Faiblesses d'IPSec en déploiements réels

Yvan Vanhullebus

Netasq
vanhu@netasq.com

Résumé IPSec regroupe une suite de protocoles destinés à sécuriser des échanges sur un réseau non protégé. D'un niveau de sécurité très élevé en théorie, il existe cependant de nombreux problèmes pratiques posés par l'utilisation d'IPSec, parfois dus à la complexité des RFCs ou à des faiblesses des protocoles, parfois à des bugs d'implémentations, et parfois simplement à la méconnaissance des administrateurs.

Nous verrons dans cet article un résumé du fonctionnement d'IPSec, ainsi qu'une étude plus détaillée de plusieurs points de faiblesses potentiels des tunnels IPSec, qu'ils relèvent des protocoles, des implémentations ou des configurations, et nous étudierons les possibilités de solutions à ces faiblesses.

1 Présentation rapide d'IPSec

1.1 Objectifs d'IPSec

IPSec est un ensemble de protocoles et de normes, formalisés par des RFCs (voir Annexe 5.5), destinés à permettre la mise en place de tunnels VPNs[1]. Le support d'IPSec est optionnel pour IPv4, mais obligatoire pour IPv6.

Ces VPNs peuvent par exemple servir à interconnecter à moindre frais des réseaux d'agences au travers d'Internet, à protéger efficacement des liaisons difficilement sécurisables de type Wifi, ou à protéger des protocoles peu sécurisés qu'on ne peut pas facilement remplacer (NFS, etc...).

Cet ensemble de protocoles fournit essentiellement trois fonctionnalités.

Confidentialité Le rôle le plus évident d'IPSec est de chiffrer les données qui transitent, à l'aide d'algorithmes classiques de chiffrement symétriques, fonctionnant par blocs. Les algorithmes les plus courants sont DES, 3DES, Blowfish et AES, généralement utilisés en mode CBC[2], mais cette liste n'est pas exhaustive, et il est possible d'implémenter et utiliser tout algorithme de son choix.

[1] Virtual Private Networks, soit Réseaux Privés Virtuels
[2] Cipher Block Chaining mode, [CBC].

Intégrité Certains modes d'IPSec permettent de garantir également l'intégrité des données transitées, à l'aide d'algorithmes de hash classiques (comme MD5 ou SHA1, cette liste est également extensible). Nous verrons par la suite que cette vérification d'intégrité peut couvrir une partie variable des données transitées.

Authentification La fonctionnalité la moins flagrante d'IPSec, et pourtant indispensable à un bon niveau de sécurité, est la validation de l'identité du correspondant.

Cela permet de s'assurer que le tunnel IPSec est bien établi avec le correspondant légitime.

1.2 Fonctionnement d'IPSec

IPsec se décompose en plusieurs phases.

La première consiste à disposer d'une paire[3] d'IPSec SA (Security Association, un élément qui contient les informations nécessaires pour chiffrer et/ou hasher les données), soit de façon statique, soit en les négociant à l'aide du protocole IKE (Internet Key Exchange, décrit section 1.6).

Une fois cette SA disponible pour chaque correspondant (et identifiée de façon unique par un numéro, appelé SPI), le trafic effectif est chiffré et/ou hashé via les informations de la SA, puis encapsulé dans un protocole dédié.

Le correspondant peut alors procéder aux opérations inverses lors de la réception du paquet encapsulé (la SA est retrouvée grace au SPI, seule information en clair dans les paquets).

Mode tunnel et mode transport Les deux modes de fonctionnement d'IPSec utilisent sensiblement les mêmes mécaniques et algorithmes. La différence essentielle est le niveau d'encapsulation.

Le mode Transport permet d'encapsuler la donnée d'un paquet IP, et d'ajouter un en-tête intermédiaire qui protège cette donnée. L'en-tête IP du paquet est donc inchangé :

```
Avant encapsulation : | IP | Payload |
Après encapsulation : | IP | IPSec header | Payload |
```

Ce mode permet d'encapsuler du trafic entre deux machines pouvant déjà communiquer ensemble, tout en réduisant le surcoût de l'encapsulation IPSec. Le mode Transport ne peut ainsi être utilisé que dans des configurations ou chaque

[3] Les IPSec SAs sont unidirectionnelles, et sont donc généralement utilisées par paires, une pour le trafic entrant, une pour le trafic sortant

extrémité de trafic est également l'extrémité de tunnel.

Le mode Tunnel encapsule l'intégralité du paquet IP dans un nouveau paquet :

```
Avant encapsulation : | IP | Payload |
Après encapsulation : | New IP | IPSec header | IP | Payload |
```

Ce mode d'encapsulation génère un surcoût d'encapsulation plus important, mais permet par exemple l'interconnexion de deux réseaux ne pouvant pas directement communiquer ensemble si chacun dispose d'une passerelle, et si ces deux passerelles peuvent communiquer ensemble. Ce mode est typiquement utilisé dans des configurations « Host to Net » ou « Net to Net » et, en général, dans des configurations dont une des extrémités de trafic n'est pas l'extrémité de tunnel.

