Sécurité des réseaux et du Web
8INF135

Travail Pratique #2

présenté à
M. Valère Plantevin

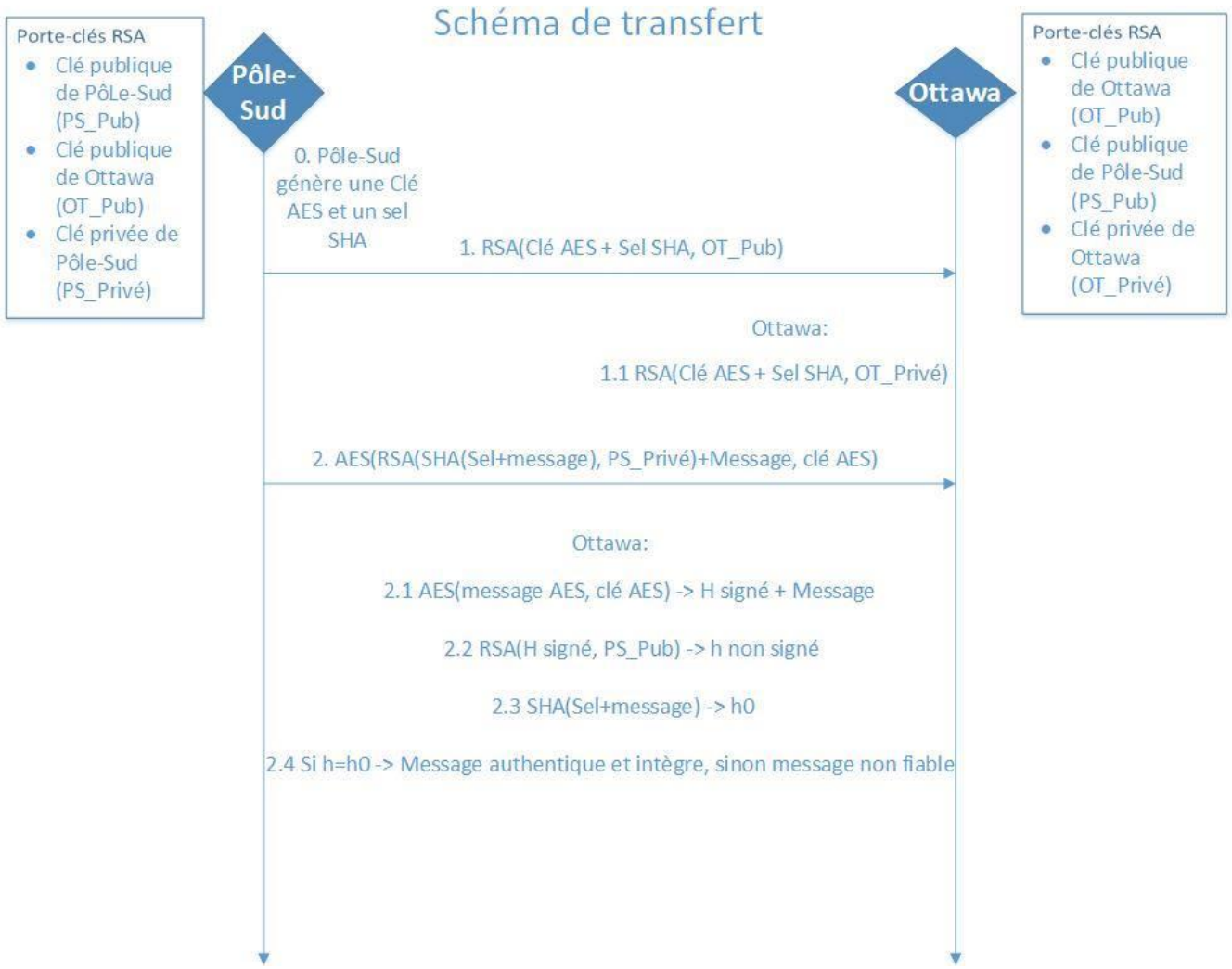
Par :
Julien Haegman ()
Mathieu Blackburn (BLAM 0304 9208)
Sébastien Tremblay (TRES 1408 8101)

Le dimanche 12 novembre 2017

Fonctionnement du relais

Le relais qui a été développé (nommé « SecuRelay »), prévoit une communication bidirectionnelle, c'est-à-dire que le même relais peut être utilisé autant à Ottawa qu'au Pôle-Sud, et ce autant pour envoyer que pour recevoir des messages chiffrés. Le serveur prévoit plusieurs paramètres de lignes de commande pour en personnaliser l'exécution de part et d'autre du canal de communication. Également, un fichier porte-clés RSA différent doit être utilisé sur chaque serveur.

