Groupe de travail Réseau
Request for Comments : 3631
Catégorie : Information
Traduction Claude Brière de L'Isle

S. Bellovin, IAB
J. Schiller, IAB
C. Kaufman, IAB

décembre 2003

Mécanismes de sécurité pour l'Internet

Statut de ce mémoire

Le présent mémoire apporte des informations pour la communauté de l'Internet. Il ne spécifie aucune sorte de norme de l'Internet. La distribution du présent mémoire n'est soumise à aucune restriction.

Résumé

La sécurité doit être incorporée aux protocoles de l'Internet pour que ces protocoles offrent leurs services en toute sécurité. De nombreux problèmes de sécurité peuvent être imputés à des mises en œuvre inappropriées. Cependant, même une mise en œuvre correcte va avoir des problèmes de sécurité si le protocole fondamental est lui-même exploitable. Comment la sécurité devrait exactement être mise en œuvre dans un protocole va varier, à cause de la structure du protocole lui-même. Cependant, il y a beaucoup de protocoles pour lesquels les mécanismes déjà développés de sécurité standard de l'Internet, peuvent être applicables. Celui qui est précisément approprié dans une situation donnée peut varier. Nous allons passer en revue un certain nombre de différente possibilités, en expliquant les propriétés de chacune.

Table des matières

1. Introduction

La compromission de la sécurité de l'Internet peut être divisée en plusieurs classes, allant du déni de service à la compromission de l'hôte. Les attaques de déni de service fondées sur le pur volume de trafic sortent du domaine d'application du présent document, bien qu'elles soient le sujet de beaucoup de discussions et recherches actuelles. Il est important de noter que beaucoup de ces attaques sont rendues plus difficiles par de bonnes pratiques de sécurité. La compromission d'un hôte (très couramment causée par des débordements de mémoire tampon non détectés) représente des fautes de mises en œuvre individuelles plus que des fautes dans les protocoles. Néanmoins, des protocoles conçus avec soin peuvent rendre la survenance de telles fautes moins probables et plus difficiles à exploiter.

Cependant, il y a des compromissions de la sécurité qui sont facilitées par les protocoles mêmes qui sont utilisés sur l'Internet. Si un problème de sécurité est inhérent à un protocole, aucune manière de le mettre en œuvre ne sera capable d'empêcher le problème.

Il est donc d'une importance vitale que les protocoles développés pour l'Internet fournissent cette sécurité fondamentale.

Comment exactement un protocole devrait être sécurisé dépend du protocole lui-même ainsi que des besoins de sécurité du protocole. Cependant, nous avons développé un certain nombre de mécanismes de sécurité standard à l'IETF. Dans de nombreux cas, une application appropriée de ces mécanismes peut fournir la sécurité nécessaire pour un protocole.

Un certain nombre de mécanismes possibles peuvent être utilisés pour fournir la sécurité sur l'Internet. Ceux qui devraient être sélectionnés dépend de nombreux facteurs différents. On tente ici de fournir des conseils, en énumérant les facteurs et les solutions actuellement normalisées (ou sur le point de l'être) comme on l'a discuté à l'atelier de l'IAB sur l'architecture de sécurité [RFC2316].

La sécurité est cependant un art, pas une science. Tenter d'appliquer aveuglément une recette peut conduire à un désastre. Comme toujours, le bon sens devrait prévaloir dans la conception des protocoles.

Finalement, les mécanismes de sécurité ne sont pas une poudre magique qui peut être saupoudrée par dessus des protocoles achevés. Il est rare que la sécurité puisse être rajoutée au produit fini. De bons concepts – c'est-à-dire, des concepts sûrs, propres, et efficaces – se font jour lorsque les mécanismes de sécurité sont conçus en même temps que le protocole. Aucun entraînement concevable en cryptographie ne peut sécuriser un protocole qui a des hypothèses sémantiques fautives.

2. Facteurs de décision

2.1 Modèle de menace

Le facteur le plus important pour choisir un mécanisme de sécurité est le modèle de menace. C'est-à-dire, de qui peut-on s'attendre à une attaque, contre quelles ressources, en utilisant quelles sortes de mécanismes ? Une cible de faible valeur, comme un site de la Toile qui offre seulement des informations publiques, peut ne pas mériter beaucoup de protection. À l'inverse, une ressource qui si elle est compromise, pourrait exposer des parties significatives de l'infrastructure de l'Internet, disons, un routeur majeur du cœur du réseau ou un serveur de noms de domaines de haut niveau, devrait être protégé par des mécanismes très puissants. La valeur d'une cible pour un attaquant dépend du propos de l'attaque. Si le propos est d'accéder à des informations sensibles, tous les systèmes qui traitent ces informations ou servent d'intermédiaires pour y accéder sont précieux. Si l'objet est de faire des ravages, les systèmes sur lesquels dépendent de large parts de l'Internet sont d'une valeur extrême. Même si seules des informations publiques sont publiées sur un site de la Toile, changer son contenu peut causer bien des embarras à son propriétaire et pourrait résulter en des dommages substantiels. Il est difficile lors de la conception d'un protocole de prédire quelle utilisation aura un jour ce protocole.

Tous les systèmes connectés à l'Internet exigent une quantité minimum de protection. Commencé en 2000 et qui se poursuit à présent, nous avons été témoins de l'avènement d'un nouveau type d'attaque contre la sécurité de l'Internet : un programme de "ver" Internet qui cherche et attaque automatiquement les systèmes qui sont vulnérables à la compromission via un certain nombre d'attaques incorporées dans le programme du ver lui-même. Ces vers peuvent compromettre littéralement des milliers de systèmes en très peu de temps. Noter que le premier ver Internet était le ver "Morris" de 1988. Cependant, il n'a pas été suivi par des programmes similaires pendant plus de 12 ans !