Il permet par exemple de relier des réseaux non routables [RFC1918] par l'intermédiaire de passerelles IPSec disposant d'une adresse IP routable.

1.3 Protocoles d'encapsulation

IPSec est prévu pour encapsuler des paquets IP (ou supérieur) dans des paquets IP. Pour cela, deux protocoles d'encapsulation distincts sont disponibles, répondant à des contraintes différentes.

ESP ESP (RFC2406) permet d'encapsuler une donnée de façon chiffrée, et optionnellement hashée. Il fournit ainsi la confidentialité des données, et optionnellement leur intégrité. Seul le SPI de la SA utilisée est directement lisible dans l'en-tête ESP.

```
Transport :

         Données chiffrées
           <--------->
| IP | ESP | Payload |
     <---------------->
         Données hashées

Tunnel :
           Données chiffrées
               <-------------->
| New IP | ESP | IP | Payload |
         <--------------------->
            Données hashées
```

ESP est généralement utilisé en mode Tunnel, mais peut aussi être utilisé en mode transport, par exemple pour chiffrer un trafic précis entre deux correspondants.

AH Le protocole AH (RFC 2402) ne fournit qu'un hash de validation d'intégrité, mais pas de chiffrement. Cependant, à la différence d'ESP, le hash couvre l'ensemble du paquet IP.

```
Transport :
  Données hashées
<------------------->
| IP | AH | Payload |

Tunnel :
       Données hashées
<----------------------------->
| New IP | ESP | IP | Payload |
```

AH est utilisé presque exclusivement en mode Transport.

1.4 ESP+AH

Dans certains cas particuliers, il peut être nécessaire de chiffrer les données ET d'assurer l'intégrité de l'ensemble du paquet, y compris l'en-tête IP. On utilise alors une double encapsulation ESP+AH[4]. Ce cas de figure est cependant très peu utilisé.

1.5 Encapsulation NAT-T UDP

Les protocoles ESP et AH posant de gros problèmes pour passer au travers des équipements faisant du NAT (à cause de l'absence de notion de ports source/destination, et de par l'impossibilité d'associer trafic sortant et trafic entrant), une extension IPSec (Nat-Traversal, RFC 3947 et RFC 3948) propose d'encapsuler l'ESP ou l'AH dans un paquet UDP, plus facile à faire transiter au travers d'équipements de NAT.

1.6 Négociation de clés : IKE

Déployer manuellement des SAs statiques sur des équipements est une opération fastidieuse, difficile à maintenir, pour un résultat finalement peu sûr (voir section 3.1). Une partie d'IPSec permet donc la négociation dynamique de SAs : IKE (Internet Key Exchange, RFC 2409, qui est une implémentation d'ISAKMP, RFC 2408), qui se décompose en plusieurs parties.

[4] Généralement une encapsulation en mode tunnel et une en mode transport. On appelle parfois cette combinaison Tunnel/Transport « Nested mode ».

IKE phase 1 La première partie d'un échange IKE permet aux correspondants de s'authentifier mutuellement, de négocier un ensemble de paramètres (algorithmes de chiffrement, d'authentification, durée de vie, clés de session, etc...) d'une[5] SA spéciale : la IsakmpSA. Cette SA sera alors utilisée pour protéger toutes les communications ultérieures entre les deux correspondantes IKE.

Cette phase peut être effectuée selon deux modes différents : le mode « principal », également appelé « identity protection mode », et le mode « aggressif », plus rapide en nombre de paquets, mais dans lequel plus d'informations transitent sans être chiffrées.

L'authentification mutuelle des correspondants peut être effectuée principalement de deux façons : par un secret pré-partagé, ou par certificats X509.

Durant cet échange, aucun secret ne transite sur le réseau, les clés de sessions communes sont générées par chaque correspondant à l'aide de groupes Diffie-Hellman[DH76].

XAuth XAuth (draft-beaulieu-ike-xauth-02) est une extension IKE permettant un niveau d'authentification supplémentaire entre la phase 1 et la phase 2. XAuth est généralement utilisé pour effectuer des authentifications sur une base RADIUS ou pour utiliser des mots de passes jetables.
Les échanges XAuth sont protégés par la IsakmpSA.

IKE phase 2 Lorsque les correspondants IKE disposent d'une IsakmpSA pour pouvoir protéger leurs communications, ils peuvent alors établir une ou plusieurs IPSec SAs via des négociation de phase2, également appelée « quick mode ». Comme pour la IsakmpSA, plusieurs paramètres sont négociés par les correspondants, comme les algorithmes de chiffrement et de hash, la durée de vie de la SA, etc...

IKE Informational exchanges Dans certains cas, un des correspondants peut avoir à informer l'autre de certains évènements (DELETE-SA), ou simplement vérifier sa présence (Dead Peer Detection). Il utilisera alors un message de type « Informational », qui est protégé par la IsakmpSA de la même façon que les phases 2.

IKE V2 Une nouvelle version du protocole IKE, IKE V2, a récemment été normalisée par l'IETF (RFC 4306). Parmi de nombreuses modifications plus ou moins significatives, il est ici important de signaler en particulier la disparition des deux modes de phase1 au profit d'un seul nouveau mode en 4 échanges (au moins).