Vue d'ensemble du processus



Description sommaire du processus

Sur le premier site (ex : Pole-Sud) :

- Le serveur du relais charge son porte-clés RSA personnalisé contenant 3 clés :
  - Sa clé privée de 4096 bits
  - Sa clé publique de 4096 bits
  - La clé publique de 4096 bits du serveur distant (ex : d'Ottawa)
- Un message en claire (non chiffré) est envoyé au relais par un poste du réseau interne.
- Une connexion vers le serveur distant (ex : Ottawa) est ouverte.
- Le relais génère une clé AES de 256 bits ainsi qu'un sel cryptographique de 64 octets (512 bits). Tous deux sont générés de façon aléatoire en utilisant l'algorithme cryptographique SHA1PRNG.
- Ces deux informations (clé AES et sel) sont chiffrées avec l'algorithme asymétrique RSA en utilisant la clé publique du serveur distant, de sorte que seul le serveur distant sera en mesure d'en déchiffrer le message en utilisant sa clé privée.
- Ce paquet de 512 octets (4096 bits) est transmis au serveur distant, en gardant la connexion ouverte puisqu'un autre paquet suivra.
- Le relais génère un hash de la concaténation du sel cryptographique généré plus tôt et du message clair en utilisant l'algorithme SHA-2 512 bits (SHA-512).
- Le hash est signé avec l'algorithme RSA en utilisant la clé privée du serveur local.
- Le hash signé est concaténé avec le message clair et le tout est chiffré en AES avec la clé AES générée au début. Le mode d'opération utilisé est CTR avec un compteur initialisé à zéro.
- Ce paquet chiffré est à son tour transmis au serveur distant.
- La connexion au serveur distant est fermée puisque toutes les données nécessaires ont été transmises.

Sur le second site (ex : Ottawa) :

- Le serveur du relais charge son porte-clés RSA personnalisé contenant, lui aussi, 3 clés :
  - Sa clé privée de 4096 bits
  - Sa clé publique de 4096 bits (la même que la clé publique « distante » de l'autre serveur)
  - La clé publique de 4096 bits du serveur distant (la même que la clé publique « personnelle » de l'autre serveur)
- Un message chiffré est reçu par le relais en provenance d'Internet.
- Le premier bloc de 512 octets, que l'on sait être un message chiffré avec RSA, est déchiffré en utilisant la clé privée, le relais récupère ainsi 2 informations : la clé de chiffrement AES 256 bits ainsi que le sel cryptographique à utiliser pour refaire le hash du message (voir plus loin).
- Le reste du message est chiffré en AES, le relais utilise donc la clé de chiffrement AES pour décoder le reste du message. Il récupère ainsi le hash signé du message (premiers 512 octets) ainsi que le message lui-même (le reste).
- Le hash signé est déchiffré par l'algorithme RSA en utilisant la clé publique du relais distant pour récupérer le hash original « non-signé » (nommons le « *h* »).
- Le relais génère sa propre version du hash de la concaténation du sel cryptographique et du message qu'il vient de déchiffrer (nommons les « *h0* »).
- Les deux hash « *h* » et « *h0* » sont comparés. S'ils sont identiques, on en conclut que le message est authentique et inaltéré. Si ce n'est pas le cas, on affiche une erreur dans la console et on termine le processus concernant ce message.
- On sait maintenant que le message est valide, on le diffuse donc sur le groupe multicast.

