de pouvoir enchaîner avec une autre attaque. Ensuite, celles dont l'objectif est de bloquer l'accès à un service en le mettant à terre, connues sous le nom de *Denial of Service* (**DoS**). Enfin, celles dont le but est la reconnaissance et l'infiltration dans les systèmes.

2.1 Poisoning

L'objectif d'attaques de type "Poisoning" est d'introduire, dans une base de données, des informations erronées. Celles-ci pourront alors servir pour la mise en place d'un autre type d'attaque comme, par exemple, celle du type *Man in the Middle* (**MitM**) permettant de dérouter le trafic vers l'attaquant et ainsi d'obtenir, et modifier s'il le désire, les informations échangées entre 2 noeuds IP. Voici quelques unes des attaques de ce type les plus connues :

- ARP Poisoning
  L'*Address Resolution Protocol* (**ARP**) est un mécanisme [2] qui permet à un noeud IPv4 de connaître l'adresse de niveau 2 (e.g., adresse MAC) associée à une adresse IPv4 attribuée à un autre noeud IPv4. La réponse à une requête ARP est stockée dans une table ARP dans le noeud IPv4. Un attaquant accomplit un "ARP Poisoning", soit en envoyant à la victime des messages de type "Unsolicited ARP", soit en répondant à des message de type "ARP Request" émis par la victime. Les messages envoyés par l'attaquant contiennent des informations erronées (e.g., adresse MAC de l'attaquant) corrompant la table ARP de la victime, ce qui peut entraîner par la suite une attaque de type MitM ou DoS.
- NDP Poisoning
  Le mécanisme *Neighbor Discovery Protocol* (**NDP**) [3] a globalement la même fonction qu'ARP mais pour le protocole IPv6. Comme pour ARP, l'attaquant envoie des messages NDP contenant des informations erronées pour corrompre les bases de données NDP (e.g., "Neighbor Cache", "Default Router List"), aidant par la suite pour une attaque de type MitM ou DoS. La base de données NDP ainsi modifiée dépend du type de message NDP utilisé [4] lors de l'attaque.
- Table de routage
  Le rôle d'un message "ICMP Redirect", envoyé par un routeur, est d'avertir le noeud IP le recevant que le meilleur chemin pour une communication vers un noeud spécifique est via un autre routeur. Un attaquant peut envoyer un message "ICMP Redirect" [4], contenant des informations erronées, pour corrompre la table de routage de la victime.
- DNS Cache Poisoning
  Le *Domain Name System* (**DNS**) [5] permet d'associer un nom de machine (e.g. www.example.org) à une adresse IP. Afin de réduire les requêtes DNS, la technologie "DNS Cache" est déployée au sein d'entreprises et de fournisseurs d'accès Internet. L'attaque "DNS Cache Poisoning" est une attaque contre l'infrastructure DNS [6] où l'attaquant pousse des informations erronées au sein d'un "DNS Cache".

2.2 Denial of Service

L'objectif d'une attaque de type *Denial of Service* (**DoS**) est de perturber, voir d'interrompre, un service fourni. Généralement, ce genre d'attaque s'appuie sur l'envoie massif de messages sur l'entité fournissant le service, communément appelé "Flooding". Voici quelques unes des attaques de ce type les plus connues :

- LAND
  L'attaque *Local Area Network Denial* (**LAND**) est basée sur l'envoie d'un paquet IP dont l'adresse IP source et destination sont l'adresse IP de la victime. A la réception de celui-ci, la victime va continuellement s'envoyer des paquets IP. Ceci a pour conséquence une utilisation croissante des ressources de la victime (e.g., le paquet falsifié est un paquet TCP SYN), pouvant même entraîner son "crash"[3].
- UDP Flooding
  L'attaque "UDP Flooding"[4] est basée sur l'envoie massif par l'attaquant de datagrammes UDP, dont l'adresse IP source est falsifiée et les ports UDP sont aléatoires, à destination de sa victime. Cette dernière répond alors avec des messages ICMP de type "Destination Unreachable" entraînant une consommation importante de ses ressources.
- TCP SYN Flooding
  L'attaque "TCP SYN Flooding" [7] est basée sur l'envoie massif par l'attaquant de messages TCP SYN, dont l'adresse IP source est falsifiée, à destination de sa victime. Ceci entraîne une allocation importante de ses ressources afin de maintenir les demandes de connexion provenant de l'attaquant et peut résulter à des refus de demandes de connexion légitimes provenant d'autres noeuds IP.
- ICMP Flooding
  De nombreuses attaques DoS basées sur ICMP existent. Une des plus connues est celle nommée "Smurf attack"[5] et qui consiste à l'envoie massif par l'attaquant de messages ICMP de type "Echo Request", dont l'adresse IP destination est une adresse de type broadcast ou multicast et dont l'adresse IP source est falsifiée et correspond à l'adresse IP de la victime. Ainsi, tous les noeuds IP recevant ce type de message ICMP répondent un message ICMP de type "Echo Reply" à destination de la victime qui se retrouve submergée de messages.

2.3 Reconnaissance et infiltration

Afin de préparer ses attaques, un attaquant peut scanner de potentielles vulnérabilités sur une cible spécifique. En utilisant des adresses IP source falsifiées, l'attaquant peut ainsi cacher sa localisation. Ainsi, par exemple, l'application

[3] `http://insecure.org/sploits/land.ip.DOS.html`
[4] `http://www.cert.org/advisories/CA-1996-01.html`
[5] `http://www.cert.org/advisories/CA-1998-01.html`

`nmap`[6], un outil bien connu de reconnaissance réseau, dispose d'une telle fonction.

De la même manière, les vers et "malwares" pourraient cacher la localisation des terminaux déjà infectés durant leur propagation. Par exemple, même si le ver nommé "SQL Slammer"[7], basé sur UDP, ne falsifie pas son adresse IP source, rien n'empêche des codes dérivés de ce ver d'ajouter une telle fonctionnalité.

3 "Network Ingress Filtering"

Suite à l'augmentation[8] d'attaques DoS basées sur l'usurpation d'adresse IP source, l'IETF a standardisé la technique de "Network Ingress Filtering" et a encouragé son déploiement. Plusieurs variantes de cette technique existent : soit statique, connue sous le terme "BCP 38" [8], soit dynamique, connue sous le terme "BCP 84" [9] ou aussi *unicast Reverse Path Forwarding* (**uRPF**). Malheureusement, cette technique a certaines limites qui permettent encore l'usurpation d'adresse IP source dans certains cas mais surtout elles peuvent entraîner que du trafic légitime soit considéré comme illégitime (i.e., trafic ayant des adresses IP source falsifiées) et alors rejeté par l'entité appliquant la technique.

3.1 BCP 38

La variante statique de la technique de "Network Ingress Filtering" a été standardisée en premier afin de répondre rapidement à l'augmentation d'attaques DoS basées sur l'usurpation d'adresse IP source comme indiqué précédemment. L'idée est simplement de mettre en place des règles de filtrage au niveau des bordures du réseau des entreprises et fournisseurs d'accès Internet (e.g., au sein des routeurs de bordure). Ces règles vont vérifier si l'adresse IP source de chaque paquet, transitant par l'entité appliquant la technique, est légitime ou pas. Par légitime, cela indique que l'adresse IP source du paquet entrant sur l'interface de l'entité, est bien incluse dans le ou les préfixes IP alloués au réseau rattaché à cette interface. Tout paquet IP qui ne respecte pas ces règles est rejeté et loggé.

Ainsi, supposons, par exemple, que la technique de "Network Ingress Filtering" est déployée dans l'architecture réseau illustrée dans la Figure 1, où :

- Le préfixe A a été alloué au Client A par le fournisseur d'accès Internet ;
- Le préfixe B a été alloué au Client B par le fournisseur d'accès Internet ;
- Le fournisseur d'accès Internet a déployé du "Network Ingress Filtering" au niveau des routeurs A et B.

Alors, le Routeur A ne laissera sortir du réseau du client A que les paquets dont l'adresse IP source est comprise dans le préfixe A. De même, le Routeur B ne laissera sortir du réseau du client B que les paquets dont l'adresse IP source est comprise dans le préfixe B.

[6] `http://nmap.org/`
[7] `http://www.cert.org/advisories/CA-2003-04.html`
[8] `http://www.cert.org/advisories/CA-1996-21.html`

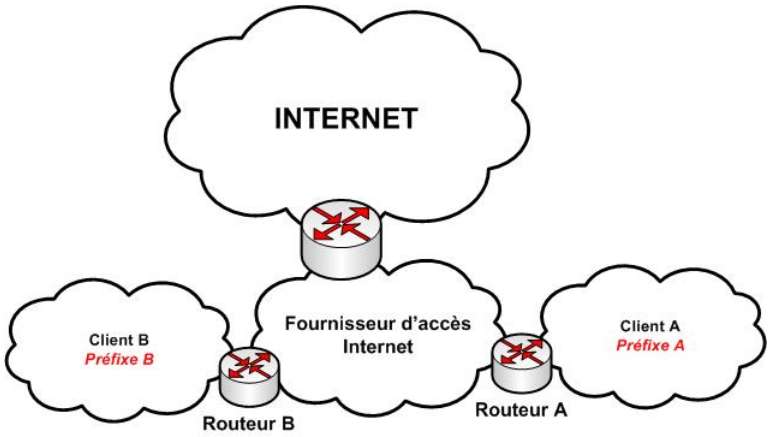

Figure 1. Exemple d'architecture intégrant du "Network Ingress Filtering"

3.2 BCP 84

La limitation majeure du BCP 38 est que les règles de filtrages sont configurées manuellement : cela peut entraîner des rejets de trafic légitime provenant d'un réseau, dans le cas d'une erreur humaine lors de la configuration des règles mais aussi, dans le cas où un nouveau préfixe IP serait alloué au réseau et la mise à jour des règles de filtrages ne se ferait pas assez rapidement. Le BCP 84 [9], plus connu sous le nom de *unicast Reverse Path Forwarding* (**uRPF**), a été standardisé par l'IETF afin de répondre à cette problématique et est actuellement le plus répandu au niveau déploiement.

uRPF permet de mettre en place dynamiquement les règles de filtrage et pour cela, il s'appuie sur les tables de routage (e.g., BGP [10] [11]). En effet, pour savoir si un paquet, contenant une adresse IP source spécifique et arrivant à l'entité effectuant le filtrage, est légitime, cette dernière va vérifier, grâce à sa table de routage, si dans le cas où elle aurait reçu un paquet contenant cette même adresse IP, mais cette fois-ci en adresse IP destination, elle l'aurait pu le router. Si ce n'est pas le cas, le paquet IP est rejeté et loggé.