[5] Contrairement aux IPSec SAs, la IsakmpSA est bidirectionnelle

Ce nouveau fonctionnement garantit une négociation toujours à l'avantage du répondeur, et permet en particulier de se protéger de dénis de services par consommation de puissance de calcul, puisque l'initiateur devra systématiquement effectuer en premier tout calcul cryptographique lourd.

2 Classification des faiblesses

Nous allons par la suite étudier plusieurs types de faiblesses relatives à IPSec. Il est important de garder à l'esprit les différentes catégories de faiblesses possibles, leur impact potentiel, leur degré de dangerosité, et d'en déduire un risque réel représenté, en fonction du contexte.

2.1 Dénis de services

La faiblesse la plus courante rencontrée lors de déploiements de configurations IPSec est un « simple » déni de service (DOS). Si le niveau de dangerosité direct de ces dénis de services est très modéré (aucune fuite et aucune corruption d'informations), le contexte de déploiement peut parfois rendre ce DOS très génant, par exemple s'il bloque le transit de données importantes (données bancaires, opérations financières, etc....).

2.2 Corruption de données

Le second niveau de dangerosité est le risque de modifications de données sur le chemin sans détection par le destinataire du paquet IPSec.

Les modifications à l'aveugle restent un risque assez faible. En effet, les paquets seront probablement détectés comme étant invalides, soit au niveau IP, soit au niveau applicatif. le risque maximum est donc un déni de service au niveau applicatif, si le programme concerné n'effectue pas assez de vérifications des données qu'il traite.

La possibilité de modifier certaines données de façon précise est un risque beaucoup plus important, puisqu'il permet de changer une information précise et ciblée. Cela nécessite cependant d'être capable de prévoir la donnée d'origine, pour pouvoir la modifier sans avoir pu déchiffrer le paquet IPSec.

2.3 Interception de données

La possibilité de pouvoir lire des données chiffrées est l'un des risques les plus importants à considérer, mais doit être considéré en fonction du contexte.

S'il est impossible de garantir la confidentialité des données sur durée importante (de l'ordre de plusieurs années), ce risque reste généralement peu important.

La plupart des données n'ont en effet qu'une durée d'intérêt assez limitée dans le temps, de l'ordre de la seconde (voire moins) pour les données courantes (le seul intérêt de déchiffrer ces données est de pouvoir les modifier à la volée), à une durée de vie de l'ordre de quelques jours pour des données « modérément sensibles ».

Cependant, certaines données peuvent avoir une durée de validité (donc une durée de confidentialité nécessaire) nettement plus importante. On peut par exemple citer des données bancaires, des données sur une société, ou tout autre donnée qui peut être exploitable à postériori sur une durée importante.

Il existe également des données encore plus délicates à traiter, comme par exemple les données médicales[6], lorsque la durée de confidentialité nécessaire pour ces données dépasse largement les durées de « solidité » de tous les algorithmes symétriques connus et utilisés à ce jour.

Il est donc important, pour évaluer le risque réel d'une interception de données, de tenir compte de la durée de vie de la donnée, et de la comparer au temps nécessaire à l'attaque.

2.4 Accès non autorisés

Le dernier risque principal est la possibilité d'accéder à des ressources sans y être autorisé par la politique de sécurité.

Ici encore, il faut tenir compte du contexte, des prérequis à l'attaque, et comparer en particulier la durée de l'attaque et la durée d'intérêt de l'accès.

3 Faiblesses des protocoles

La suite de protocoles IPSec a été conçue pour répondre à de forts critères de sécurités. Cependant, plusieurs faiblesses intrinsèques de certains de ses protocoles existent, indépendemment des implémentations et configurations.

Ces problèmes sont souvent les plus méconnus, alors qu'il s'agit des seuls qui sont absolument impossibles à corriger.

3.1 IPSec statique

Outre des problèmes de maintenance et d'administration, le fonctionnement IPSec par SAs statiques pose un problème de sécurité, puisqu'il permet potentiellement à un attaquant bien placé d'intercepter assez de trafic chiffré par la

[6] En France, la législation impose de garantir une confidentialité d'au moins 30 ans pour les données médicales

même clé pour commencer des attaques statistiques.

Puisque cette clé n'est jamais changée, l'attaquant pourra alors non seulement lire à postériori tout le trafic capturé, mais pourra aussi désormais lire en temps réel (et éventuellement modifier) tout nouveau trafic protégé par cette clé.

3.2 Mode aggressif et secret pré-partagée

Lors d'une négociation de phase1 en mode aggressif, le hash d'authentification du répondeur n'est pas chiffré (contrairement à une négociation en mode principal).

Il est donc possible pour un observateur sur le chemin (ou pour un attaquant ayant pu intercepter ou deviner assez d'éléments pour émettre un premier paquet acceptable) de récupérer ce hash et d'en déduire la clé par force brute, hors connexion, par exemple à l'aide de l'outil [IKECrack]. Cette attaque est décrite plus en détails dans [PSKAttack].