Description détaillée du fonctionnement du serveur

Au démarrage, SecuRelay démarre un thread persistant qui ouvre un port d'écoute sur l'interface du côté du réseau local pour recevoir des messages « en clair » et qui accepte toute demande de connexion à ce port en transmettant le « socket » accepté à un autre thread qui traitera la connexion. Ce thread a comme travail d'accumuler toutes les données reçues sur le socket jusqu'à ce que la connexion soit fermée, puis de s'occuper du chiffrement du message. Nous le nommerons ci-après le thread de chiffrement. Celui-ci, après avoir reçu le message clair, commence par démarrer un thread d'envoi de message chiffré, puis procède aux diverses opérations de chiffrement. En découpant ces actions en deux threads distincts, cela permet d'éviter des délais inutiles avec des opérations bloquantes. En effet, pendant que le thread de chiffrement est occupé à effectuer une opération de chiffrement, le thread d'envoi de message s'occupe d'établir la connexion avec le serveur distant. Une fois que le thread de chiffrement a chiffré le premier message RSA contenant la clé AES et le sel cryptographique, le thread d'envoi devrait avoir eu le temps d'initialiser le « préambule » de connexion. Que ce soit le cas ou non, le thread de chiffrement ajoute simplement le message RSA à la file d'attente des messages à envoyer, et le thread d'envoi l'enverra lorsqu'il sera prêt. Pendant qu'il envoie ce message, le thread de chiffrement continue son travail; il génère le hash du message, le signe en le chiffrant en RSA avec sa clé privée, concatène le résultat au message clair, et hash le tout en AES avec la clé générée plus tôt. Le thread de chiffrement ajoute ce nouveau message chiffré à la file d'attente du thread d'envoi puis lui indique qu'il ne recevra plus de nouveaux messages. Le thread de chiffrement se termine, pendant que le thread d'envoi termine de vider sa file d'attente, puis ferme la connexion et se termine lui aussi.

Au démarrage, SecuRelay démarre également un second thread persistant qui ouvre un port sur l'interface du côté « internet » pour recevoir des messages chiffrés en provenance de l'autre relais. Ce thread accepte toute demande de connexion à ce port et transmet le socket accepté à un autre thread qui traitera la connexion. Tout comme pour le récepteur de messages clairs, ce thread accumule toutes les données reçues sur le socket jusqu'à ce que la connexion soit fermée, puis effectue les actions suivantes. D'abord, il valide et déchiffre toutes les informations reçues; déchiffrement des 512 premiers octets avec RSA en utilisant la clé privée pour récupérer la clé AES et le sel cryptographique, déchiffrement du reste du message avec AES en utilisant la clé AES reçue et déchiffrement du hash signé en utilisant RSA et la clé publique de l'autre relais. Puis, il génère lui-même le hash du message qu'il a reçu. Finalement, il le compare au hash déchiffré qu'il a reçu. S'ils sont identiques, il considère le message authentique et le diffuse sur le groupe multicast.

De plus, pour faciliter la validation de l'acheminement du message (on s'adresse ici au professeur et non à l'état-major), SecuRelay démarre un 3^{e} thread persistant qui écoute toutes les diffusions effectuées sur le groupe multicast et les affiche dans la console.

SecuRelay utilise un pool de 50 threads, en plus des 3 threads persistants, pour gérer les différentes demandes de chiffrement et de déchiffrement.

Étude de cas

Checklist

Besoins

- Algorithmes :
  - SHA-512
  - RSA-4096
  - AES-256
  - Génération aléatoire forte
- Communication :
  - Certitude que le paquet est bien arrivé
  - Certitude que personne n'a pu lire le paquet
  - Certitude que personne n'a pu modifier le paquet
  - Certitude de qui a envoyé le paquet

Assets

- Matériel
  - 5 serveurs
    - 1 réservé à la communication avec Ottawa
    - 1 pour nous et les membres du personnel ayant accès aux secrets militaires
    - 1 pour les mécaniciens
    - 1 pour les données des scientifiques
    - 1 serveur de calcul pour les scientifiques
  - 51 postes de travail Windows
    - Le nôtre
    - 15 pour le personnel
    - 15 pour les scientifiques
    - 20 pour le reste de la base
- Logiciel
  - Linux
  - Windows
  - Application de calcul scientifique
- Données
  - Secrets militaires
  - Schémas techniques de la base
  - Données des scientifiques
- Moyens de communication
  - 2 pares-feux basés sur « iptables »
  - 1 routeur
  - 2 switch gérant les VLAN

Risques

- Les communications avec Ottawa peuvent être interceptées (risque à la confidentialité)
- Les communications avec Ottawa peuvent être altérées (risque à l'intégrité)
- Une tierce partie peut tenter d'envoyer un faux message à Ottawa, se faisant passer pour nous (risque à l'authenticité)
- Une tierce partie peut tenter de nous envoyer un faux message, se faisant passer pour Ottawa (risque à l'authenticité)
- La base peut être victime d'une attaque de déni de service (risque à la disponibilité)