Ainsi, par exemple, en reprenant l'architecture de la Figure 1, le Routeur A sait, grâce à sa table de routage, qu'il doit transmettre tout paquet IP contenant une adresse IP destination incluse dans le préfixe A vers le réseau du client A et que tout paquet avec une autre adresse IP destination doit être envoyé vers le réseau du fournisseur d'accès Internet. Aussi, en s'appuyant sur la table de routage, il considérera que tout paquet IP, sortant du réseau du client A, ne sera légitime que si l'adresse IP source est incluse dans le préfixe A. Toute autre adresse IP source entraînera que le paquet IP sera considéré comme illégitime (i.e., le paquet est rejeté et loggé).

Il existe 4 modes pour uRPF :

- mode "Strict uRPF"

C'est le mode de base de uRPF. Dans ce mode, basé seulement sur le meilleur chemin de routage (i.e., "best routing path"), un paquet IP, avec une adresse IP source spécifique, reçu sur une interface de l'entité appliquant uRPF, est considéré comme légitime que si un paquet IP avec cette même adresse IP, mais cette fois ci en tant qu'adresse IP destination, serait envoyé par la même interface.

- mode "Feasible uRPF"
  Ce mode est une extension au mode "Strict uRPF" : les chemins de routage alternatifs sont pris en compte pour effectuer le test de légitimité d'un paquet IP (i.e., le paquet IP peut arriver sur plus d'une interface pour qu'il soit considéré comme légitime).
- mode "Loose uRPF"
  Ce mode est similaire au mode "Strict uRPF" mais pour que le paquet soit considéré comme légitime, il suffit qu'il existe un chemin de routage pour l'adresse IP source, et l'existence d'une route par défaut remplit la condition. De plus, l'interface sur laquelle le paquet IP arrive n'est pas tenu en compte (i.e., le paquet peut arriver sur n'importe interface de l'entité exécutant uRPF).
- "Loose uRPF" (en ignorant les routes par défaut)
  Ce mode est identique au mode "Loose uRPF" mais les routes par défaut ne sont pas tenues en compte pour le test de légitimité d'un paquet IP.

4 Limites

Même si la technique de "Network Ingress Filtering" permet d'atténuer la circulation d'un bon nombre de paquets IP contenant une adresse IP source falsifiée, cette technique a ses limites. En effet, dans certains cas (i.e., routage asymétrique et réseau "multihomé"), du trafic légitime peut être considéré comme illégitime par les entités appliquant du "Network Ingress Filtering". Inversement, suite à une faible granularité dans les règles de filtrage, le "Network Ingress Filtering" peut considérer du trafic illégitime comme légitime.

4.1 Routage asymétrique

Un réseau pratiquant du routage asymétrique est un réseau où les flux IP entrants et les flux IP sortants passent par des routeurs distincts.

Ainsi, comme illustré dans la Figure 2, le Routeur 1 achemine le trafic sortant du réseau du client vers le fournisseur d'accès Internet et le Routeur 2 achemine le trafic entrant vers le réseau du client en provenance du fournisseur d'accès. Dans le cas où le Routeur 1 appliquerait du "Network Ingress Filtering", si les informations de routage ne sont pas partagées convenablement entre les Routeur 1 et Routeur 2, du trafic légitime sera considéré comme illégitime et donc rejeté par le Routeur 1.

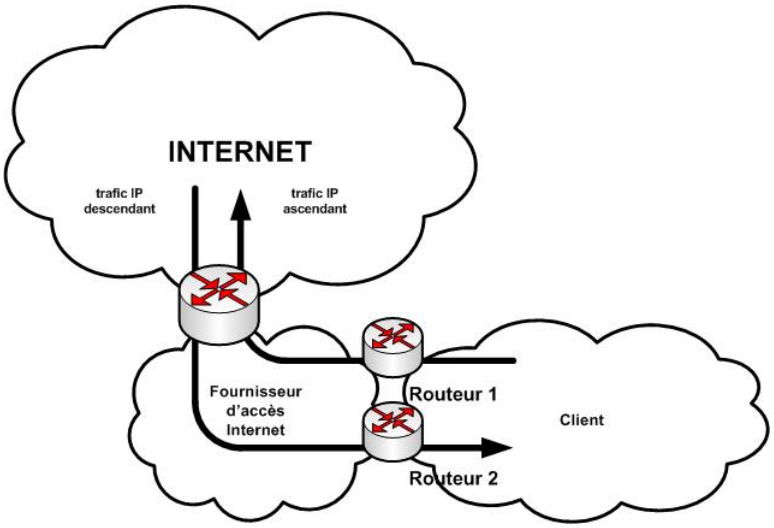

Figure 2. Routage asymétrique

4.2 Réseau "multihomé"

Un réseau client dit "multihomé" est un réseau connecté à plusieurs fournisseurs d'accès Internet. Généralement, chacun de ces fournisseurs alloue un préfixe réseau différent au client. Maintenant, suivant la politique de routage du client, des paquets IP ayant une adresse IP source basée sur le préfixe d'un des fournisseurs pourrait être dirigés vers le réseau d'un des autres fournisseurs. Si ce dernier a déployé du "Network Ingress Filtering" sur le routeur connecté au client, ces paquets IP pourraient être considérés comme illégitimes et rejetés.

Par exemple, comme illustré dans la Figure 3, la politique de routage du client est d'acheminer tout son trafic sortant via le fournisseur d'accès Internet A (e.g., pour des raisons de QoS) et de recevoir tout son trafic entrant via le fournisseur d'accès B. Si le routeur A applique du "Network Ingress Filtering", et le fournisseur B ne lui fournit aucune information, le trafic IP sortant du client, ayant une adresse IP source basée sur le préfixe réseau B, sera considéré comme illégitime et rejeté.

4.3 Granularité

Les principes du BCP 38 et BCP 84 reposent sur l'utilisation de règles de filtrage se basant sur des préfixes pour savoir si un paquet IP avec une adresse IP source spécifique est légitime ou pas. Cela signifie qu'un attaquant peut encore falsifier son adresse IP source si l'adresse utilisée est "topologiquement" correcte : l'attaquant utilise une adresse liée au préfixe IP alloué au réseau dans lequel il se trouve.

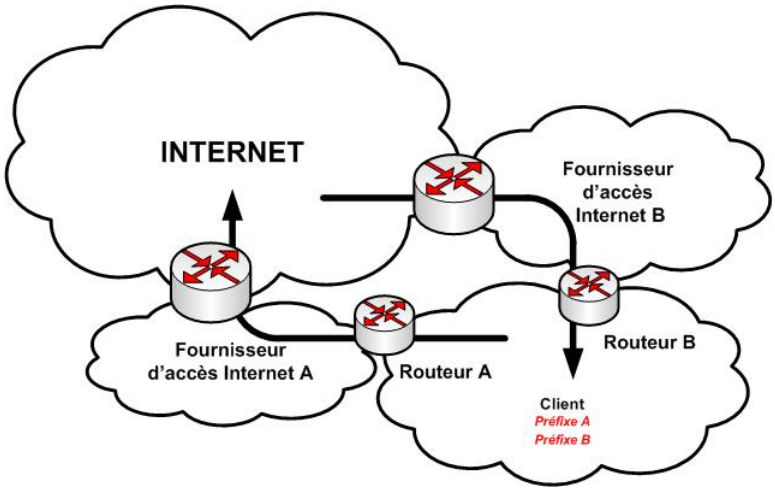

FIGURE 3. Réseau "multihomé"

Ainsi, comme illustré dans la Figure 4, nous supposons que le fournisseur d'accès Internet a alloué le préfixe IP A au client A et le routeur A de ce fournisseur d'accès effectue du "Network Ingress Filtering". Alors, rien n'empêchera le terminal A2, qui a l'adresse IP IP@A2 normalement, d'usurper l'adresse IP du terminal A1, à savoir l'adresse IP IP@A1. Le routeur A considérera alors le trafic de A2 comme légitime alors que ce n'est pas le cas. Ayant même conséquence, le terminal A2 peut aussi usurper une adresse IP non assignée et liée au préfixe A.

5 SAVI

Le but premier du groupe de travail *Source Address Validation Improvements* (**SAVI**) à l'IETF[9] est de répondre à la dernière limite explicitée précédemment : spécifier des mécanismes qui empêchent des terminaux connectés sur un même lien IP d'usurper des adresses IP de ce même lien. Il est à noter qu'il existe de tels mécanismes déjà mais ceux-ci sont partiels et propriétaires (e.g., *IP source guard* [10]).

5.1 Principes

Les solutions spécifiés dans SAVI [12] reposent sur l'observation du trafic des terminaux IP et l'observation, voir l'utilisation, des protocoles d'assignation/allocation d'adresses IP. Ces derniers sont :

[9] `http://datatracker.ietf.org/wg/savi/charter/`

[10] `http://www.cisco.com/en/US/docs/switches/lan/catalyst6500/ios/12.2SY/configuration/guide/ip_source_guard.html`

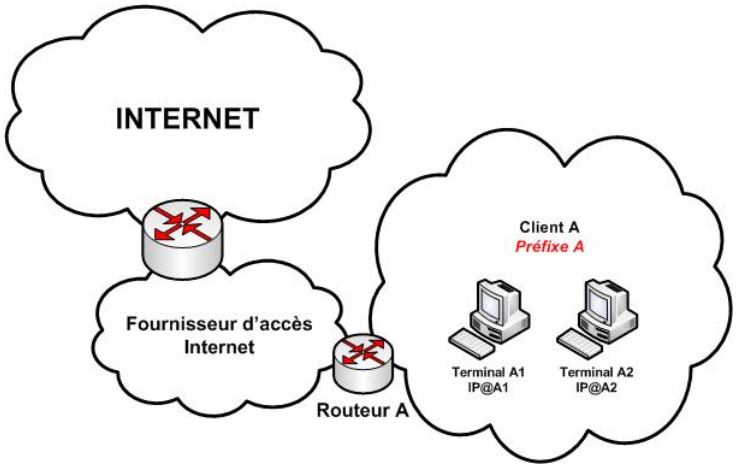

Figure 4. Faiblesse de la granularité

- *Dynamic Host Configuration Protocol* (**DHCP**) [13] en IPv4,
- *Stateless Address Autoconfiguration* (**SLAAC**) [14] en IPv6,
- *Secure Neighbor Discovery* (**SEND**) [15] en IPv6,
- *Dynamic Host Configuration Protocol for IPv6* (**DHCPv6**) [16] en IPv6.