Au moment de la rédaction du présent document, tous ces vers ont tiré parti des erreurs de programmation dans la mise en œuvre de protocoles par ailleurs raisonnablement sûrs. Cependant, il n'est pas difficile d'envisager une attaque qui ciblerait une faute de sécurité fondamentale dans un protocole largement déployé. Il est donc impératif que nous nous efforcions de minimiser de telles fautes dans les protocoles que nous concevons.

La valeur d'une cible pour un attaquant peut dépendre de sa localisation. Une station de surveillance du réseau qui est physiquement sur un câble de cœur de réseau est une cible majeure, car elle pourrait aisément être transformée en station d'espionnage. La même machine, si elle es située sur un réseau d'extrémité et utilisée pour du traitement de texte, serait de beaucoup moindre utilité pour un attaquant sophistiqué, et donc courra un risque significativement moindre.

On doit aussi considérer à quelles sortes d'attaques on peut s'attendre. Au minimum, l'espionnage doit être vu comme une menace sérieuse ; il y a eu de très nombreux incidents de cette sorte depuis au moins 1993. Souvent, des attaques actives, c'est-à-dire, des attaques qui impliquent l'insertion ou la suppression de paquets par l'attaquant, sont aussi un risque. Il vaut de noter que de telles attaques peuvent être lancées avec des outils du commerce, et ont en fait été observées "sur le vif". D'un intérêt particulier est une forme d'attaque appelée "capture de session", où quelqu'un sur une liaison entre les deux parties communicantes attend que l'authentification soit terminée pour se faire passer pour une des parties et continuer la connexion avec l'autre.

Un des plus importants outils disponibles pour sécuriser les protocoles est le chiffrement. La cryptographie permet d'appliquer diverses sortes de protection aux données lorsque elles traversent le réseau, sans avoir à dépendre d'aucune propriété de sécurité particulière du réseau lui-même. C'est important parce que l'Internet, par sa gestion et son contrôle répartis, ne peut pas être considéré par lui-même comme un support de confiance. Sa sécurité découle des mécanismes que l'on construit dans les protocoles eux-mêmes, indépendamment du support ou des opérateurs de réseaux sous-jacents.

Finalement, bien sûr, il y a le coût de l'utilisation de la cryptographie pour le défenseur. Ce coût chute rapidement ; la loi de Moore, plus la facile disponibilité des composants et trousses à outils cryptographiques, rend relativement facile d'utiliser des techniques de protection fortes. Bien qu'il y ait des exceptions, le fonctionnement des clés publiques est toujours coûteux, peut-être, prohibitif, de sorte que si le coût de chaque opération de clé publique est réparti sur trop peu de transactions, une conception attentive de l'ingénierie peut généralement nous permettre d'étaler ces coûts sur de nombreuses transactions.

En général, le comportement par défaut d'aujourd'hui devrait être d'utiliser le plus fort chiffrement disponible dans tout protocole. Une cryptographie forte ne coûte souvent pas plus cher, et parfois moins, qu'une cryptographie plus faible. Le coût réel de performance d'un algorithme est souvent sans relation avec la sécurité qu'il fournit. Selon le matériel disponible, le chiffrement peut être effectué à de très hauts débits (1+Gbit/s) et même dans les logiciels son impact en performances se rétrécit au fil du temps.

2.2 Un mot sur les mécanismes obligatoires

L'IETF a fait évoluer la notion de mécanisme de "mise en œuvre obligatoire". Cette philosophie a évolué à partir de notre principal désir de s'assurer de l'interopérabilité entre des mises en œuvre différentes d'un protocole. Si un protocole offre de nombreuses options pour la façon d'effectuer une tâche particulière, mais ne réussit pas à en fournir au moins une que tous doivent mettre en œuvre, il est possible que plusieurs mises en œuvre non interopérables en résulte. C'est la conséquence du choix de mécanismes ne se recoupant pas qui sont déployés dans les différentes mises en œuvre.

Bien qu'un certain protocole puisse faire usage d'un seul ou de quelques uns des mécanismes de sécurité, ces mécanismes eux-mêmes peuvent souvent utiliser plusieurs systèmes cryptographiques. Les divers systèmes cryptographiques ont des forces et des performances variées. Cependant, dans de nombreux protocoles il est nécessaire de spécifier un "de mise en œuvre obligatoire" pour assurer que deux mises en œuvre au hasard vont finalement être capables de négocier un système cryptographique commun entre elles.

Il y a certains protocoles qui ont été conçus à l'origine pour fonctionner dans un domaine très limité. Il est souvent avancé que le domaine de mise en œuvre d'un certain protocole est suffisamment bien défini et sûr pour que le protocole lui-même n'ait pas besoin de fournir de mécanisme de sécurité.

L'histoire a montré que cet argument est faux. Inévitablement, des protocoles réussis – même si ils ont été développés pour un usage limité – flanchent dans un environnement plus large, où les hypothèses initiales de sécurité ne tiennent plus.

Pour résoudre ce problème, l'IETF exige que *TOUS* les protocoles fournissent les mécanismes de sécurité appropriés, même lorsque leur domaine d'application est au premier abord très limité.

Il est important de comprendre que les mécanismes obligatoires sont obligatoires *à mettre en œuvre*. Il n'est pas nécessairement obligatoire que l'utilisateur final utilise réellement ces mécanismes. Si un utilisateur final sait qu'on déploie un protocole sur un réseau "sûr", il peut alors choisir de désactiver les mécanismes de sécurité dont ils estime qu'ils ont une valeur ajouté insuffisante comparée au coût de leur performance. (On est généralement sceptique sur la sagesse d'une décision de désactivation d'une sécurité forte même dans ce cas, mais cela sort du cadre du présent document.)