Contre-mesures

- On ne peut éviter que les communications avec Ottawa soient interceptées, c'est pourquoi nous devons chiffrer toutes les communications entre la base et Ottawa afin que les données qui pourraient être potentiellement interceptée soient dénuées de sens pour tout utilisateur qui n'est pas le destinataire légitime : RSA-4096 et AES-256
- On ne peut éviter que les paquets envoyés depuis ou vers Ottawa ne soient altérés. Il faut donc inclure dans les échanges un hash du message prouvant au destinataire, en refaisant lui-même le hash, que le message n'a pas été altéré si les hashs sont identiques : SHA-512 avec sel cryptographique généré aléatoirement.
- Pour éviter qu'une tierce partie ne réussisse à se faire passer pour l'une ou l'autre des parties (base du Pole-Sud ou Ottawa), il faut chiffrer le hash avec la clé privée de l'émetteur, prouvant ainsi au récepteur que le message est authentique s'il peut le déchiffrer avec la clé publique correspondante que le hash correspond bien à celui auquel il s'attend : RSA-4096 et SHA-512
- La meilleure façon de se prévenir contre les attaques de déni de service est de bloquer au maximum le trafic entrant et sortant au niveau du pare-feu, ne laissant passer que le trafic légitime : Pare-feu basé sur iptables

Équilibrages

- Aucun compromis possible sur la sécurité (sécurité de grade militaire)

Décisions concernant le relais

Puisqu'aucune information n'a été fournie concernant le format ou le protocole des communications pour les messages clairs entrant dans le relais, la seule façon de connaître la fin du message entrant était de lire toutes les données entrantes jusqu'à ce que la connexion soit fermée. Ainsi, il n'était pas possible d'assurer à l'utilisateur envoyant le message clair que le paquet est bien arrivé à destination puisque la connexion est fermée avant même que le message ne soit chiffré et envoyé. De plus, comme aucun protocole n'est établi, aucune méthode n'est prévue pour informer l'expéditeur du succès ou de l'échec de l'envoi de son message. Pour pallier au mieux à cette situation, nous utilisons une connexion TCP pour effectuer la communication entre les relais. En cas d'erreur, un message est affiché dans la console du relais.

Pour assurer la confidentialité du message, nous avons opté pour un chiffrement AES du message avec un mode d'opération CTR, débutant avec un compteur initialisé à 0. Puisque nous générons une nouvelle clé AES aléatoirement à chaque nouvel envoi de message, nous pouvons nous permettre d'utiliser CTR puisque nous n'utilisons ainsi jamais deux fois la même combinaison de clé AES et de valeur du compteur. De plus, ce mode d'opération permet d'éviter la génération d'un IV aléatoire et sa transmission sécurisée entre les 2 parties. Cependant, la clé AES doit rester secrète et n'être connue que de l'expéditeur et du récepteur. C'est pourquoi elle est elle-même chiffrée avec RSA en utilisant la clé publique du destinateur, s'assurant que seul celui-ci pourra la déchiffrer avec sa clé privée qu'il est seul à posséder.

Pour assurer l'intégrité du message, nous effectuons un hash du message avec l'algorithme SHA-512 et nous chiffrons ce hash avec AES en même temps que le message lui-même. Pour éviter les attaques par tables arc-en-ciel sur ce hash, nous ajoutons un sel cryptographique de 512 bits (généré aléatoirement) devant le message avant d'effectuer le hash. Pour éviter qu'un attaquant ne prenne possession de ce sel, celui-ci est chiffré avec RSA en même temps que la clé AES, encore une garantie que seul le possesseur de la clé privée du destinateur pourra récupérer ce sel. Le récepteur du message n'aura qu'à générer lui-même le hash du message qu'il a reçu, en utilisant le même sel, pour s'assurer que le message est intègre si les deux hashs correspondent.