La plupart des implémentations IKE réagissent également différemment selon la validité de l'identifiant du correspondant (elles ne répondent que si l'identifiant est connu). Il est donc possible d'émettre beaucoup de requêtes avec des identifiants différents pour les tester. Si une attaque exhaustive est difficilement envisageable, il est par exemple beaucoup plus réalisable d'effectuer une présélection d'identifiants possibles (des adresses mail de collaborateurs de la société, etc...), puis de tester ces identifiants.

3.3 Lien entre les clés de sessions

Lors d'une négociation de phase 2, les clés de sessions générées sont dérivées à partir d'informations connues. En réussissant à trouver (par une attaque brute classique) une clé de session pour une SA X, il devient donc plus facile de trouver par la suite la clé de la SA X+1 (c'est à dire la SA générée pour remplacer la SA X lors de son expiration).

Cela rallonge donc la durée de pertinence d'une clé de session, puisque, même après l'expiration de celle-ci, elle peut rester une information intéressante pour accélerer la recherche des clés de sessions ultérieures.

La solution à ce problème est de systématiquement utiliser le PFS[7], qui garantit que chaque clé de session est dérivée uniquement d'une nouvelle opération Diffie-Hellman, et est donc indépendante des autres clés de session.

[7] Perfect Forward Secrecy.

3.4 Permutations de bits ESP

La plupart des algorithmes de chiffrement sont utilisés exclusivement en mode CBC.
Ce mode est connu pour être vulnérable à une attaque dite par « bit flipping », qui permet d'inverser certains bits choisis dans la donnée d'origine, sans pour autant avoir besoin de déchiffrer le paquet.

Dans le cadre d'une utilisation de ces algorithmes pour chiffrer les données, il est donc possible d'inverser certains bits des données chiffrées. Si cette possibilité est inexploitable dans la plupart des cas (mais permet éventuellement des dénis de services d'applications, en envoyant une donnée corrompue), des bulletins d'alerte signalent la possibilité de changer l'adresse IP source et/ou destination d'un paquet encapsulé (en mode Tunnel, donc), et d'espérer obtenir un message ICMP qui serait réémis en clair, et qui permettrait donc éventuellement d'obtenir une partie de la donnée (copiée dans la donnée du message ICMP).

Ce type d'attaque reste cependant délicat à produire : s'il est facile de prédire ou se trouveront les adresses IP du paquet encapsulé sans avoir besoin de le déchiffrer, il est plus délicat de prédire les IPs (donc de pouvoir générer une IP « avantageuse » pour l'attaquant), et il faut également que l'implémentation IPSec recevant le paquet ESP soit assez mauvaise pour ne pas confronter le paquet décapsulé à la police de chiffrement IPSec (et donc de se rendre compte que le paquet ne devrait pas sortir du tunnel).

Une solution simple à ce problème reste de systématiquement activer l'authentification ESP, ce qui permettra de détecter une modification du paquet, et donc de le rejeter immédiatement.

4 Problèmes d'implémentations

Comme tous programmes, les implémentations IKE sont parfois sujettes à des bugs. Les problèmes d'interopérabilité, de fonctionnement et autres sont hors sujet ici, mais certains problèmes d'implémentations peuvent être exploités pour compromettre la sécurité du système.

Si la plupart de ces problèmes sont corrigés par les développeurs, il est important pour tout utilisateur d'avoir conscience de ces risques, et donc de suivre les bulletins d'alerte, pour savoir éventuellement quand faire des mises à jour de sécurité, quand mettre en place des solutions de contournement pour rendre ces failles inutilisables, voire si nécessaire quand désactiver le service, si le risque de compromission est trop important.

4.1 Parsing IKE

IKE est décrit par un ensemble de 3 RFCs (RFC 2407, RFC 2408, RFC 2409), qui se référencent mutuellement. Ces RFCs séparent un protocole générique de

négociation (ISAKMP) et une instanciation de ce protocole (IKE).

Cette séparation implique une lourdeur importante de lecture des RFCs, ainsi qu'une abstraction inutile : les DOIs (Domains Of Interpretations), puisque seul le IPSec DOI est effectivement utilisé. Toute cette complexité contribue à rendre difficile une implémentation d'IKE propre, interopérable et exempte de bugs.

De plus, la nature des champs IKE (pour la plupart binaires, de taille variable, certains obligatoires ou optionnels en fonction du contexte) rend leur analyse sujette à des bugs d'implémentations.

De nombreuses vulnérabilités ont été relevées au fil des ans sur la plupart des implémentations IKE. La dernière série de vulnérabilités a été remontée par la suite de tests [PROTOS] en 2005, et a mis en évidence des dénis de services sur la plupart des implémentations (CVE-2005-3671 pour *SWAN, CVE-2005-3669 pour Cisco, etc...). la majorité de ces dénis de services étaient assez simples à obtenir, puisqu'ils nécessitaient souvent uniquement d'émettre un premier paquet de négociation de phase 1 (dans certains cas, il n'était même pas nécessaire que les propositions et identifiants soient valides) mais disposant d'une malformation spécifique (en-tête obligatoire absent, champ de données plus important qu'annoncé, etc...).