A partir de cette observation, une solution SAVI identifie quelle adresse IP est légitime pour un terminal IP. Elle peut alors lier, *SAVI Binding* (**SAVI-B**), cette adresse IP à un "point d'ancrage", *Binding Anchor* (**BAnchor**), qui identifie ce qui relie le terminal IP au réseau et ainsi au dispositif matériel intégrant la solution SAVI, l'*Entité SAVI*. Le niveau de sécurité du BAnchor conditionne le niveau de fiabilité des solutions SAVI. Comme illustré dans la Figure 5, si l'entité SAVI utilise comme BAnchor l'adresse MAC des terminaux IP, tout d'abord il ne pourra différencier les adresses MAC des terminaux Terminal 1 et Terminal 2, respectivement @MAC1 et @MAC2, car ces terminaux se trouvent derrière un switch de couche 2. L'entité SAVI ne peut connaître que l'adresse MAC de ce switch. Plus grave, il est facilement envisageable pour le terminal Terminal 3 d'usurper cette adresse MAC. Aussi, généralement, ce sont des points d'ancrage de l'entité SAVI qui sont choisis comme BAnchor, comme par exemple :

- si c'est un switch de couche 2, le BAnchor est un port de switch,
- si c'est un routeur IP, le BAnchor est soit une interface réseau de ce routeur, soit l'adresse MAC de cette interface réseau,
- si c'est un point d'accès réseau 802.11, le BAnchor est un "Security Association" 802.1x.

Finalement, la solution SAVI met en place la règle de filtrage suivante : un paquet est considéré comme légitime s'il existe un SAVI-B qui associe l'adresse IP source de ce paquet et le BAnchor par lequel transite ce paquet. Dans le cas

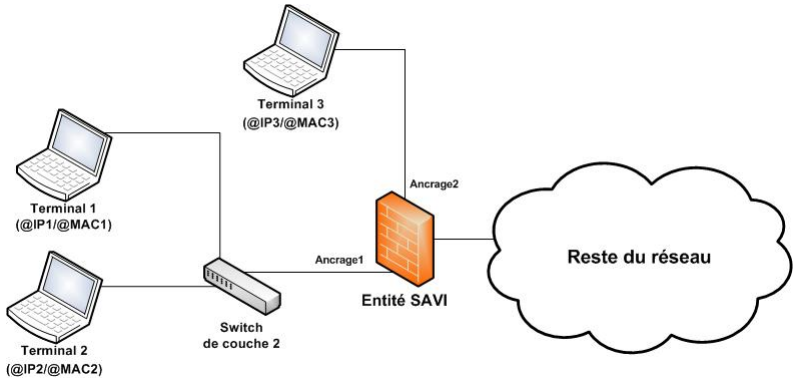

Figure 5. Architecture SAVI

contraire, le paquet est considéré comme illégitime : il sera rejeté et l'événement sera loggé.

Dans la suite de ce document, il est assumé lors de la description des différentes solutions SAVI que le BAnchor est de type port de switch car les entités SAVI déployées seront généralement un switch de couche 2.

Afin d'optimiser les solutions SAVI en réduisant au minimum l'activité des entités SAVI, le principe de périmètre de protection SAVI a été défini. Ce périmètre de protection est délimité par les entités SAVI. Il permet de classifier les flux IP en 2 catégories : ceux provenant de l'intérieur du périmètre, qui seront considérés comme de confiance et donc légitimes, et ceux provenant de l'extérieur du périmètre, et qui ne seront pas considérés de confiance. Ainsi, les solutions SAVI ne s'appliqueront qu'à cette dernière catégorie de flux. Comme illustré dans la figure 6, ainsi l'entité SAVI SAVI 3 n'a pas à se soucier des flux provenant du noeud Noeud IP 1, car les flux provenant de ce noeud sont déjà filtrés par l'entité SAVI SAVI 1 qui fait partie du périmètre de protection SAVI.

Enfin, les solutions SAVI peuvent nécessiter de générer ou reconfigurer des SAVI-B pour certaines raisons. En effet, tout d'abord, la signalisation SLAAC, ainsi que SEND, n'est pas fiable et certains des échanges peuvent être "perdus", ce qui empêchera la génération correcte d'un SAVI-B. Un autre cas est lorsque le noeud IP est mobile et il se peut qu'il change de BAnchor lors d'un mouvement. Enfin, tout simplement, suite à un soucis technique (e.g., coupure de courant), l'entité SAVI peut perdre toute sa base de données stockant les SAVI-B. Ainsi, chaque solution SAVI a un mécanisme de récupération de SAVI-B.

5.2 FCFS SAVI

La solution *First-Come First-Serve* (**FCFS**) SAVI [17] utilise la signalisation SLAAC pour générer les SAVI-B. Avec SLAAC, c'est le premier noeud générant une adresse IPv6 spécifique qui peut se l'assigner et ainsi, avec cette solution,

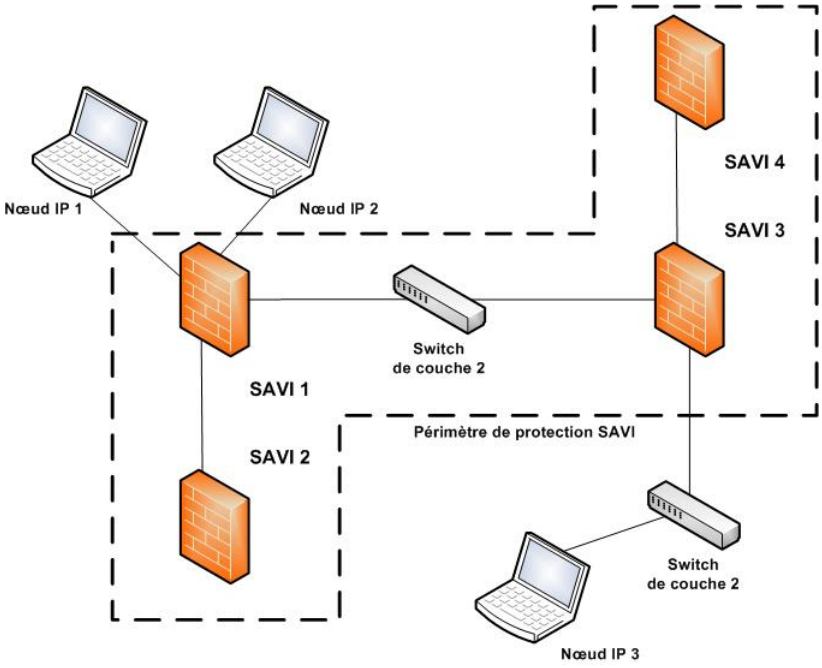

Figure 6. Exemple de Périmètre de protection SAVI

aura un SAVI-B correspondant à cette adresse IPv6. C'est de là que provient le nom de cette solution SAVI.

"Neighbor Discovery" en IPv6. Le mécanisme SLAAC [14] repose sur le protocole *Neighbor Discovery* (**NDP**) [3], qui lui même repose sur l'utilisation de messages ICMPv6 [18] qui sont :

- *Router Solicitation* (**RS**), requête envoyée par un terminal IPv6 afin d'obtenir des informations d'un routeur,
- *Router Advertisement* (**RA**), message envoyé par un routeur IPv6 contenant des informations sur le routeur ainsi que sur le lien réseau (e.g., préfixe IPv6 du lien réseau),
- *Neighbor Solicitation* (**NS**), requête envoyée par un noeud IPv6 afin d'obtenir des informations concernant un autre noeud IPv6,
- *Neighbor Advertisement* (**NA**), message envoyé par un noeud IPv6 contenant des informations concernant ce noeud.

Lors d'un SLAAC, un noeud IPv6 va générer une adresse IPv6 et afin de vérifier si aucun autre noeud sur le même lien réseau ne se l'est pas assignée déjà, il va effectuer une procédure appelée *Duplicate Address Detection* (**DAD**) [14]. Pour cela, comme illustré dans la Figure 7, (1) le noeud IPv6 va demander si

un autre noeud possède déjà l'adresse @IP qu'il a généré grâce à un message NS multicasté. Si c'est le cas, (2) le noeud en question va l'informer, via un message NA unicasté, qu'il s'est assigné déjà l'adresse @IP. Le noeud IPv6, recevant ce message, ne devra pas alors l'utiliser. Il se peut aussi qu'un autre noeud génère dans le même laps de temps la même adresse @IP et donc (3) envoie lui aussi un message NS multicasté pour vérifier l'unicité de cette adresse. Dans ce cas, aucun des 2 noeuds ne devra utiliser cette adresse. Si au bout d'un certain temps, le noeud IPv6 n'a reçu ni de message NS ni de message NA concernant cette adresse @IP, (4) il se l'assigne et pourra l'utiliser.

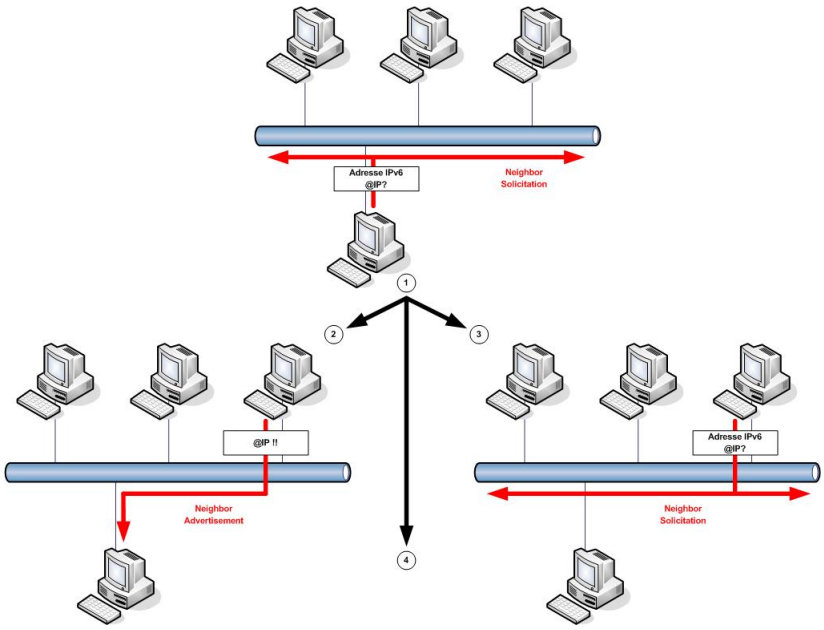

Figure 7. Procédure DAD

La procédure *Neighbor Unreachability Detection* (**NUD**) [3] permet à un noeud IPv6 de savoir si un noeud IPv6 d'adresse @IP est encore joignable à cette adresse. Pour cela, il envoie un message NS unicasté plusieurs fois à destination de cette adresse et s'il ne reçoit aucun message NA unicasté en retour, cela signifie que le noeud en question n'est plus joignable à cette adresse @IP.