Insister pour que certains mécanismes soient de mise en œuvre obligatoire signifie que les utilisateurs finaux qui ont besoin du protocole fourni par le mécanisme de sécurité le trouvent disponible quand ils en ont besoin. Particulièrement avec les mécanismes de sécurité, juste parce qu'un mécanisme est de mise en œuvre obligatoire n'implique pas qu'il devrait être le mécanisme par défaut ou qu'il ne peut pas être désactivé par configuration. Si un algorithme de mise en œuvre obligatoire est vieux et faible, il vaut mieux le désactiver lorsque un algorithme plus fort devient disponible.

2.3 Granularité de la protection

Certains mécanismes de sécurité peuvent protéger un réseau entier. Bien que cela fasse des économies sur le matériel, cela peut laisser l'intérieur de tels réseaux ouverts à des attaques en interne. D'autres mécanismes peuvent fournir une protection jusqu'à l'utilisateur individuel d'une machine en temps partagé, bien que peut-être au risque que quelqu'un se fasse passer pour un usager légitime si la machine a été compromise.

Pour fixer la granularité de protection désirée, les concepteurs de protocoles devraient tenir compte des schémas d'usage probables, des couches de mise en œuvre (voir ci-dessous) et de la capacité de déploiement. Si un protocole va vraisemblablement être utilisé seulement au sein d'un groupe de machines sûres (disons par exemple, un centre d'opérations réseau) la granularité du sous-réseau peut être appropriée. À l'opposé, un mécanisme de sécurité particulier à une seule application ferait mieux d'être incorporé dans cette application, plutôt que dans TCP ; autrement, le déploiement sera très difficile.

2.4 Couche de mise en œuvre

Les mécanismes de sécurité peuvent être situés dans toute couche. En général, mettre un mécanisme dans une couche inférieure protège une plus grande variété de protocoles de couche supérieure, mais peut n'être pas capable de les protéger aussi bien. Un chiffreur de couche liaison peut protéger non seulement les paquets IP, mais aussi les paquets ARP. Cependant, sa portée est cette seule liaison là. À l'inverse, un message électronique signé est protégé même si il est envoyé à travers de nombreuses passerelles de messagerie à livraison différée, il peut identifier l'envoyeur réel, et la signature peut être vérifiée longtemps après la livraison du message. Cependant, ce seul type de message est protégé. Les messages de format similaire, comme certains envois de Netnews, ne sont pas protégés sauf si le mécanisme est spécifiquement adapté et ensuite mis en œuvre dans les programmes de traitement des nouvelles.

3. Mécanismes de sécurité standard

3.1 Mots de passe à usage unique

Les schémas de mot de passe à utilisation unique, tels que ceux décrits dans la [RFC2289], sont bien plus forts que les mots de passe conventionnels. L'hôte n'a pas besoin de mémoriser une copie du mot de passe de l'usager, pas plus qu'il n'est jamais transmis sur le réseau. Cependant, il y a quelques risques. Comme la chaîne transmise est déduite d'un mot de passe caractéristique d'un utilisateur, des attaques par devinette sont quand même possibles. (Bien sûr, un programme pour lancer cette attaque est déjà disponible.) De plus, la capacité de l'usager à se connecter expire nécessairement après un nombre prédéterminé d'utilisations. Alors que dans de nombreux cas, ceci est une caractéristique donnée, une mise en œuvre a très vraisemblablement besoin de fournir un moyen pour réinitialiser la base de données d'authentification, sans exiger que le nouveau mot de passe soit envoyé en clair à travers le réseau.

Il y a des jetons d'authentification de matériel commercial. À part la question de la capture de session, la prise en charge de tels jetons (en particulier, les jetons de mise au défi/réponse, où le serveur envoie un nombre aléatoire différent pour chaque tentative d'authentification) peut exiger des messages de protocole supplémentaires.

3.2 HMAC

HMAC [RFC2104] est la technique préférée d'authentification de secret partagé. Si les deux côtés connaissent la même clé secrète, HMAC peut être utilisé pour authentifier tout message. Cela inclut des défis aléatoires, ce qui signifie que HMAC peut être adapté pour empêcher des répétitions de vieilles sessions. Un désavantage malencontreux de l'utilisation de HMAC pour l'authentification de la connexion est que le secret doit être connu en clair par les deux parties, ce qui rend cela indésirable lorsque la durée de vie des clés est longue.

Lorsque il convient, HMAC devrait être utilisé de préférence à des techniques plus anciennes, en particulier les fonctions de hachage de clés. Les hachages de clé simples fondés sur MD5 [RFC1321], tels que ceux utilisés dans le mécanisme de sécurité de session BGP [RFC2385], sont spécialement à éviter dans les nouveaux protocoles, étant données les indications de faiblesse de MD5.

HMAC peut être mis en œuvre en utilisant toute fonction de hachage sûre, y compris MD5 et SHA-1 [RFC3174]. SHA-1 est préférable pour les nouveaux protocoles parce qu'il est plus fréquemment utilisé à cette fin et peut être plus sûr.

Il est important de comprendre qu'un mécanisme fondé sur HMAC doit être employé sur toutes les unités de données du protocole (à savoir, les paquets). C'est une erreur d'utiliser un système fondé sur HMAC pour authentifier le début d'une session TCP puis d'envoyer tout le reste des données sans aucune protection.

Il existe des programmes d'attaque qui permettent de voler une session TCP. Un attaquant a simplement besoin d'utiliser un tel outil pour voler une session après que l'étape HMAC a été effectuée.

3.3 IPsec

IPsec [RFC2401], [RFC2402], [RFC2406], [RFC2407], [RFC2411] est le protocole générique de chiffrement et d'authentification de la couche IP. Comme tel, il protège toutes les couches supérieures, y compris TCP et UDP. Sa granularité normale de protection est d'hôte à hôte, d'hôte à passerelle, et de passerelle à passerelle. La spécification permet une protection par utilisateur, mais ceci est relativement rare. À ce titre, IPsec est actuellement inapproprié lorsque la granularité par hôte est trop grossière.