Si la majeure partie de ces failles se limitent à rendre possible un déni de service (en fonction de la configuration), et si l'exploitation d'un dépassement de tampon pour exécuter du code à distance est rendue assez complexe par les traitements et par le fait qu'IKE soit basé sur le protocole UDP, il a cependant déjà existé par le passé des failles beaucoup plus importantes, telle que [CISCO64424], qui pouvait permettre à un utilisateur non autorisé d'établir un tunnel IPSec avec la passerelle, voire qui permettaient de l'exécution de code à distance sur la passerelle IPSec (Checkpoint : CVE-2004-0469)

L'impact de telles failles est relativement limité quand leur exploitation nécessite un accès légitime, mais devient critique dès lors qu'elles peuvent être exploitées par une personne ne disposant pas d'un tel accès.

4.2 Validations incomplètes de certificats

La plupart des implémentations IKE ont eu au moins un problème de gestion de certificats lors de la phase1.

Mauvaise gestion des CRLs Lorsqu'une implémentation supporte mal (voire pas du tout) les validations de certificats par rapport aux CRLs[8], il est alors possible pour un utilisateur révoqué (qu'il s'agisse d'un attaquant disposant d'un

[8] Certificate Revocation List, une liste de certificats valides, mais pourtant révoqués.

certificat compromis ou d'un utilisateur légitime révoqué pour d'autres raisons) d'avoir un accès qui devrait pourtant lui être interdit.

Si l'absence totale de support des CRLs est désormais rarissime, certaines implémentations considèrent encore qu'un certificat révoqué ne doit générer qu'un avertissement dans les logs, et l'acceptent quand meme pour la négociation.

Erreurs de validation de certificats Plusieurs problèmes d'implémentation de la validation de certificats ont engendré des failles dans des implémentations IKE.

On peut par exemple citer des erreurs de validations de la CRL (racoon : CAN-2004-0607), qui permettent d'établir un tunnel avec un certificat révoqué.

Des erreurs de validation du certificat (*SWAN : CAN-2004-0590, Cisco : CVE-2002-1106) sont beaucoup plus importantes, puisqu'elles permettent d'établir un tunnel avec un faux certificat, et ne nécessitent même pas de certificat valide révoqué.

4.3 Configurations par défaut

Certaines implémentations proposent des valeurs par défaut qui baissent notablement le niveau de sécurité des configurations. Quelques exemples flagrants sont :

Paramètres de négociation Les configurations faibles par défaut concernent essentiellement les paramètres de négociation, de phase 1 et/ou de phase 2.

Le premier cas très fréquent est une durée de vie par défaut particulièrement élevée (parfois de l'ordre de la semaine). De telles durées de vie rendent les clés de sessions plus vulnérables à des attaques statistiques, en fonction des algorithmes choisis et des volumes de trafic.

Le second cas très fréquent est la liste des algorithmes proposés par défaut, liste dans laquelle est très souvent inclus le DES, algorithme pourtant considéré comme « faible » depuis plusieurs années, et plus sujet aux attaques que la plupart des autres algorithmes, comme Blowfish, AES, ou meme 3DES.
Outre le fait que cet algorithme est proposé dans la liste par défaut de certaines implémentations, il est parfois même listé en premier, constituant ainsi la première proposition que le correspondant évaluera. Or, de nombreuses implémentations choisissent la première proposition valide qui leur est faite, et deux implémentations peuvent donc négocier une phase en DES alors qu'elles avaient toutes les deux d'autres algorithmes plus résistants dans leur configuration.

Le troisième cas de configuration faible par défaut est l'utilisation d'ESP sans algorithme d'authentification, qui rend alors les paquets ESP sujets à des attaques par « bit flipping » (voir 3.4).

La désactivation par défaut du PFS est le dernier cas fréquent de configuration faible par défaut, et rend les tunnels utilisés longuement sensibles à une attaque des clés de sessions (voir 3.3).

Modes d'authentification faibles Certains modes d'authentification peuvent s'avérer plus faibles que d'autres. En particulier, l'utilisation de l'extension XAuth est souvent faite sans une réelle compréhension de ses mécanismes de sécurité.

Le draft d'XAuth stipule bien que toute la sécurité d'XAuth se repose sur le fait que cette partie de la négociation est protégée par la IsakmpSA. Or, en pratique, la plupart des administrateurs considèrent que l'authentification est réellement faite au niveau d'XAuth, et donc utilisent une configuration de phase 1 simpliste et faible, généralement en utilisant un « group password », qui est en fait une clé pré-partagée commune à tous les utilisateurs nomades.

La configuration est donc doublement affaiblie, d'abord par l'utilisation du mode aggressif en clés prépartagées, ensuite par l'utilisation d'un secret faible et commun à tous les correspondants.

Contrôles faibles de la négociation Certains démons permettent de paramétrer la souplesse de négociation. Par exemple, dans le démon *racoon* d'*ipsec-tools*, il est possible de spécifier la souplesse de négociation parmi les possibilités « exact » (rigoureusement la configuration locale), « strict » (propositions d'un niveau supérieur ou égal à la configuration locale), « claim » (comme « strict », mais tente de négocier des durées de vie innapropriées), ou « obey » (accepte systématiquement la première proposition du correspondant).