Déroulement de FCFS SAVI. La solution FCFS SAVI définit 2 types de port : le *Trusted Port* (**TP**) et le *Validating Port* (**VP**). Les flux IP arrivant sur un TP proviennent de l'intérieur du périmètre de protection SAVI alors que ceux arrivant sur un VP proviennent de l'extérieur de ce périmètre. Il est à noter

que la solution FCFS SAVI assume que tout routeur IP est inclus au sein du périmètre de protection, comme illustré dans la Figure 8.

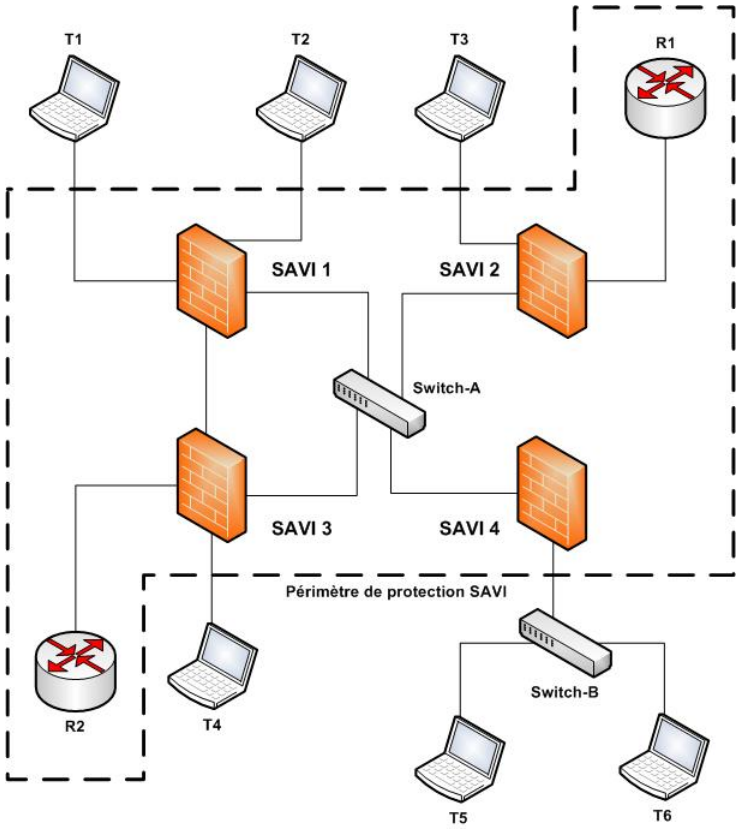

Figure 8. Architecture FCFS SAVI

La solution FCFS SAVI utilise 2 bases de données. La première, nommée *FCFS SAVI Data Base* (**FCFS SAVI DB**), permet de stocker les SAVI-B. Lors de l'initialisation de l'entité SAVI, la FCFS SAVI DB est vide. Chaque entrée de cette base de données est constituée des informations suivantes :

- `IP ADDRESS`
  Indique une adresse IP source.
- `BINDING ANCHOR`
  Indique le BAnchor lié à cette adresse IP source.
- `LIFETIME`
  Indique la durée de vie du SAVI-B.

- `STATUS`
  Indique l'état du SAVI-B (i.e., `TENTATIVE`, `VALID`, `TESTING_VP` ou `TESTING_TP-LT`).
- `CREATION TIME`
  Indique quand l'entrée a été la première fois créée.

La deuxième base de données est la *FCFS SAVI Prefix List* (**FCFS SAVI PL**). Elle permet de stocker les préfixes IP employés sur les liens réseaux observés par l'entité SAVI. La FCFS SAVI PL est complétée, soit manuellement par l'administrateur de l'entité SAVI, soit dynamiquement grâce aux messages RA reçus sur des TPs. Chaque entrée de la FCFS SAVI PL contient les informations suivantes :

- `PREFIX`
  Indique un préfixe IP.
- `PORT`
  Indique le port sur lequel est observé ce préfixe IP.

Lorsqu'un paquet IP est reçu sur l'un des ports de l'entité SAVI, le port `P` par exemple, celui-ci est traité en considérant les différents cas suivants :

1. Si le port est un TP,
   Alors le paquet est considéré comme de confiance (i.e., il provient de l'intérieur du périmètre de protection SAVI) et donc légitime. De plus, si c'est un message NS impliqué dans une procédure DAD (**DAD_NS**), cela active le processus FCFS SAVI.
2. Si le port est un VP et dans le cas où,
   (a) le préfixe lié à l'adresse IP source n'est pas dans la FCFS SAVI PL,
   celui-ci doit provenir d'un noeud IP dans un lien réseau non directement connecté à l'entité SAVI, donc devrait être normalement acheminé par un routeur et arrivé sur un TP. Du coup, le paquet IP est considéré comme illégitime et rejeté.
   (b) le préfixe lié à l'adresse IP source est dans la FCFS SAVI PL,
   alors le processus FCFS SAVI, décrit par la suite, est activé.
   (c) l'adresse IP source est l'adresse indéfinie (*unspecified address*, `::0/128`),
      i. et c'est un message DAD_NS,
      alors le processus FCFS SAVI est activé.
      ii. sinon,
      le paquet IP est considéré comme légitime et est acheminé.

Par défaut, une adresse IP source, `@IP`, n'ayant pas d'entrée dans la FCFS SAVI DB, se trouve dans l'état `NO_BIND`. Suite à une des conditions d'activation de la procédure FCFS SAVI mentionnées auparavant, l'entité SAVI vérifie si un SAVI-B existe pour l'adresse `@IP`.

Si ce n'est pas le cas, un SAVI-B est créé et (1) l'état devient `TENTATIVE`, comme illustré dans la Figure 9. Dans le cas 2.c.i, le processus FCFS SAVI attend la fin de la procédure DAD : si elle échoue, (2) l'état repasse en `NO_BIND` sinon (3) il passe en `VALID`. Pour le cas 2.b, l'entité SAVI initie une procédure DAD

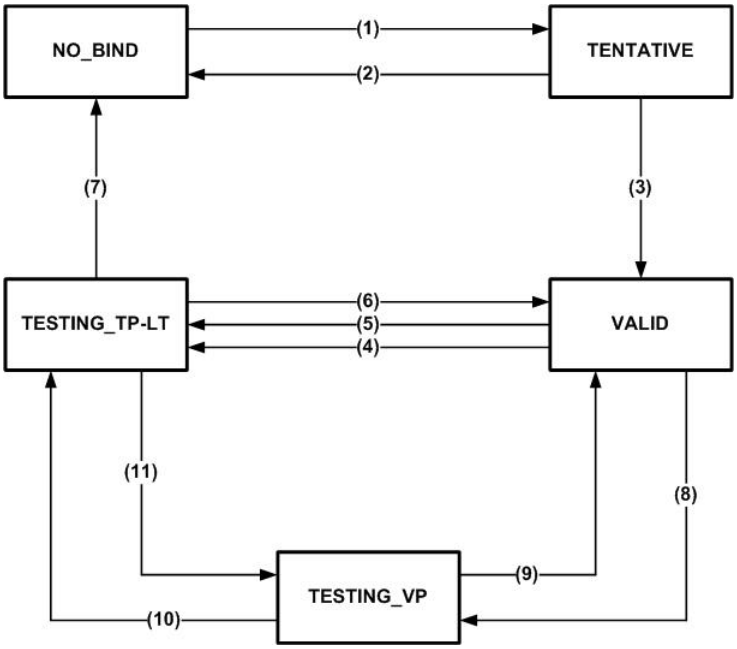

Figure 9. Machine à états de FCFS SAVI

en se faisant passer pour le noeud d'adresse IP `@IP` (i.e., l'entité SAVI usurpe l'adresse IP `@IP`) : si elle échoue, (2) l'état repasse en `NO_BIND` sinon (3) il passe en `VALID`. En arrivant à l'état `VALID`, tout paquet ayant comme adresse IP source `@IP` passant par le port `P` est considéré comme légitime.

Si un SAVI-B existe pour l'adresse `@IP`, avec un état `VALID` pour le port `P`, et qu'un DAD_NS est reçu sur un TP (i.e., cas 1), cela peut indiquer que le noeud IP s'est connecté ailleurs et (4) l'état passe à `TESTING_TP-LT`. Si la procédure DAD réussie, alors (7) l'état passe à `NO_BIND` sinon si elle échoue, soit (6) l'état passe à `VALID` si la réponse dans le DAD était un NA reçu sur le port `P`, soit (11) l'état passe à `TESTING_VP` si la réponse a été reçue sur un port différent `P'`.

Si un SAVI-B existe pour l'adresse `@IP`, avec un état `VALID` pour le port `P`, et que le paquet a été reçu sur un port différent `P'`, alors (8) l'état passe à `TESTING_VP`. Dans le cas où le paquet était un DAD_NS et que la procédure DAD réussi, alors (9) l'état repasse à `VALID` avec maintenant le port `P'` comme BAnchor dans le SAVI-B. Si la procédure DAD échoue, à cause d'un message NA reçu sur le port `P` alors (9) l'état repasse à `VALID`. Si la procédure DAD échoue, à cause d'un DAD_NS reçu sur le port `P''`, différent de `P` et `P'`, alors l'état reste à `TESTING_VP`. Si la procédure DAD échoue, à cause d'un DAD_NS reçu via un TP, alors (10) l'état passe à `TESTING_TP-LT`. Dans le cas où le paquet était autre chose qu'un DAD_NS, alors l'entité SAVI initie une procédure DAD en se

faisant passer pour le noeud d'adresse IP `@IP` (i.e., l'entité SAVI usurpe l'adresse IP `@IP`) et le même processus qu'avec un DAD_NS est appliqué.