Comme IPsec est installé à la couche IP, il est assez intrusif pour le code de réseautage. Le mettre en œuvre exige généralement soit un nouveau matériel, soit une nouvelle pile de protocole. D'un autre côté, il est très transparent aux applications. Les applications qui fonctionnent sur IPsec peuvent avoir une sécurité améliorée sans changer du tout leurs protocoles. Mais au moins jusqu'à ce que IPsec soit plus largement déployé, la plupart des applications ne devraient pas supposer qu'elles vont fonctionner sur IPsec comme solution de remplacement à la spécification de leurs propres mécanismes de sécurité. La plupart des systèmes d'exploitation modernes ont la disponibilité d'IPsec ; la plupart des routeurs ne l'ont pas, au moins pour le chemin de contrôle. Une application qui utilise TLS va plus vraisemblablement être capable d'assurer des applications spécifiques pour tirer parti de son authentification.

La gestion de clé pour IPsec peut utiliser des certificats ou des secrets partagés. Pour toutes sortes de raisons évidentes, les certificats sont préférés ; cependant, ils peuvent constituer un véritable casse-tête pour le gestionnaire de système.

Il y a un fort potentiel de conflit entre IPsec et un NAT [RFC2993]. Le NAT ne coexiste pas facilement avec tout protocole contenant une adresse IP incorporé ; avec IPsec, chaque paquet de chaque protocole contient de telles adresses, ne serait-ce que dans les en-têtes. Le conflit peut parfois être évité en utilisant le mode tunnel, mais ce n'est pas toujours un choix approprié pour d'autres raisons. Des travaux en cours cherchent à faire passer IPsec plus facilement à travers les NAT [RFC3947].

L'usage le plus courant d'IPsec est dans les réseaux privés virtuels. En supposant que les autres contraintes soient satisfaites, IPsec est le protocole de sécurité choisi pour les situations du genre VPN, y compris le scénario d'accès distant où une seule machine tunnelle en retour dans son réseau de rattachement sur l'Internet en utilisant IPsec.

3.4 TLS

TLS [RFC2246] fournit un canal chiffré et authentifié qui fonctionne par dessus TCP. Alors que TLS était à l'origine conçu pour être utilisé par les navigateurs de la Toile, il ne se restreint en aucune façon à cela. En général, cependant, chaque application qui souhaite utiliser TLS aura besoin d'être convertie individuellement.

Généralement, le côté serveur est toujours authentifié par un certificat. Les clients peuvent posséder aussi des certificats, qui fournissent une authentification mutuelle, bien que ce soit rarement déployé. La triste réalité est que même l'authentification du côté serveur n'est pas, en pratique, si sûre que ce que la cryptographie devrait impliquer, parce que la plupart des mises en œuvre permettent aux utilisateurs d'ignorer les défaillances d'authentification (en cliquant sur OK à un avertissement) ce que la plupart des usagers font de façon routinière [Bell98]. Les concepteurs devraient donc avoir la prudence d'exiger des mots de passe en clair, même sur des connexions protégées par TLS. (Cette exigence peut être assouplie si il est probable que les mises en œuvre seront capables de vérifier l'authenticité et l'autorisation du certificat du serveur.)

Bien qu'une modification d'application soit généralement requise pour utiliser TLS, il existe des outils, aussi bien libres que commerciaux, qui en fournissent des mises en œuvre. Celles-ci sont conçues pour être incorporées dans le code de l'application. Une application qui utilise TLS va plus vraisemblablement être capable d'appliquer des politiques de certificat spécifique de l'application que celle qui utilise IPsec.

3.5 SASL

SASL [RFC2222] est un cadre pour la négociation d'un mécanisme d'authentification et de chiffrement à utiliser sur un flux TCP. À ce titre, ses propriétés de sécurité sont celles du mécanisme négocié. Précisément, sauf si le mécanisme négocié authentifie tous les messages suivants ou le protocole de protection sous-jacent comme si TLS était utilisé, les connexions TCP sont vulnérables au vol de session.

Si on a besoin d'utiliser TLS (ou IPsec) sous SASL, pourquoi se soucier d'abord de SASL ? Pourquoi ne pas simplement utiliser les facilités d'authentification de TLS et s'en contenter ?

La réponse est assez subtile. TLS fait un usage extensif de certificats pour l'authentification. Comme ils sont couramment déployés, seuls les serveurs ont des certificats, tandis que les clients restent non authentifiés (au moins par le traitement TLS lui-même).

SASL permet l'utilisation de technologies d'authentification de client plus traditionnelles, telles que les mots de passe (à utilisation unique ou autres). Une combinaison puissante est celle de TLS pour la protection sous-jacente et l'authentification du serveur, et d'un système fondé sur SASL pour l'authentification des clients. Il faut veiller à éviter la vulnérabilité aux attaques par interposition lorsque des techniques d'authentification différentes sont utilisées dans les différentes directions.

3.6 GSS-API

GSS-API [RFC2744] fournit un cadre à utiliser par les applications lorsque elles requièrent l'authentification, la protection de l'intégrité, et/ou de la confidentialité. À la différence de SASL, GSS-API peut être facilement utilisé avec des applications fondées sur UDP. Cela fournit la création de jetons d'authentification opaques (autrement dit de tronçons de mémoire) qui peuvent être incorporés dans des unités de données d'un protocole. Noter que la sécurité des protocoles protégés par GSS-API dépend des mécanismes de sécurité sous-jacents ; ceux-ci doivent être évalués indépendamment. Des considérations similaires s'appliquent bien sûr, à l'interopérabilité.