Or, dans les exemples de configuration fournis, toutes les configurations étaient en « obey » il y a encore quelques mois. Les commentaires « for testing only » n'avaient probablement une grande efficacité pour inciter les administrateurs à changer cette configuration.

4.4 Problèmes dans les piles IPSec

La gestion des paquets ESP et AH est également susceptible de poser des problèmes de fonctionnement et/ou de sécurité plus ou moins graves.

Validations incomplètes Lors de la mise en place d'une politique IPSec, tout paquet doit être confronté à cette politique.

Les paquets sortants doivent y être confrontés pour savoir s'ils doivent être envoyés au travers d'un tunnel. Le fait d'encapsuler un paquet qui ne devrait pas l'être représente un risque modéré : diffusion d'informations à un correspondant IPSec (qui va probablement jeter ce paquet dès sa sortie du tunnel), et dysfonctionnement du service (le paquet émis dans le tunnel n'arrivera pas à sa destination normale).

Le risque le plus important est donc l'envoi en clair sur le réseau de paquets qui devraient être envoyés dans un tunnel, comme c'était le cas pour la faille OpenBSD CVE-1999-0727. Cependant, ce genre de failles est généralement non provocable par un attaquant, qui ne peut qu'espérer que le problème se produira « naturellement ».

Le second risque important concerne la réception de paquets depuis un tunnel.

Validation de la politique de sécurité incomplète (Linux), qui permet d'envoyer au travers d'un tunnel IPSec un paquet qui ne correspond pas au trafic de ce tunnel.

Failles dans le traitement des paquets La gestion des paquets IPSec (ESP, AH et IPComp) est un processus complexe, qui pose parfois des problèmes d'implémentation. S'il est nécessaire d'être un correspondant légitime pour pouvoir exploiter ces bugs, le risque reste limité. Cependant, il suffit parfois de pouvoir envoyer un paquet vers la passerelle, ou au travers de celle ci pour pouvoir engendrer des dénis de service.
C'est par exemple le cas d'un bug de la pile IPSec KAME, où un test fait à l'envers faisait un appel à *panic*[9] lorsqu'elle faisait transiter certains paquets AH spécialement construits.

Dans d'autres cas, le problème peut être encore plus grave, comme par exemple la faille OpenBSD CVE-2001-0284, qui conçernait aussi des paquets AH, mais qui permettait cette fois une exécution à distance de code malicieux.

4.5 Configurations trop complexes

Outre la complexité intrinsèque d'IPSec, plusieurs implémentations ont des mécaniques et syntaxes qui rendent la compréhension des configurations encore plus complexe.

Dans certains cas, c'est l'apparente simplicité qui engendre des problématiques de sécurité. La configuration de VPNs sur Checkpoint génère automatiquement un réseau « virtuel » contenant l'ensemble des extrémités de trafic concernées par

[9] Un panic est un appel système spécial, qui arrête immédiatement le fonctionnement du noyau, et se met dans un mode qui facilite le débuggage des problèmes.

le tunnel. Dans des configurations ou les plans d'adressage ne sont pas contigus, le réseau ainsi négocié peut être nettement plus large que ce que veut l'administrateur.

Dans les cas de configurations par fichiers, les configurations sont souvent complètes, mais complexes à mettre en oeuvre.

OpenBSD : Isakmpd Le fichier de configuration du démon d'OpenBSD, Isakmpd, est par exemple construit par de nombreuses sections détaillant certains éléments de la configuration, et référencés entre eux. Une configuration triviale entre deux correspondants nécessitera au minimum une dizaine de sections, plus le remplissage d'un autre fichier de configuration, *isakmpd.policy*, qui peut fournir des possibilités de configuration d'accès très poussées, mais qui est très difficilement compréhensible.

```
<<<<<<<<<< begin isakmpd.conf example
[General]
Retransmits= 5
Exchange-max-time= 120
Listen-on= 172.16.1.1

[Phase 1]
172.17.1.1= local-remote

[local-remote]
Phase= 1
Transport= udp
Local-address= 172.16.1.1
Address= 172.17.1.1
Configuration= Default-main-mode
Authentication= XXXXX_SHARED_SECRET_GOES_HERE_XXXXX

[Phase 2]
Connections= VPN-local-remote

[VPN-local-remote]
Phase= 2
ISAKMP-peer= local-remote
Configuration= Default-quick-mode
Local-ID= network-local
Remote-ID= network-remote

[network-local]
ID-type= IPV4_ADDR_SUBNET
Network= 192.168.1.0
```

```
Netmask= 255.255.255.0

[network-remote]
ID-type= IPV4_ADDR_SUBNET
Network= 10.0.0.0
Netmask= 255.0.0.0

[Default-main-mode]
DOI= IPSEC
EXCHANGE_TYPE= ID_PROT
Transforms= 3DES-SHA

[Default-quick-mode]
DOI= IPSEC
EXCHANGE_TYPE= QUICK_MODE
Suites= QM-ESP-3DES-SHA-SUITE
>>>>>>>>>> end isakmpd.conf

<<<<<<<<<< begin isakmpd.policy
Keynote-version: 2
Authorizer: "POLICY"
Conditions: app_domain == "IPsec policy" &&
            esp_present == "yes" &&
            esp_enc_alg != "null" -> "true";
>>>>>>>>>> end isakmpd.policy
```