Finalement, comme décrit auparavant, chaque SAVI-B a une durée de vie (i.e., champ `LIFETIME`). Cette dernière est renouvelée régulièrement en se basant sur l'activité du noeud IP (e.g., flux data). Un SAVI-B ayant l'état `VALID` et dont la durée de vie expire, (5) passe à l'état `TESTING_TP-LT` et l'entité SAVI initie une procédure DAD en se faisant passer pour le noeud d'adresse IP `@IP` (i.e., l'entité SAVI usurpe l'adresse IP `@IP`) : si elle échoue, (2) l'état repasse en `NO_BIND` sinon (3) il passe en `VALID`.

5.3 SEND SAVI

Le mécanisme SEND SAVI [19] utilise la signalisation SEND pour générer les SAVI-B. En effet, lorsque SEND est déployé, la solution FCFS SAVI ne peut être utilisée : l'entité SAVI peut être obligée d'usurper l'adresse IP d'un des noeuds connectés, ce qui n'est pas possible avec SEND.

"Secure Neighbor Discovery". Le mécanisme SEND [15] est la version sécurisée du protocole NDP. Il repose sur 2 types de matériel de sécurité pour sécuriser les échanges NDP : des adresses IPv6 dites "Cryptographically Generated Addresses" (**CGA**) [20] et des certificats X.509 [21]. Les CGA sont formées à partir du hachage, grâce à SHA-1, de la concaténation d'une clé publique, provenant d'une paire de clés publique/privée de type RSA, et divers paramètres (e.g., le préfixe réseau). Les CGA servent à signer les messages NS, NA et RS alors que les certificats X.509 servent à signer les messages RA. Les CGA permettent de prouver qu'un noeud IPv6 est le possesseur légitime d'une adresse IPv6 et les certificats X.509 permettent de prouver qu'un noeud IPv6 est autorisé à être routeur.

Déroulement de SEND SAVI. Comme FCFS SAVI, la solution SEND SAVI définit 2 types de port : le *Trusted Port* (**TP**) et le *Validating Port* (**VP**). Les flux IP arrivant sur un TP proviennent de l'intérieur du périmètre de protection SAVI alors que ceux arrivant sur un VP proviennent de l'extérieur de ce périmètre. La solution SEND SAVI assume qu'un seul noeud IP est connecté sur chaque port et que, contrairement à FCFS SAVI, un routeur IP n'est pas obligatoirement inclus au sein du périmètre de protection, comme illustré dans la Figure 10.

La solution SEND SAVI utilise 3 bases de données. La première, nommée *SEND SAVI Data Base* (**SEND SAVI DB**), permet de stocker les SAVI-B. Lors de l'initialisation de l'entité SAVI, la SEND SAVI DB est vide. Chaque entrée de cette base de données est constituée des informations suivantes :

- `IP ADDRESS`
  Indique une adresse IPv6 source.
- `BINDING ANCHOR`
  Indique le BAnchor lié à cette adresse IPv6 source.

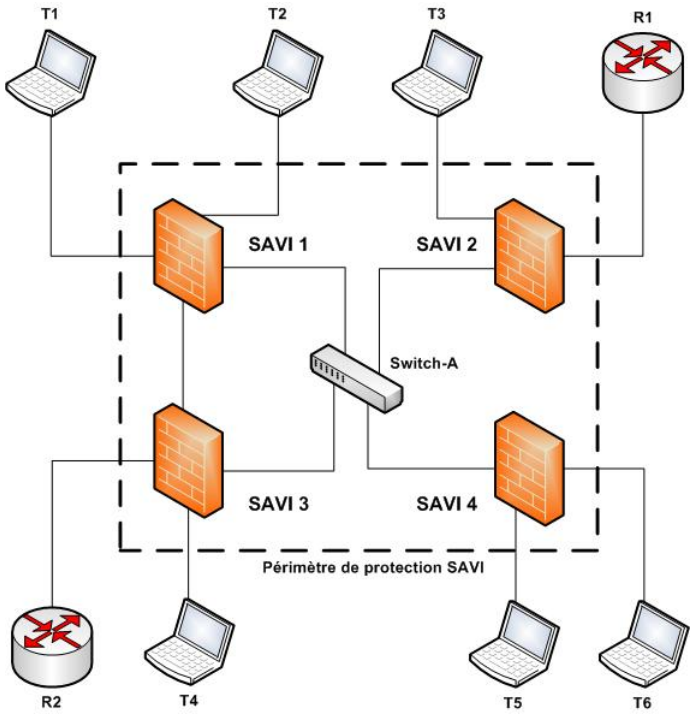

Figure 10. Architecture SEND SAVI

- `LIFETIME`
  Indique la durée de vie du SAVI-B.
- `STATUS`
  Indique l'état du SAVI-B (i.e., `TENTATIVE_DAD`, `TENTATIVE_NUD`, `VALID`, `TESTING_VP` ou `TESTING_VP'`).
- `ALTERNATIVE BINDING ANCHOR`
  Indique un BAnchor, différent de celui dans `BINDING ANCHOR`, par lequel un DAD_NS ou un paquet de données est arrivé.
- `CREATION TIME`
  Indique quand l'entrée a été la première fois créée.

La deuxième base de données est la *SEND SAVI Prefix List* (**SEND SAVI PL**). Elle permet de stocker les préfixes IP employés sur les liens réseaux observés par l'entité SAVI. La SEND SAVI PL est complétée, soit manuellement par l'administrateur de l'entité SAVI, soit dynamiquement grâce aux messages RA reçus suite à une requête RS envoyée régulièrement par l'entité SAVI. Chaque entrée de la SEND SAVI PL contient les informations suivantes :

- `PREFIX`
  Indique un préfixe IPv6.

– `PREFIX LIFETIME`
  Indique le temps de vie du préfixe IPv6.

Enfin, la dernière base de donnée est la *SEND SAVI Router List* (**SEND SAVI RL**). Elle permet de répertorier les routeurs attachés à un VP. La SEND SAVI RL est complétée, soit manuellement par l'administrateur de l'entité SAVI, soit dynamiquement grâce aux messages RA reçus suite à une requête RS envoyée régulièrement par l'entité SAVI. Chaque entrée de la SEND SAVI RL contient les informations suivantes :

– `IPV6 ADDRESS`
  Indique l'adresse IPv6 d'un routeur.
– `ROUTER LIFETIME`
  Indique le temps de vie de ce routeur.

La solution SEND SAVI ne peut initier une procédure DAD en se faisant passer pour le noeud d'adresse IP `@IP` (i.e., l'entité SAVI usurpe l'adresse IP `@IP`), comme c'est le cas avec FCFS SAVI. En effet, l'entité SAVI ne possède pas les secrets nécessaires pour signer les messages avec SEND. Aussi, la solution SEND SAVI va utiliser à la place la procédure NUD, décrite à la section 5.2. Du coup, il est nécessaire de configurer une CGA sur l'entité SAVI.

Lorsqu'un paquet IP est reçu sur l'un des ports de l'entité SAVI, le port `P` par exemple, celui-ci est traité en considérant les différents cas suivants :

1. Si le port est un TP,
   Alors le paquet est considéré comme de confiance (i.e., il provient de l'intérieur du périmètre de protection SAVI) et donc légitime.
2. Si le port est un VP et dans le cas où,
   (a) le préfixe lié à l'adresse IP source n'est pas dans la SEND SAVI PL,
      i. s'il existe un SAVI-B pour le port `P` et que l'adresse IP `@IP` associée est dans la SEND SAVI RL,
         alors c'est du trafic acheminé par un routeur et le paquet IP est considéré comme légitime.
      ii. sinon,
         le paquet est illégitime et rejeté.
   (b) le préfixe lié à l'adresse IP source est dans la SEND SAVI PL,
      alors le processus SEND SAVI, décrite par la suite, est activé.
   (c) l'adresse IP source est l'adresse indéfinie (*unspecified address*, `::0/128`),
      alors le processus SEND SAVI est activé.

Par défaut, une adresse IP source, `@IP`, n'ayant pas d'entrée dans la SEND SAVI DB, se trouve dans l'état `NO_BIND`. Suite à une des conditions d'activation de la procédure SEND SAVI mentionnées auparavant, l'entité SAVI vérifie si un SAVI-B existe pour l'adresse `@IP`.

Si ce n'est pas le cas, un SAVI-B est créé. Dans le cas 2.c, (1) l'état devient `TENTATIVE_DAD`, comme illustré dans la Figure 11. Le processus SEND SAVI attend la fin de la procédure DAD : si elle échoue, (2) l'état repasse en `NO_BIND`

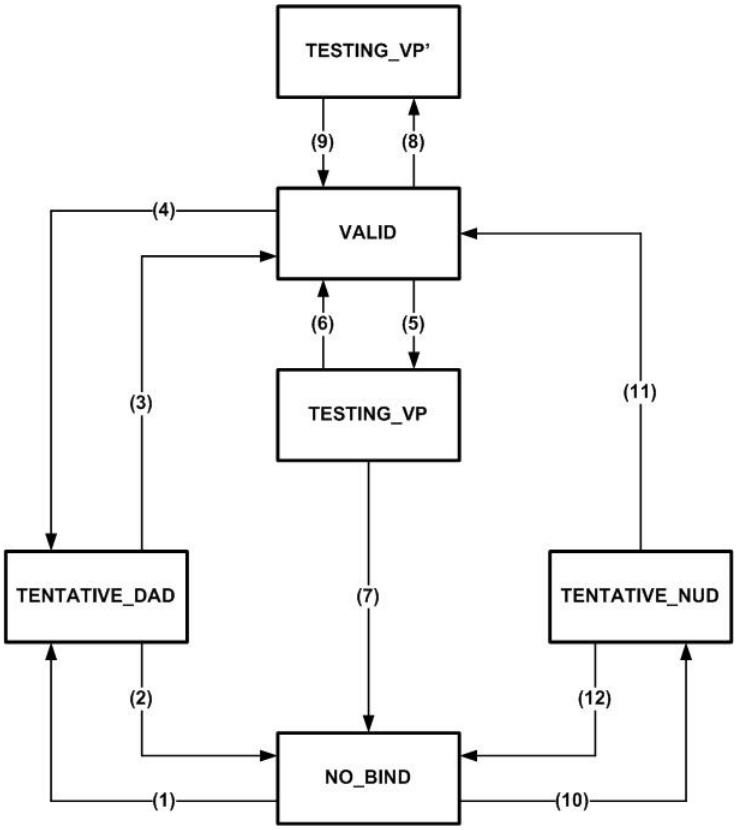

Figure 11. Machine à états de SEND SAVI

sinon (3) il passe en `VALID`. Pour le cas 2.b, (10) l'état devient `TENTATIVE_NUD` et l'entité SAVI initie une procédure NUD : si elle échoue, (12) l'état repasse en `NO_BIND` sinon (11) il passe en `VALID`.