3.7 DNSSEC

DNSSEC [RFC2535] signe numériquement les enregistrements du DNS. C'est un outil essentiel pour la protection contre les attaques par contamination des antémémoires du DNS [Bell95] ; celles-ci peuvent ensuite être utilisées pour déjouer l'authentification fondée sur le nom et pour rediriger le trafic vers, ou d'un, attaquant. Cela fait de DNSSEC un composant essentiel de certains autres mécanismes de sécurité, notamment IPsec.

Bien que non largement déployé dans l'Internet au moment de la rédaction du présent document, il offre la fourniture potentielle d'un mécanisme sûr pour transposer des noms de domaines en adresses du protocole IP. Il peut aussi être utilisé pour associer en toute sécurité d'autres informations à un nom du DNS.

Ces informations peuvent être aussi simples qu'un service qui est pris en charge sur un certain nœud, ou une clé à utiliser avec IPsec pour négocier une session sûre. Noter que le concept de mémorisation de clés d'application d'utilisation générale dans le DNS a été déconseillée dans la [RFC3445], mais la normalisation de la mémorisation de clés pour des applications particulières – en particulier IPsec – suit son cours.

3.8 Sécurité/Multiparties

Sécurité/Multiparties [RFC1847] est le mécanisme préféré pour la protection de la messagerie électronique. Plus précisément, c'est le cadre MIME au sein duquel le chiffrement et/ou les signatures numériques sont incorporés. S/MIME et OpenPGP (voir ci-dessous) utilisent tous deux Sécurité/Multiparties pour leur codage. Les lecteurs de messagerie conformes peuvent aisément reconnaître et traiter les portions cryptées de la messagerie.

Sécurité/Multiparties représente une forme de la "sécurité de l'objet", où l'objet intéressant pour l'utilisateur final est protégé, indépendamment du mécanisme de transport, du stockage intermédiaire, etc. Actuellement, il n'y a pas de forme générale de protection disponible sur l'Internet.

Pour un bon exemple d'utilisation de S/MIME en dehors du contexte de la messagerie électronique, voir le protocole d'initialisation de session [RFC3261].

3.9 Signatures numériques

Une des plus fortes formes d'authentification par mise au défi/réponse se fonde sur les signatures numériques. Utiliser la

cryptographie à clé publique est préférable aux schémas fondés sur des chiffrements à clé secrète parce que aucun serveur n'a besoin d'une copie du secret du client. Dans ce cas, le client a une clé privée, et les serveurs ont la clé publique correspondante.

Utiliser correctement les signatures numériques est délicat. Un client ne devrait jamais signer la mise au défi exacte qui lui est envoyée, car il y a plusieurs attaques subtiles fondées sur la théorie des nombres qui peuvent être lancées dans de telles situations.

La norme de signature numérique (DSS, *Digital Signature Standard*) [DSS] et RSA [RSA] sont tous deux de bons choix ; chacun a ses avantages. Signer avec DSA requiert l'utilisation de bons nombres aléatoires [RFC1750]. Si l'ennemi peut récupérer le nombre aléatoire utilisé pour une certaine signature, ou si on utilise le même nombre aléatoire pour deux documents différents, la clé privée peut être découverte. DSS a de bien meilleures performances que RSA pour générer de nouvelles clés privées, et des performances un peu meilleures pour générer des signatures, alors que RSA a de bien meilleures performances pour vérifier les signatures.

3.10 OpenPGP et S/MIME

Les signatures numériques peuvent être utilisées pour construire des applications de "sécurité d'objet" qui peuvent être utilisées pour protéger les données dans des protocoles de remise différée comme la messagerie électronique.

Au moment de cette rédaction, deux protocoles différents de messagerie sécurisée, OpenPGP [OpenPGP] et S/MIME [S/MIME], ont été proposés pour remplacer PEM [PEM]. On ne sait pas clairement lequel gagnera, s'il en est un. Bien que spécifiés pour être utilisés avec la messagerie sécurisée, tous deux peuvent être adaptés pour protéger les données portées par d'autres protocoles. Tous deux utilisent des certificats pour identifier les utilisateurs ; tous deux peuvent fournir la confidentialité et l'authentification des messages électroniques ; cependant, les formats des certificats sont très différents. Historiquement, la différence entre les messages fondés sur PGP et ceux fondés sur S/MIME a été le style de la chaîne de certificats. Dans S/MIME, les usagers possèdent des certificats X.509 ; le graphe de certification est une arborescence avec un très petit nombre de racines. À l'opposé, PGP utilise ce qu'on appelle la "toile de confiance", où tout utilisateur peut signer le certificat de quelqu'un d'autre. Ce graphe de certification est en réalité un graphe arbitraire ou un ensemble de graphes.

Avec tout schéma de certificats, la confiance dépend de deux caractéristiques principales. D'abord, elle doit commencer par une source dont la fiabilité est connue, soit une racine X.509, soit quelqu'un qui a toute la confiance du vérificateur, souvent lui-même. Ensuite, la chaîne de signatures doit être fiable. C'est-à-dire que chaque nœud dans le graphe de certification est crucial ; si il est malhonnête ou a été compromis, aucun certificat qu'il a approuvé ne peut être de confiance. Tous les autres facteurs étant égaux par ailleurs (et ils le sont rarement) les plus courtes chaînes sont préférables.

Certaines des différences reflètent une tension entre deux positions philosophiques représentées par ces technologies. D'autres résultent de ce que les équipes de conception étaient distinctes.

S/MIME est conçu comme étant "à l'épreuve des fous". C'est-à-dire que très peu de configuration est requise de l'utilisateur final. Précisément, les utilisateurs finaux n'ont pas besoin d'être au courant des relations de confiance, etc. L'idée est que si un client S/MIME dit , "cette signature est valide", l'usager devrait être capable de "faire confiance" à cette déclaration pour sa valeur faciale sans avoir besoin d'en comprendre les implications sous-jacentes.