En pratique, la complexité du *isakmpd.policy* fait que la plupart des administrateurs vont tout simplement faire une configuration « pass all » :

```
<<<<<<<<<< begin isakmpd.policy
Authorizer: "POLICY"
Comment: Ce filtre accepte n'importe quel certificat
>>>>>>>>>> end isakmpd.policy
```

KAME / Linux 2.6 La configuration de la pile IPSec KAME (fournie avec FreeBSD et NetBSD), identique à celle des versions récentes de Linux, pose elle aussi ses problèmes de compréhension. La principale difficulté ici est l'apparente décorrélation entre la police de sécurité, la configuration des phases 1 et la configuration des phases 2.

```
<<<<<<<<<< begin racoon.conf
remote GW_remote
{
    # Phase 1
    exchange_mode main;
```

```
    lifetime time 3600 sec;
    proposal_check strict;

    proposal{
        encryption_algorithm aes 128;
        hash_algorithm sha1;
        dh_group 2;
        authentication_method pre_shared_key;
    }
}

sainfo address net_local/mask any address net_remote/mask any
{
    # Phase 2, SA sortante
    lifetime time 3600 sec;
    encryption_algorithm aes 128, blowfish 128;
    authentication_algorithm hmac_sha1, hmac_md5;
    pfs_group 2;
    compression_algorithm deflate;
}

sainfo address net_remote/mask any address net_local/mask any
{
    # Phase 2, SA entrante
    lifetime time 3600 sec;
    encryption_algorithm aes 128, blowfish 128;
    authentication_algorithm hmac_sha1, hmac_md5;
    pfs_group 2;
    compression_algorithm deflate;
}
>>>>>>>>>> end racoon.conf

<<<<<<<<<< begin spd

spdadd net_local/mask net_remote/mask any -P out ipsec \
                esp/tunnel/GW_locale-GW_remote/unique;
spdadd net_remote/mask net_local/mask any -P in ipsec  \
                esp/tunnel/GW_remote-GW_locale/unique;

>>>>>>>>>> end spd
```

4.6 Problèmes de cryptographie

La sécurité d'IPSec se base sur de nombreux principes cryptographiques, qu'il est important de ne pas oublier. En effet, toute faiblesse de conception ou d'implémentation d'une de ces briques cryptographiques peut affaiblir la sécurité

des configurations IPSec.

Comme exemple de faiblesses de ce type, on peut citer une faille du client VPN CISCO (CVE-2002-1107), qui générait des aléas faibles (donc prévisibles). Ces aléas faibles le rendaient sensible à des attaques de type usurpation d'identité.

4.7 Problèmes annexes

Une pile IPSec fonctionne dans un environnement, et en particulier sur un système d'exploitation, qui fournit souvent d'autres services. Lors d'un déploiement spécifique d'IPSec sur un système d'exploitation « classique », d'autres failles du système ou de ses outils peuvent compromettre l'ensemble de la passerelle, donc l'accès IPSec.

De même, sur les postes nomades, il y a parfois des possibilités de récupérer des informations normalement non accessibles. On peut par exemple citer la faille CVE-2002-1105 (CISCO), qui permet d'obtenir le mot de passe de groupe de la configuration IPSec sur le poste client.

5 Problèmes de déploiements

Tout déploiement de tunnels IPSec, quelle que soit l'implémentation, reste facile à affaiblir avec une mauvaise configuration. Il est alors important pour les administrateurs de connaitre tous ces pièges courants pour pouvoir aisément les éviter.

5.1 Tunnel sans filtrage

De nombreux administrateurs considèrent que l'établissement d'un tunnel IPSec (ou d'un tunnel VPN en général) est un gage de sécurité suffisant pour garantir tous les flux circulant au travers de ce tunnel. Or, si un tunnel IPSec peut garantir le transit des flux au travers de ce tunnel, il ne peut en aucun cas garantir la non-compromission des extrémités de trafic distantes.

Il est donc indispensable de mettre en place une politique de filtrage sur les flux des tunnels IPSec, et de les réduire au strict nécessaire, de la même facon que sont controlés les autres flux de la passerelle.

5.2 Mauvaise compréhension des configurations

Dans certains cas, il est possible de créer des configurations qui ne correspondent pas à ce qui semble naturel pour l'administrateur.
Par exemple, dans la configuration de la pile IPSec KAME, tous les tutoriels et tous les fichiers d'exemples indiquent de déclarer des polices de sécurité avec

le mot clé « require ». Or, dans le cas d'un correspondant IPSec avec lequel on désirerait établir plusieurs phases2 de niveaux de sécurité différents (algorithmes, durées de vies, etc...), le mot clé « require » indique que n'importe laquelle des SAs négociées avec le correspondant peut être utilisée pour chiffrer un paquet de n'importe laquelle des polices de sécurité vers ce correspondant.
Dans le pire des cas, le traffic censé être le mieux protégé peut finalement se retrouver chiffré avec la SA la plus faible.