Si un SAVI-B existe pour l'adresse `@IP`, avec un état `VALID` pour le port `P`, cela peut indiquer que le noeud IP s'est connecté ailleurs. L'entité SAVI initie une procédure NUD et (5) l'état passe à `TESTING_VP`. Si la procédure NUD échoue, alors (7) l'état passe à `NO_BIND` sinon (6) l'état repasse à `VALID`.

Si un SAVI-B existe pour l'adresse `@IP`, avec un état `VALID` pour le port `P`, et qu'un DAD_NS est reçu sur un port différent `P'`, cela peut indiquer que le noeud IP s'est connecté ailleurs. L'entité SAVI initie une procédure NUD et (8) l'état passe à `TESTING_VP'`. Si la procédure NUD réussie, alors (7) l'état repasse à `VALID` sinon (9) l'état repasse à `VALID` avec maintenant le port `P'` comme BAnchor dans le SAVI-B.

Si un SAVI-B existe pour l'adresse `@IP`, avec un état `VALID` pour le port `P`, et qu'un DAD_NS est reçu sur `P`, (4) l'état devient `TENTATIVE_DAD`. Le processus

SEND SAVI attend la fin de la procédure DAD : si elle échoue, (2) l'état passe en `NO_BIND` sinon (3) il repasse en `VALID`.

Finalement, comme décrit auparavant, chaque SAVI-B a une durée de vie (i.e., champ `LIFETIME`). Cette dernière est renouvelée régulièrement en se basant sur l'activité du noeud IP (e.g., flux data). Un SAVI-B ayant l'état `VALID` et dont la durée de vie expire, (5) passe à l'état `TESTING_VP` et l'entité SAVI initie une procédure NUD : si elle échoue, alors (7) l'état passe à `NO_BIND` sinon (6) l'état repasse à `VALID`.

Avec la solution FCFS SAVI, un paquet, arrivant sur le port `P` avec une adresse IP source `@IP`, est considéré comme légitime s'il existe un SAVI-B associé à `@IP` avec `P` comme BAnchor et `VALID` comme état. Avec la solution SEND SAVI, le même paquet est considéré légitime aussi avec les états `TENTATIVE_DAD`, `TESTING_VP` et `TESTING_VP'`.

5.4 DHCP SAVI

Le mécanisme DHCP SAVI [22] utilise la signalisation DHCP (i.e., DHCPv4 [13] et DHCPv6 [16]) pour générer les SAVI-B.

DHCPv4 et DHCPv6. Les mécanismes DHCPv4 et DHCPv6 s'appuient sur le modèle client-serveur.

En DHCPv4, comme illustré dans la Figure 12, un client DHCPv4 localise en premier lieu les serveurs DHCPv4 disponibles grâce à un message *DHCPDISCOVER*. Les serveurs disponibles se signalent en répondant avec un message *DHCPOFFER* contenant les informations de configuration. Le client choisit une des configurations proposées en utilisant un message *DHCPREQUEST*. Ce type de message sert aussi à prolonger le temps d'allocation d'une configuration. Le serveur DHCPv4 confirme la configuration et fournit l'adresse IPv4 avec un message *DHCPACK*. Le client libérera l'adresse allouée grâce à un message *DHCPRELEASE*. Enfin, grâce à un message *DHCPDECLINE*, le client informe le serveur DHCPv4 que l'adresse allouée est déjà utilisée.

En DHCPv6, comme illustré dans la Figure 13, un client DHCPv6 localise en premier lieu les serveurs DHCPv6 disponibles grâce à un message *SOLICIT*. Les serveurs disponibles se signalent en répondant avec un message *ADVERTISE*. Le client choisit un des serveurs et demande une configuration en utilisant un message *REQUEST*. Le serveur DHCPv6 fournit la configuration et l'adresse IPv6 allouée avec un message *REPLY*. Un client DHCPv6 a la possibilité de rajouter une option *RAPID COMMIT* dans un message SOLICIT afin de demander directement une réponse de type REPLY de la part du serveur DHCPv6. Le client peut demander un rallongement du temps d'allocation grâce à un message *RENEW*. Enfin, grâce à un message *CONFIRM*, le client peut demander au serveur DHCPv6 si l'adresse allouée est toujours valide.

Certains équipement peuvent nécessiter de connaître l'état des adresses allouées par un serveur DHCP. Les messages *DHCP Leasequery* permettent de fournir les informations en IPv4 [23] et en IPv6 [24].

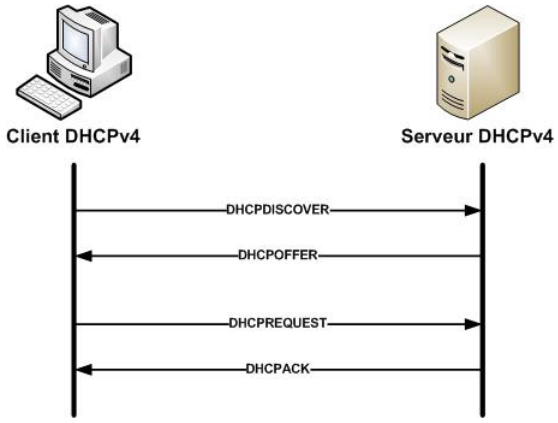

Figure 12. Échanges DHCPv4

Déroulement de DHCP SAVI. La solution DHCP SAVI définit plusieurs types d'attribut de configuration d'un port. Les principaux sont *SAVI-Validation* et *SAVI-SAVI* correspondant respectivement à VP et TP pour les solutions FCFS SAVI et SEND SAVI. Ainsi les flux IP arrivant sur un port ayant l'attribut SAVI-SAVI sont considérés comme légitimes. Alors qu'un flux IP arrivant sur un port ayant l'attribut SAVI-Validation activeront le processus DHCP SAVI. Le périmètre de protection SAVI en découlant est illustré dans la Figure 14.

La solution DHCP SAVI utilise 2 bases de données. La première, nommée *Binding State Table* (**BST**), permet de stocker les SAVI-B. Lors de l'initialisation de l'entité SAVI, la BST est vide. Chaque entrée de cette base de données est constituée des informations suivantes :

- `IP ADDRESS`
  Indique une adresse IP source.
- `BINDING ANCHOR`
  Indique le BAnchor lié à cette adresse IP source.
- `STATE`
  Indique l'état du SAVI-B (i.e., `INIT_BIND` ou `BOUND`).
- `LIFETIME`
  Indique la durée de vie du SAVI-B.
- `TID`
  Indique le *Transaction ID* de l'échange DHCP.

La deuxième base de données est la *Filtering Table* (**FT**). Elle permet de stocker les règles de filtrage découlant de la BST. Chaque entrée de la FT contient les informations suivantes :

- `BINDING ANCHOR`
  Indique un BAnchor.

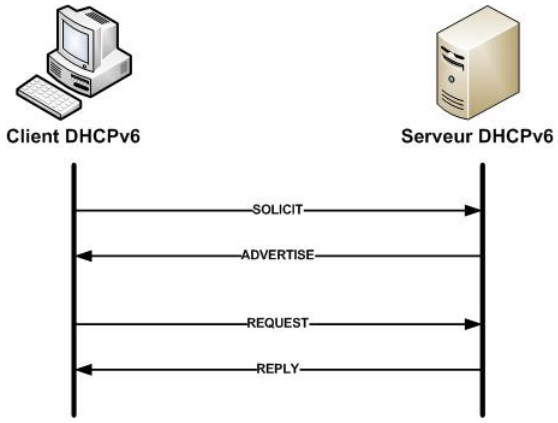

Figure 13. Échanges DHCPv6

– `IP ADDRESS`
  Indique une adresse IP source autorisée à passer par le BAnchor.

La solution DHCP SAVI définit les événements suivants :

– `EVE_DHCP_REQUEST`
  Un message DHCPv4 DHCPREQUEST est reçu sur un port ayant un attribut SAVI-Validation.
– `EVE_DHCP_CONFIRM`
  Un message DHCPv6 CONFIRM est reçu sur un port ayant un attribut SAVI-Validation.
– `EVE_DHCP_OPTION_RC`
  Un message DHCPv6 SOLICIT incluant une option RAPID COMMIT est reçu sur un port ayant un attribut SAVI-Validation.
– `EVE_DHCP_REPLY`
  Un message DHCPv4 DHCPACK ou un message DHCPv6 REPLY est reçu et doit être acheminé vers un port ayant un attribut SAVI-Validation.
– `EVE_DHCP_DECLINE`
  Un message DHCPv4 DHCPDECLINE est reçu sur un port ayant un attribut SAVI-Validation.
– `EVE_DHCP_RELEASE`
  Un message DHCPv4 DHCPRELEASE est reçu sur un port ayant un attribut SAVI-Validation.
– `EVE_LEASEQUERY_REPLY`
  Un message DHCP Leasequery répondant positivement à une demande d'informations est reçu.

Par défaut, une adresse IP source, `@IP`, n'ayant pas d'entrée dans la BST, se trouve dans l'état `NO_BIND`.