Pour réaliser cela, S/MIME est normalement fondé sur un nombre limité d'autorités de certification (CA, *Certifying Authority*) "racines". Le but est de construire une infrastructure mondiale de certificats de confiance.

Le mauvais côté de cette approche est qu'elle exige qu'une infrastructure de clés publiques soit déployée avant que cela fonctionne. Deux utilisateurs finaux peuvent n'être pas capables d'obtenir simplement un logiciel à capacité S/MIME et de commencer la communication en toute sécurité. Ceci n'est pas une limitation du protocole, mais une restriction typique de la configuration pour les logiciels couramment disponibles. Un d'eux, ou les deux peuvent avoir besoin d'obtenir un certificat d'un CA mutuellement de confiance ; de plus, ce CA doit déjà être de confiance pour leurs logiciels de traitement de messagerie. Ce procès peut impliquer des coûts et des obligations légales. Il en résulte finalement que la technologie est plus difficile à déployer, en particulier dans un environnement où l'utilisateur final n'apprécie pas nécessairement la valeur reçue pour les tracas subis.

L'approche PGP de "toile de confiance" présente l'avantage que deux utilisateurs finaux peuvent juste obtenir le logiciel PGP et commencer immédiatement à communiquer en toute sécurité. Aucune infrastructure n'est requise et aucune redevance ni accord légal n'a besoin d'être signé pour commencer. À ce titre, PGP plaît aux personnes qui ont besoin d'établir des associations de sécurité ad hoc.

Le mauvais côté de PGP est qu'il exige de l'utilisateur final qu'il ait la compréhension de la technologie de sécurité sous-jacente afin d'en faire une utilisation efficace. Précisément, il est très facile de tromper les utilisateurs naïfs et de leur faire accepter un message "signé" qui est en fait un faux.

Aujourd'hui, PGP a recueilli un haut niveau d'acceptation parmi les individus conscients des problèmes de sécurité qui ont besoin de messagerie électronique sûre dans un environnement dépourvu de l'infrastructure globale nécessaire.

À l'opposé, S/MIME fonctionne bien dans un environnement d'entreprise où un système de CA interne sûr peut être déployé. Il n'exige pas beaucoup de connaissance de la sécurité de la part de l'utilisateur final. S/MIME peut être utilisé entre des institutions en établissant une certification croisée méticuleuse, mais c'est plus dur à faire qu'il n'y paraît.

Au moment de cette rédaction, une infrastructure de certificat mondiale continue de nous échapper. Des questions sur le modèle d'exploitation convenable, ainsi que les considérations de confidentialité, pourraient nous empêcher de le voir jamais émerger.

3.11 Les pare-feu et la topologie

Les pare-feu sont un mécanisme de défense topologique. C'est-à-dire qu'ils s'appuient sur une frontière bien définie entre le bon "intérieur" et le mauvais "extérieur" d'un certain domaine, le pare-feu s'interposant dans le passage des informations. Bien que les pare-feu puissent être très précieux s'ils sont employés à bon escient, il y a des limites à leur capacité à protéger un réseau.

La première limitation, bien sûr, est que les pare-feu ne peuvent pas protéger contre les attaques de l'intérieur. Bien que le taux d'incidence réel de telles attaques ne soit pas connu (et soit probablement inconnaissable) il ne fait pas de doute qu'il est substantiel, et constitue indiscutablement la majorité des problèmes de sécurité. Plus généralement, étant donné que les pare-feu exigent une frontière bien délimitée, dans la mesure où une telle frontière n'existe pas, les pare-feu ne sont d'aucune aide. Toutes les connexions externes, que ce soient des protocoles qui sont délibérément passés à travers le pare-feu, des liaisons qui sont tunnelées au travers, des LAN sans fils non protégés, ou des connexions directes externes avec des hôtes nominalement à l'intérieur, affaiblissent la protection. Les pare-feu tendent à devenir moins efficaces au fil du temps, à mesure que les utilisateurs tunnellent les protocoles à travers eux et peuvent avoir une sécurité inadéquate sur les points d'extrémité du tunnel. Si les tunnels sont chiffrés, il n'y a aucun moyen pour le pare-feu de les censurer. Un avantage souvent cité du pare-feu est qu'il cache l'existence des hôtes internes aux yeux de l'extérieur. Cependant, considérant la quantité de fuites, la probabilité de réussite de la dissimulation des machines est assez faible.

Dans une veine plus subtile, les pare-feu portent atteinte au modèle de bout en bout de l'Internet et de ses protocoles. Bien sûr, tous les protocoles ne passent pas sûrement et facilement à travers les pare-feu. Les sites qui s'appuient sur les pare-feu pour leur sécurité peuvent se trouver un peu coupés des aspects nouveaux et utiles de l'Internet.

Les pare-feu fonctionnent le mieux lorsque ils sont utilisés comme éléments d'une structure de sécurité totale. Par exemple, un pare-feu strict peut être utilisé pour séparer un serveur de la Toile exposé d'une base de données en arrière plan, avec pour seule ouverture le canal de communication entre les deux. De façon similaire, un pare-feu qui ne permet que le trafic d'un tunnel chiffré pourrait être utilisé pour sécuriser une morceau d'un VPN. D'un autre côté, dans ce cas, l'autre extrémité du VPN aurait également besoin d'être sécurisée.