5.3 Secrets mal protégés

Cette faiblesse potentielle est probablement l'une des plus génériques à tout contexte de sécurité, mais la sécurité d'un tunnel IPSec est directement dépendante de la qualité de ses secrets.

Il est évident qu'un tunnel IPSec dont la clé prépartagée ou une clé privée de certificat est facilement accessible présente un niveau de sécurité quasi nul pour toute personne pouvant disposer de ce secret et capable d'intercepter le traffic IPSec.

5.4 Mauvaise gestion de PKI

La sécurité d'une architecture IPSec par certificats dépend également beaucoup de la capacité de gérer correctement la chaine de certificats.

La première faiblesse courante est une gestion mauvaise ou absente des listes de révocations (CRLs) de la CA. Il devient alors possible de s'authentifier sur une passerelle IPSec avec un certificat pourtant connu comme étant révoqué.

La seconde faiblesse importante est une mauvaise gestion de la clé privée de la CA qui permettrait, d'une façon ou d'une autre (copie de la clé privée de la CA, etc...), de signer des certificats « illégitimes », permettant alors également de s'authentifier sur une passerelle IPSec sans pourtant en avoir le droit légitime.

5.5 Configurations faibles

Des configurations faibles sont très fréquemment rencontrées lors de déploiements effectifs, parfois par manque de connaissance de l'administrateur, parfois par incompétence ou par facilité. Ces configurations faibles facilitent ou permettent souvent l'exploitation de failles vues précédemment, qu'elles soient des faiblesses de protocoles ou des faiblesses courantes d'implémentations.

Parmi les configurations faibles, les plus courantes sont :
- Phase1 en mode aggressif et secret pré-partagé (voir section 3.2).
- Utilisation d'algorithmes faibles : DES, MD5, pas de PFS pour les phases 2, etc...

- ESP sans hash, qui le rend sensible aux attaques par bit flipping CBC, voir [CVE-2005-0039].
- Secret partagés faibles.
- Mauvaise configuration de CAs.
- Pas de gestion de CRLs.

Les RFCs IPSec

Les implémenations actuelles d'IPSec sont basées essentiellement sur tout ou partie des RFCs suivantes :

- RFC 2367 : PFKEY Key Management API, Version 2.
- RFC2401 : Security Architecture for the Internet Protocol.
- RFC2402 : IP Authentication Header.
- RFC2403 : The Use of HMAC-MD5-96 within ESP and AH.
- RFC2404 : The Use of HMAC-SHA-1-96 within ESP and AH.
- RFC2405 : The ESP DES-CBC Cipher Algorithm With Explicit IV.
- RFC2406 : IP Encapsulating Security Payload (ESP).
- RFC2407 : The Internet IP Security Domain of Interpretation for ISAKMP.
- RFC2408 : Internet Security Association and Key Management Protocol (ISAKMP).
- RFC2409 : The Internet Key Exchange (IKE).
- RFC2410 : The NULL Encryption Algorithm and Its Use With IPsec.
- RFC2411 : IP Security Document Roadmap.
- RFC2412 : The OAKLEY Key Determination Protocol.
- RFC2451 : The ESP CBC-Mode Cipher Algorithms.
- RFC2857 : The Use of HMAC-RIPEMD-160-96 within ESP and AH.
- RFC3706 : A Traffic-Based Method of Detecting Dead Internet Key Exchange (IKE) Peers.
- RFC3947 : Negotiation of NAT-Traversal in the IKE.
- RFC3948 : UDP Encapsulation of IPsec ESP Packets.

Très récemment, de nouvelles RFCs sont sorties pour la « nouvelle » version d'IPSec :

- RFC 4301 : Security Architecture for the Internet Protocol.
- RFC 4302 : IP Authentication Header.
- RFC 4303 : IP Encapsulating Security Payload (ESP).
- RFC 4304 : Extended Sequence Number (ESN) Addendum to IPsec Domain of Interpretation (DOI) for Internet Security Association and Key Management Protocol (ISAKMP).
- RFC 4305 : Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH).
- RFC 4306 : Internet Key Exchange (IKEv2) Protocol.
- RFC 4307 : Cryptographic Algorithms for Use in the Internet Key Exchange Version 2 (IKEv2).
- RFC 4308 : Cryptographic Suites for IPsec.

– RFC 4309 : Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP).

Références

[PSKAttack] Michael Thumann, PSK Cracking using IKE Aggressive mode

[IKECrack] Anton T. Rager, IKE cracking tool, `http://ikecrack.sourceforge.net`

[PROTOS] University of Oulu, Protos IKE test suite, `http://www.ee.oulu.fi/research/ouspg/protos/testing/c09/isakmp,2005`

[CISCO64424] CISCO Systems, Vulnerabilities in the Internet Key Exchange Xauth Implementation, 2005.

[CVE-2005-0039] CVE, bit flipping weakness of CBC mode encryption.

[RFC1918] RFC 1918, Address Allocation for Private Internets

[CBC] NIST, Modes of operations for symetric key block ciphers.

[DH76] W. Diffie and M.E. Hellman, New directions in cryptography, IEEE Transactions on Information Theory 22 (1976), 644-654.