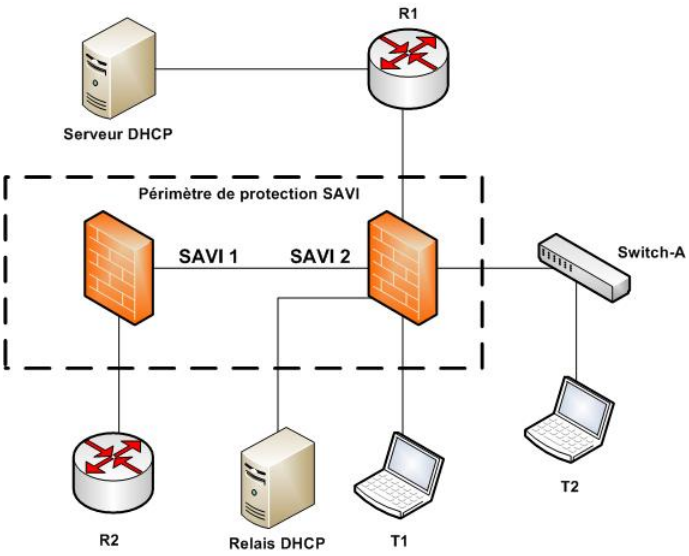

Figure 14. Architecture DHCP SAVI

Lorsque `EVE_DHCP_REQUEST` ou `EVE_DHCP_OPTION_RC` se produit, l'entité SAVI crée un SAVI-B dans la BST et (1) son état passe à `INIT_BIND`, comme illustré dans la Figure 15. Si au bout d'un laps de temps, l'événement `EVE_DHCP_REPLY` ne s'est pas produit alors (4) son état repasse en `NO_BIND` et le SAVI-B est retiré de la BST. Dans le cas contraire, (5) l'état passe en `BOUND`.

Lorsque `EVE_DHCP_CONFIRM` se produit, l'entité SAVI crée un SAVI-B dans la BST et (2) son état passe à `INIT_BIND`. L'entité SAVI émet une requête LEASEQUERY afin de récupérer la durée d'allocation de l'adresse. Si au bout d'un laps de temps, l'événement `EVE_LEASEQUERY_REPLY` ne se produit pas alors (4) son état repasse en `NO_BIND` et le SAVI-B est retiré de la BST. Dans le cas contraire, (6) l'état passe en `BOUND`.

Lorsque `EVE_DHCP_DECLINE` ou `EVE_DHCP_RELEASE` se produit, (8) l'état passe en `NO_BIND` et le SAVI-B est retiré de la BST. De même, lorsque la durée de vie du SAVI-B expire, (9) l'état passe en `NO_BIND` et le SAVI-B est retiré de la BST. La durée de vie du SAVI-B est mise à jour grâce aux messages DHCPv4 DHCPREQUEST et DHCPv6 RENEW.

Dans le cas où un paquet IP avec l'adresse IP source `@IP` arrive sur le port `P` et l'entité SAVI n'a pas de SAVI-B correspondant, DHCP SAVI peut lancer un processus de récupération de SAVI-B qui se décompose en 2 étapes. D'abord, DHCP SAVI va vérifier que l'adresse n'est pas utilisée par un autre noeud grâce, en IPv4, à ARP [2] [25] et, en IPv6, à DAD [14]. Si ce n'est pas le cas, alors ensuite, l'entité SAVI envoie une requête DHCP LEASEQUERY afin de confirmer que l'adresse a été allouée et de connaître la durée de cette allocation.

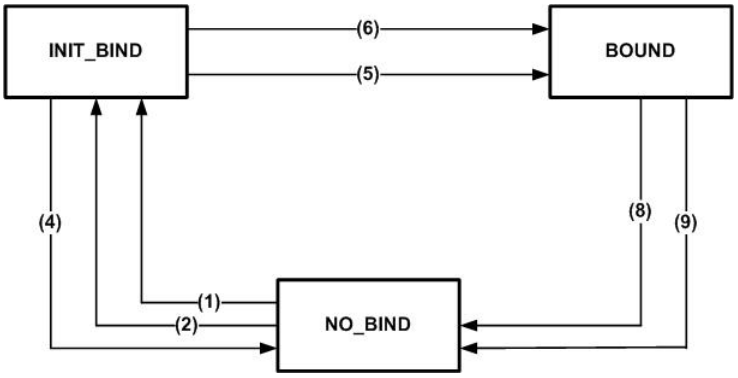

Figure 15. Machine à états de DHCP SAVI

5.5 Mix SAVI

Différentes solutions SAVI peuvent être déployées dans un même réseau si ce dernier utilise plusieurs mécanismes d'allocation/assignation d'adresse IP (e.g., Autoconfiguration IPv6 et SEND) [26]. Cela peut entraîner des problèmes de collisions lors de la génération de SAVI-B.

Afin de limiter de potentielles collisions, il est recommandé d'utiliser des préfixes réseau différents pour chacun des mécanismes d'allocation/assignation d'adresse IP.

Il existe deux cas de collision : une même adresse se retrouve liée à 2 BAnchors (e.g., un noeud A, connecté au BAnchor `X`, récupère l'adresse `@IP` via SLAAC et plus tard un noeud B, connecté au BAnchor `Y`, se voit allouer la même adresse `@IP` via DHCPv6) et une même adresse se retrouve liée plusieurs fois à un même BAnchor (e.g., un noeud A, connecté au BAnchor `X`, récupère l'adresse `@IP` via SLAAC et plus tard, se voit allouer la même adresse `@IP` via DHCPv6). Il est recommandé dans le premier cas de garder le SAVI-B qui est créé en premier, sauf quand SAVI-SEND est utilisé et alors c'est le SAVI-B généré par cette solution qui doit être gardé. Pour le second cas, le principal impact concerne le temps de vie du SAVI-B, aussi, il est recommandé de garder le SAVI-B jusqu'à ce que toutes les solutions SAVI déployés recommandent sa suppression.

6 Intégrations, implémentations et déploiement actuel de SAVI

Afin d'optimiser le potentiel des solutions SAVI, il est important de décider où seront déployés les entités SAVI. Aussi, un exemple d'intégration des solutions SAVI est décrit par la suite. De même, il est intéressant de savoir quelles sont les implémentations disponibles à ce jour et si elles sont actuellement déployées.

6.1 Intégration concrète : réseau d'accès IPv6 ADSL/Fibre

La solution FCFS SAVI, et SEND SAVI si SEND est déployé aussi, est nécessaire [27] dans une architecture IPv6 d'accès ADSL/Fibre de type "split-horizon forwarding", architecture standardisée par le "Broadband Forum" (**BBF**)[11], afin de protéger contre l'usurpation d'adresse IP source. Ce type d'architecture, illustré dans la Figure 16, sera déployé chez France Telecom - Orange par exemple.

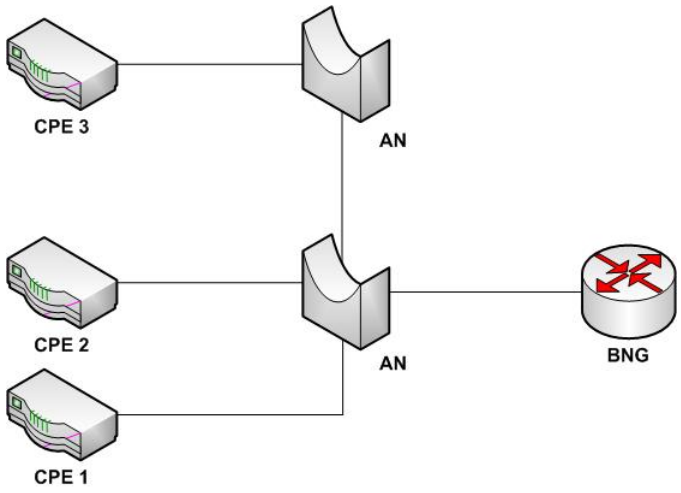

Figure 16. Architecture ADSL/Fibre de type "split-horizon forwarding"

La solution FCFS SAVI, et son équivalent sécurisé SEND SAVI, est intégré dans le *Broadband Network Gateway* (**BNG**), premier routeur IP du réseau du fournisseur d'accès Internet. Le BAnchor est l'adresse MAC du *Customer Premises Equipment* (**CPE**), qui est soit le modem-routeur ou le terminal IP du client. Le niveau de sécurité de ce BAnchor est assuré grâce au mécanisme *Virtual MAC* (**VMAC**), intégré dans l'*Access Node* (**AN**), "bridge" de couche 2. Cela assure l'unicité du point d'ancrage du terminal IP et ainsi le niveau de sécurité du BAnchor.

6.2 Implémentations existantes

A ce jour, les principales implémentations officiellement[12] connues sont :

- draft-ietf-savi-dhcp-07 [28],
- draft-bi-savi-stateless-01 [29],

[11] `http://www.broadband-forum.org/technical/download/TR-177.pdf`
[12] `http://www.ietf.org/proceedings/80/slides/savi-2.pdf`

– draft-bi-savi-mix-04 (partiellement) [30],
– draft-an-savi-mib-00 [31].

Et elles sont produites par :

– ZTE, Huawei, H3C (3Com),
– Ruijie, Digital China ("spin off" de Lenovo),
– Bitway, Centac.

6.3 Déploiement existant

Le réseau de recherche universitaire de Chine *CERNET2* a déployé les solutions SAVI implémentées qui sont mentionnées auparavant. Les entités SAVI sont des switchs de couche 2. Elles sont déployées sur une centaine de sites, illustré dans la Figure 17 par les points rouges. Cela représente environ un million d'utilisateurs impactés par cette technologie.

Figure 17. Déploiement SAVI en Chine (points rouges)

7 Limites des solutions SAVI

Les solutions SAVI ont aussi des limites. En effet, comme les solutions SAVI reposent principalement sur l'observation des protocoles utilisés dans les réseaux, elles héritent des inconvénients liés à ceux-ci.

7.1 Fragmentation

La signalisation observée devient complexe lorsque les solutions SAVI sont intégrées dans du matériel ayant peu de ressources de calcul (e.g., switch de couche 2). Dans le cas où la signalisation IP est fragmentée [32], l'interprétation de celle-ci devient coûteuse pour l'entité SAVI, voir impossible. Aussi, l'entité SAVI ne pourra être capable de générer les SAVI-B. Du coup, du trafic légitime sera considérer comme illégitime.

7.2 "Optimistic DAD"

La procédure *Optimistic DAD* [33], généralement utilisée dans un contexte de mobilité IPv6, permet à un noeud IPv6 d'utiliser une adresse IPv6 avant que la procédure DAD soit finalisée. La conséquence est que les flux IPv6 utilisant ce type d'adresse, seront considérés comme illégitimes, car aucun SAVI-B n'existera, et donc rejetés.

7.3 Privacy

Les solutions SAVI permettent d'associer un noeud IP à une adresse IP. La surveillance d'un réseau pourrait s'appuyer sur les solutions SAVI particulièrement quand une méthode d'allocation d'adresse aléatoire [34] est utilisée. Cela peut aider les administrateurs du réseau où se trouvent des noeuds IPv6 [35]. Mais cela peut aller à l'encontre du principe de la vie privée des utilisateurs [36]. Des discussions sur le sujet au sein de l'IETF retardent la standardisation des solutions SAVI.