3.12 Kerberos

Kerberos [RFC1510] fournit un mécanisme pour que deux entités s'authentifient l'une l'autre et échangent du matériel de clés. Du côté client, une application obtient un "ticket" et un "authentificateur" Kerberos. Ces éléments, qui devraient être considérés comme des données opaques, sont alors communiqués du client au serveur. Le serveur peut alors vérifier leur authenticité. Les deux côtés peuvent alors demander au logiciel Kerberos de leur fournir une clé de session qui peut être utilisée pour protéger ou chiffrer les données.

Kerberos peut être utilisé par lui-même dans un protocole. Cependant, il est aussi disponible comme mécanisme sous SASL et GSSAPI. Il a des faiblesses connues [KRBATTACK], [KRBLIM], [KRB4WEAK], mais peut être utilisé en toute sécurité.

3.13 SSH

SSH fournit une connexion sûre entre client et serveur. Il fonctionne tout à fait comme TLS ; cependant, il est optimisé comme protocole pour les connexions distantes sur des appareils du style terminal. Une de ses caractéristiques les plus innovantes est sa prise en charge du "tunnelage" d'autres protocoles sur la connexion TCP protégée par SSH. Cette caractéristique a permis à des personnes qui ont de bonnes connaissances en matière de sécurité d'effectuer des actions

comme de lire et d'envoyer de la messagerie ou des nouvelles via des serveurs non sûrs sur un réseau non sûr. Ce n'est pas un substitut d'un vrai VPN, mais cela peut souvent être utilisé à la place d'un vrai.

4. Mécanismes d'insécurité

Certains mécanismes de sécurité courants font plus partie du problème que de la solution.

4.1 Mots de passe en clair

Les mots de passe en clair sont le mécanisme de sécurité le plus courant aujourd'hui. Malheureusement, ils sont aussi le plus faible. Lorsque ils ne sont pas protégés par une couche de chiffrement, ils sont complètement inacceptables. Même quand ils sont utilisés avec chiffrement, les mots de passe en clair sont assez faibles, car ils doivent être transmis au système distant. Si ce système a été compromis ou si la couche de chiffrement ne comporte pas d'authentification efficace du serveur auprès du client, un ennemi peut collecter les mots de passe et éventuellement les utiliser contre d'autres cibles.

Une autre faiblesse apparaît à cause des techniques courantes de mise en œuvre. Il est considéré comme une bonne formule [MT79] qu'un hôte mémorise un hachage unidirectionnel des mots de passe des usagers, plutôt que de leur forme en clair. Cependant, cela peut empêcher de migrer vers de plus forts mécanismes d'authentification, comme la mise au défi/réponse fondée sur HMAC.

La plus forte attaque contre les mots de passe, autre que l'espionnage, est de deviner le mot de passe. Avec un programme convenable et un dictionnaire (et ceux-ci sont largement disponibles) 20 à 30 % des mots de passe peuvent être devinés dans la plupart des environnements [Klein90].

4.2 Authentification fondée sur l'adresse

Un autre mécanisme de sécurité courant est l'authentification fondée sur l'adresse. Au mieux, elle peut fonctionner dans des environnements très restreints. Si votre environnement consiste en un petit nombre de machines, toutes bien surveillées, avec des systèmes sûrs manœuvrés par des utilisateurs de confiance, et si le réseau est gardé par un routeur qui bloque l'acheminement de source et empêche l'usurpation des vos adresses de source, et si vous savez qu'il n'y a pas de pont sans fil, et si vous restreignez l'authentification fondés sur l'adresse aux machines de ce réseau, vous serez probablement en sûreté. Mais ces conditions sont rarement satisfaites.

Parmi les menaces figurent l'usurpation d'ARP, l'abus de mandataire local, la dénumérotation, la corruption ou les attaques de tableau d'acheminement, l'usurpation de DHCP, d'adresse IP (risque particulier pour les protocole fondés sur UDP), deviner le numéro de séquence, et les paquets en acheminement de source. Tout cela peut être assez puissant.

4.3 Authentification fondée sur le nom

L'authentification fondée sur le nom pose tous les problèmes de l'authentification fondée sur l'adresse et en ajoute de nouveaux : les attaques contre le DNS [Bell95] et le manque d'une transposition bijective entre adresses et noms. Au minimum, un processus qui restitue un nom d'hôte à partir du DNS devrait restituer les enregistrements d'adresses correspondants et les vérifications croisées. Des techniques telles que la contamination d'antémémoire du DNS peuvent souvent déjouer de telles vérifications.

DNSSEC fournit la protection contre cette sorte d'attaque. Cependant, il ne fait rien pour améliorer la fiabilité de l'adresse sous-jacente. De plus, la technique génère beaucoup de fausses alarmes. Ces recherches ne donnent pas d'informations fiables à une machine, bien qu'elles puisse être un outil utile de débogage pour les humains et pourrait être utile dans les journaux d'événements lorsque on essaye de reconstruire comment une attaque s'est déroulée.

5. Considérations pour la sécurité

Aucun mécanisme de sécurité n'est parfait. Si il n'y a rien d'autre, tout mécanisme de sécurité fondé sur le réseau peut être déjoué par la compromission des points d'extrémité. Cela dit, chacun des mécanismes décrits ici a ses propres limitations. Toutes décision d'adopter un certain mécanisme devrait tenir compte de tous les modes de défaillance possibles. Ceux-ci à leur tour devraient être évalués par rapport aux risques pour le point d'extrémité d'une défaillance de la sécurité.

6. Considérations relatives à l'IANA

Il n'y a aucune considérations relatives à l'IANA concernant ce document.

7. Remerciements

Brian Carpenter, Tony Hain, et Marcus Leech ont fait des suggestions utiles. Une grande partie de la substance de ce document vient des participants à l'atelier Architecture de sécurité de l'IAB.

8. Références pour information

[Bell95] "Using the Domain Name System for System Break-Ins". Proc. Fifth Usenix Security Conference, 1995.