Pour assurer l'authenticité du message, il n'est pas suffisant d'envoyer le hash directement parce que si pour une raison ou une autre, une tierce partie tombait en possession du sel cryptographique et de la clé AES (c'est-à-dire, si la confidentialité était compromise), celui-ci pourrait également générer son propre message et son propre hash. Le destinataire n'aurait alors aucune façon de vérifier que le message provient de la bonne personne. C'est pourquoi, avant d'envoyer le hash, nous le chiffrons en RSA en utilisant la clé privée de l'émetteur. De cette façon, le récepteur doit d'abord déchiffrer le hash en utilisant la clé publique de l'émetteur « supposé » (celui auquel il s'attend). Si le déchiffrement réussi et que les hashs correspondent, cela lui indique que le hash qu'il a reçu avait bien été chiffré par la clé privée correspondant à la clé publique utilisée, donc que l'émetteur est bien celui auquel on s'attendait.

Le générateur aléatoire utilisé est SHA1PRNG, qui est le seul algorithme de génération de nombre pseudo-aléatoire de grade cryptographique fourni avec Java.

Décisions concernant le réseau

Architecture des pare-feu

Puisque nous voulons protéger des secrets militaires qui peuvent changer le cours de l'histoire et que le matériel fourni nous le permet, nous avons penché vers l'architecture de pare-feu du double bastion en ligne. Premièrement, cette architecture nous permet d'utiliser un serveur de relais dans la DMZ dans lequel les messages que nous voulons envoyer à Ottawa seront reçus, chiffrés et puis envoyés sur leur serveur. De plus, elle prévoit l'utilisation de deux pare-feu qui sont utilisés pour filtrer les requêtes internes et externes de la base de Pôle-sud. Avec les pares-feux configurés avec iptables, les seules requêtes sortantes de notre réseau qui sont possibles sont les messages cryptés et envoyés vers l'IP d'Ottawa « rpiexplorer.io:8080 » et les seules requêtes entrantes possibles sont leurs messages cryptés à destination de notre port 8080 (IP inconnue, si nous avions connu cet IP, nous aurions pu affiner la dernière ligne de notre fichier de configuration iptables.sh pour n'autoriser que les demande de connexion à destination de cet IP sur le port 8080). De cette façon, nous pouvons nous assurer que les utilisateurs du réseau ne puissent pas communiquer avec d'autre tiers et nous pouvons nous assurer que les seuls messages que nous recevons viennent de l'IP qu'Ottawa nous a fourni. Nous avons aussi ajouté des règles pour se protéger des attaques de DoS en bloquant :

- Tous les paquets qui n'ont pas de SYN
- Tous les paquets avec des TCPMSS suspects
- Tous les paquets avec des flags inhabituels
- Toutes les connexions au-delà de 100 par IP

De plus, avec l'architecture double bastion en ligne, si d'une manière ou d'une autre, des espions réussissent à passer à travers le premier pare-feu, il y aura une deuxième couche de pare-feu entre le relais et le réseau interne abritant les secrets militaires et autres informations critiques.

VLANs

Suite à une analyse sur les types d'utilisateurs des 51 ordinateurs et les 5 serveurs qui feront partie de notre réseau, nous avons déterminé qu'il y aurait un total de 4 types de droits à accorder ainsi que 4 types d'utilisateurs :

Types de droit :

1. Secrets militaires (2 serveurs) : VLAN 2
2. Données scientifiques (2 serveurs) : VLAN 3
3. Schémas techniques de la base (1 serveur) : VLAN 4
4. Aucun droit particulier : VLAN 5

Types d'utilisateurs :

1. Scientifiques, accès uniquement aux données scientifiques (15 postes) : VLAN 3
2. Mécaniciens, accès uniquement aux schémas techniques de la base (10 postes) : VLAN 4
3. Soldats, accès à rien (20 postes) : VLAN 5
4. Nous et personnel de la base, accès à l'ensemble du réseaux (6 postes) : VLAN 2, 3, 4, 5

Les VLANs sont construits de façon à ce que chaque groupe d'utilisateurs puissent communiquer seulement entre ce même groupe ainsi que leur serveur à l'exception du personnel ayant un accès à tout. Puisque nous n'avons aucune idée de la structure des bâtiments, nous avons fait notre schéma en utilisant un seul commutateur à l'intérieur du réseau local, mais lors de l'implémentation nous aurons probablement besoin d'utiliser d'autres commutateurs et routeurs pour faire le lien avec les machines qui pourraient se situer dans d'autres bâtiments de la base.

Puisque nous sommes l'administrateur du réseau et que rien ne nous affirme le contraire, nous avons présumé avoir les privilèges d'accéder à l'ensemble du réseau comme les membres du personnel ayant accès à tout.

Nous avons schématisé le réseau comme suit :