8 Potentiels futurs travaux

A court terme, afin de simplifier le déploiement et la gestion des solutions SAVI, l'utilisation du protocole *Simple Network Management Protocol* (**SNMP**) [37] peut être utile. Aussi, il est nécessaire de spécifier une *Management Information Base* (**MIB**) pour les solutions SAVI. Les travaux sur ce sujet ont actuellement débuté [38].

A moyen terme, comme toute nouvelle solution spécifiée, il est nécessaire d'avoir un retour concret des implémentations et déploiements des solutions SAVI. Il y aura certainement des modifications à effectuer du coup dans les spécifications de ces solutions.

A long terme, comme les solutions SAVI actuellement spécifiées ne couvrent pas toutes les techniques d'allocation/assignement d'adresse IP, il pourrait être intéressant, selon le besoin de combler ce manque. Tout particulièrement, dans un contexte de *Virtual Private Network* (**VPN**) IPsec [39] où un terminal IP se connecte à son réseau d'entreprise, le protocole *Internet Key Exchange Protocol Version 2* (**IKEv2**) [40] [41] permet de lui allouer une adresse IP interne à ce réseau. Il est nécessaire alors de se protéger d'un noeud hostile localisé dans ce même réseau désirant usurper l'adresse allouée.

Les solutions SAVI, complémentaires à la technique "Network Ingress Filtering", ne règlent qu'une des limites de cette technique : la granularité. Aussi, il pourrait être nécessaire de réviser cette technique afin de résoudre, ou limiter, les 2 dernières limites.

Enfin, certaines personnes dans la communauté IETF ont soulevé la possibilité d'étendre le périmètre de protection SAVI, dépendant d'un seul opérateur, à un périmètre plus global intégrant plusieurs opérateurs et plusieurs technologies (i.e., SAVI et "Network Ingress Filtering"). Cela permettrait de former une zone de confiance où les flux à surveiller et analyser particulièrement seraient en bordure de celle-ci.

9 Conclusion

Ce document présente des attaques reposant sur l'usurpation de l'adresse IP source puis la réponse apportée par l'IETF, la technique "Network Ingress Filtering". Il explique alors les limites de cette technique qui ont conduit l'IETF a standardisé les solutions SAVI. Ensuite, ces différentes solutions, FCFS SAVI, SEND SAVI, DHCP SAVI et MIX SAVI, sont décrites. Puis, le document décrit une intégration de SAVI dans le cas concret du réseau d'accès IPv6 ADSL/Fibre et mentionne les implémentations/déploiements existants à ce jour. Enfin, il présente les limites propres à SAVI et les potentiels futurs travaux concernant le sujet.

Il est important de signaler que, comme les solutions SAVI sont encore en cours de standardisation, leurs spécifications peuvent encore évoluer dans un proche futur.

10 Remerciements

Les auteurs de ce document tiennent à remercier les membres du groupe de travail SAVI à l'IETF pour leurs implications dans la spécification et standardisation des différentes solutions SAVI.

Références

1. McPherson, D., Baker, F., Halpern, J. : SAVI Threat Scope. Internet-Draft draft-ietf-savi-threat-scope-05, Internet Engineering Task Force (April 2011) Work in progress.
2. Plummer, D. : Ethernet Address Resolution Protocol : Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware. RFC 0826, Internet Engineering Task Force (November 1982)
3. Narten, T., Nordmark, E., Simpson, W., Soliman, H. : Neighbor Discovery for IP version 6 (IPv6). RFC 4861, Internet Engineering Task Force (September 2007)
4. Nikander, P., Kempf, J., Nordmark, E. : IPv6 Neighbor Discovery (ND) Trust Models and Threats. RFC 3756, Internet Engineering Task Force (May 2004)

5. Mockapetris, P. : Domain names - concepts and facilities. RFC 1034, Internet Engineering Task Force (November 1987)
6. Atkins, D., Austein, R. : Threat Analysis of the Domain Name System (DNS). RFC 3833, Internet Engineering Task Force (August 2004)
7. Eddy, W. : TCP SYN Flooding Attacks and Common Mitigations. RFC 4987, Internet Engineering Task Force (August 2007)
8. Ferguson, P., Senie, D. : Network Ingress Filtering : Defeating Denial of Service Attacks which employ IP Source Address Spoofing. RFC 2827, Internet Engineering Task Force (May 2000)
9. Baker, F., Savola, P. : Ingress Filtering for Multihomed Networks. RFC 3704, Internet Engineering Task Force (March 2004)
10. Rekhter, Y., Li, T., Hares, S. : A Border Gateway Protocol 4 (BGP-4). RFC 4271, Internet Engineering Task Force (January 2006)
11. Marques, P., Dupont, F. : Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing. RFC 2545, Internet Engineering Task Force (March 1999)
12. Wu, J., Bi, J., Bagnulo, M., Baker, F., Vogt, C. : Source Address Validation Improvement Framework. Internet-Draft draft-ietf-savi-framework-06, Internet Engineering Task Force (January 2012) Work in progress.
13. Droms, R. : Dynamic Host Configuration Protocol. RFC 2131, Internet Engineering Task Force (March 1997)
14. Thomson, S., Narten, T., Jinmei, T. : IPv6 Stateless Address Autoconfiguration. RFC 4862, Internet Engineering Task Force (September 2007)
15. Arkko, J., Kempf, J., Zill, B., Nikander, P. : SEcure Neighbor Discovery (SEND). RFC 3971, Internet Engineering Task Force (March 2005)
16. Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., Carney, M. : Dynamic Host Configuration Protocol for IPv6 (DHCPv6). RFC 3315, Internet Engineering Task Force (July 2003)
17. Nordmark, E., Bagnulo, M., EricLevy-Abegnoli, E. : FCFS SAVI : First-Come First-Serve Source-Address Validation for LocallyAssigned IPv6 Addresses. Internet-Draft draft-ietf-savi-fcfs-14, Internet Engineering Task Force (February 2012) Work in progress.
18. Conta, A., Deering, S., Gupta, M. : Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification. RFC 4443, Internet Engineering Task Force (March 2006)
19. Bagnulo, M., Garcia-Martinez, A. : SEND-based Source-Address Validation Implementation. Internet-Draft draft-ietf-savi-send-07, Internet Engineering Task Force (March 2012) Work in progress.
20. Aura, T. : Cryptographically Generated Addresses (CGA). RFC 3972, Internet Engineering Task Force (March 2005)
21. Gagliano, R., Krishnan, S., Kukec, A. : Certificate Profile and Certificate Management for SEcure Neighbor Discovery (SEND). RFC 6494, Internet Engineering Task Force (February 2012)
22. Bi, J., Wu, J., Yao, G., Baker, F. : SAVI Solution for DHCP. Internet-Draft draft-ietf-savi-dhcp-12, Internet Engineering Task Force (February 2012) Work in progress.
23. Woundy, R., Kinnear, K. : Dynamic Host Configuration Protocol (DHCP) Leasequery. RFC 4388, Internet Engineering Task Force (February 2006)
24. Brzozowski, J., Kinnear, K., Volz, B., Zeng, S. : DHCPv6 Leasequery. RFC 5007, Internet Engineering Task Force (September 2007)
25. Cheshire, S. : IPv4 Address Conflict Detection. RFC 5227, Internet Engineering Task Force (July 2008)

26. Bi, J., Yao, G., Halpern, J., Levy-Abegnoli, E. : SAVI for Mixed Address Assignment Methods Scenario. Internet-Draft draft-ietf-savi-mix-01, Internet Engineering Task Force (October 2011) Work in progress.
27. Costa, F., Pougnard, X., LiHongyu, L., Combes, J. : Duplicate Address Detection Proxy. Internet-Draft draft-ietf-6man-dad-proxy-02, Internet Engineering Task Force (March 2012) Work in progress.
28. Bi, J., Wu, J., Yao, G., Baker, F. : SAVI Solution for DHCP. Internet-Draft draft-ietf-savi-dhcp-07, Internet Engineering Task Force (November 2010) Work in progress.
29. Bi, J., Yao, G., Wu, J., Baker, F. : SAVI Solution for Stateless Address. Internet-Draft draft-bi-savi-stateless-00, Internet Engineering Task Force (April 2010) Expired.
30. Bi, J., Yao, G., Halpern, J., Levy-Abegnoli, E. : SAVI for Mixed Address Assignment Methods Scenario. Internet-Draft draft-bi-savi-mix-04, Internet Engineering Task Force (September 2011) Work in progress.
31. An, C., JiahaiYang, J., Wu, J., Bi, J. : Definition of Managed Objects for SAVI Protocol. Internet-Draft draft-an-savi-mib-00, Internet Engineering Task Force (December 2010) Work in progress.
32. Deering, S., Hinden, R. : Internet Protocol, Version 6 (IPv6) Specification. RFC 2460, Internet Engineering Task Force (December 1998)
33. Moore, N. : Optimistic Duplicate Address Detection (DAD) for IPv6. RFC 4429, Internet Engineering Task Force (April 2006)
34. Narten, T., Draves, R., Krishnan, S. : Privacy Extensions for Stateless Address Autoconfiguration in IPv6. RFC 4941, Internet Engineering Task Force (September 2007)
35. Chown, T. : IPv6 Address Accountability Considerations. Internet-Draft draft-chown-v6ops-address-accountability-01, Internet Engineering Task Force (July 2011) Work in progress.
36. IAB, IESG : IETF Policy on Wiretapping. RFC 2804, Internet Engineering Task Force (May 2000)
37. Harrington, D., Presuhn, R., Wijnen, B. : An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks. RFC 3411, Internet Engineering Task Force (December 2002)
38. An, C., JiahaiYang, J., Wu, J., Bi, J. : Definition of Managed Objects for SAVI Protocol. Internet-Draft draft-an-savi-mib-02, Internet Engineering Task Force (December 2011) Work in progress.
39. Kent, S., Seo, K. : Security Architecture for the Internet Protocol. RFC 4301, Internet Engineering Task Force (December 2005)
40. Kaufman, C., Hoffman, P., Nir, Y., Eronen, P. : Internet Key Exchange Protocol Version 2 (IKEv2). RFC 5996, Internet Engineering Task Force (September 2010)
41. Eronen, P., Laganier, J., Madson, C. : IPv6 Configuration in Internet Key Exchange Protocol Version 2 (IKEv2). RFC 5739, Internet Engineering Task Force (February 2010)