[Bell98] S.M. Bellovin, "Cryptography and the Internet", dans Proceedings of CRYPTO '98, août 1998.

[DSS] NIST. "Digital Signature Standard". mai 1994. FIPS 186.

[Klein90] D. Klein. "Foiling the Cracker: A Survey of, and Implications to, Password Security". Usenix UNIX Security Workshop, août 1990.

[KRBATTACK] T. Wu. "A Real-World Analysis of Kerberos Password Security". Network and Distributed System Security Symposium (NDSS '99). janvier 1999.

[KRBLIM] "Limitations of the Kerberos Authentication System". Proceedings of the 1991 Winter USENIX Conference, 1991.

[KRB4WEAK] "Misplaced trust: Kerberos 4 session keys". Proceedings of the Internet Society Network and Distributed Systems Security Symposium, mars 1997.

[MT79] R.H. Morris and K. Thompson, "UNIX Password Security", Communications of the ACM. novembre 1979.

[RFC3947] T. Kivinen et autres, "Négociation de traversée de NAT dans IKE", janvier 2005. *(P.S.)*

[RFC1321] R. Rivest, "Algorithme de résumé de message MD5", avril 1992. *(Information)*

[RFC1510] J. Kohl et C. Neuman, "Service Kerberos d'authentification de réseau (v5)", septembre 1993. *(Obsolète, voir* RFC6649*)*

[RFC1750] D. Eastlake 3rd, et autres, "Recommandations d'aléa pour la sécurité", décembre 1994. *(Info., remplacée par la* RFC4086)

[RFC1847] J. Galvin, S. Murphy, S. Crocker, N. Freed, "Sécurité multiparties pour MIME : multipartie/signée et multipartie/chiffrée", octobre 1995. *(P.S.)*

[RFC2104] H. Krawczyk, M. Bellare et R. Canetti, "HMAC : Hachage de clés pour l'authentification de message", février 1997.

[RFC2222] J. Myers, "Authentification simple et couche de sécurité (SASL)", octobre 1997. (*Obsolète, voir* RFC4422, RFC4752) (*MàJ par* RFC2444) *(P.S.)*

[RFC2246] T. Dierks et C. Allen, "Protocole TLS version 1.0", janvier 1999.

[RFC2289] N. Haller, C. Metz, P. Nesser, M. Straw, "Système de mot de passe à utilisation unique", février 1998. (STD0061)

[RFC2316] S. Bellovin, "Rapport de l'atelier IAB Architecture de sécurité", avril 1998. *(Information)*

[RFC2385] A. Heffernan, "Protection des sessions de BGP via l'option de signature MD5 de TCP", août 1998. *(P.S.* (*MàJ par la* RFC6691) *(Remplacée par* RFC5925)

[RFC2401] S. Kent et R. Atkinson, "Architecture de sécurité pour le protocole Internet", novembre 1998. (*Obsolète, voir* RFC4301)

[RFC2402] S. Kent et R. Atkinson, "En-tête d'authentification IP", novembre 1998. (*Obsolète, voir* RFC4302, 4305)

[RFC2406] S. Kent et R. Atkinson, "Encapsulation de charge utile de sécurité IP (ESP)", novembre 1998. *(Obsolète, voir* RFC4303*)*

[RFC2407] D. Piper, "Le domaine d'interprétation de sécurité IP de l'Internet pour ISAKMP", novembre 1998. *(Obsolète, voir* 4306*)*

[RFC2411] R. Thayer, N. Doraswamy, R. Glenn, "Feuille de route pour la sécurité sur IP", novembre 1998. *(Remplacée par* RFC6071*)*

[RFC2535] D. Eastlake, 3rd, "Extensions de sécurité du système des noms de domaines", mars 1999. (*Obsolète, voir* RFC4033, RFC4034, RFC4035) *(P.S.)*

[RFC2744] J. Wray, "API de service générique de sécurité, version 2 : liaisons C", janvier 2000. *(P.S.)*

[RFC2993] T. Hain, "Implications architecturales des traducteurs d'adresse réseau (NAT)", novembre 2000. *(Information)*

[RFC3174] D. Eastlake 3 et P. Jones, "Algorithme US de hachage sécurisé n° 1 (SHA1)", sept. 2001. *(Info, MàJ par 4634 et 6234)*

[RFC3261] J. Rosenberg et autres, "SIP : Protocole d'initialisation de session", juin 2002. (*Mise à jour par* RFC3265, RFC3853, RFC4320, RFC4916, RFC5393, RFC6665))

[RFC3445] D. Massey, S. Rose, "Limitation de la portée de l'enregistrement de ressource (RR) KEY", décembre 2002. (*Obsolète, voir* RFC4033, RFC4034, RFC4035) (*MàJ* RFC2535) *(P.S.)*

[RSA] Rivest, R., Shamir, A. and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", Communications of the ACM, février 1978.

9. Déclaration de droits de propriété intellectuelle

10. Informations sur les auteurs

Ce document est une publication du Bureau d'architecture de l'Internet. Les membres du Bureau d'architecture de l'Internet au moment de l'achèvement du présent document étaient :
Bernard Aboba, Harald Alvestrand, Rob Austein, Leslie Daigle (président), Patrik Faltstrom, Sally Floyd, Jun-ichiro Itojun Hagino, Mark Handley, Geoff Huston, Charlie Kaufman, James Kempf, Eric Rescorla, Michael StJohns.

Internet Architecture Board
mél : iab@iab.org

Steven M. Bellovin, éditeur
mél : bellovin@acm.org

Jeffrey I. Schiller, éditeur
mél jis@mit.edu

Charlie Kaufman, éditeur
mél : charliek@microsoft.com

11. Déclaration complète de droits de reproduction

Remerciement
Le financement de la fonction d'édition des RFC est actuellement fourni par la Internet Society.