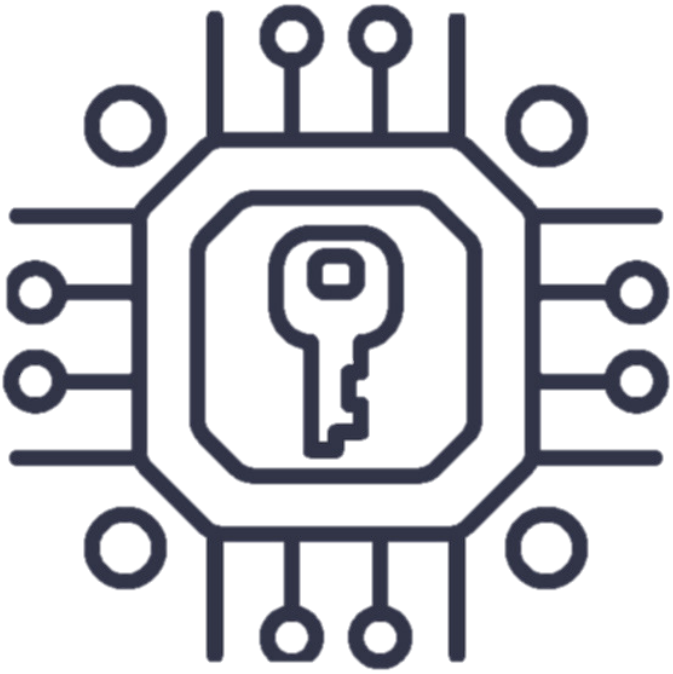
Rapport de SAE

SAE 3.04 : Cryptographie

DEMARET Sullivan - COULON Titouan

5Groupe 22B



Réponses aux questions :

Partie 1 :

1) En supposant que RSA soit utilisé correctement, Eve peut-elle espérer en venir à bout ? En vous appuyant sur votre cours, justifiez votre réponse.

On suppose que RSA est bien utilisé, donc que `p` et `q` soient d'une longueur suffisante et que la clé privée ne soit pas divulguée, Eve ne devrait pas compter sur le crackage brutal de RSA car ça lui bien trop de temps (x millions d'années) car elle devrait décomposer `n` en facteurs premiers.

2) En quoi l'algorithme SDES est-il peu sécurisé ? Vous justifierez votre réponse en analysant le nombre d'essai nécessaire à une méthode "force brute" pour retrouver la clé.

L'algorithme SDES utilise des clés de 10 bits de longueur, donc 2^10 = 1 024 clés disponibles, il faut faudrait donc 1 024/2 = 512 essais en moyenne pour "brute-forcer" le SDES avec un cassage brut simple, ce qui est très peu avec les ordinateurs actuels.

3) Est-ce que double SDES est-il vraiment plus sur ? Quelle(s) information(s) supplémentaire(s) Eve doit-elle récupérer afin de pouvoir espérer venir à bout du double DES plus rapidement qu'avec un algorithme brutal ? Décrivez cette méthode astucieuse et précisez le nombre d'essai pour trouver la clé

L'algorithme double SDES utilise des clés de 10 bits de longueur, donc pour le double SDES et ses 2 clés, 2^(2*10) = 1 048 576 clés disponibles, il faut faudrait donc 1 048 576/2 = 524288 essais en moyenne pour "brute-forcer" le double SDES avec un cassage brut simple, ce qui est mieux mais pas exceptionnel. Pour "brute-forcer" de façon plus astucieuse, il faudrait utiliser la technique du "meet in the middle" ce qui réduirait le nombre de clés à (2^10)*2 = 2048 essais, ce qui est très peu et montre que sa sécurité est faible mais il faut pour cela récupérer un message crypté et sa version en claire, ce qui est très compliqué pour Eve (en théorie). Le but de cette attaque est de tester toutes les combinaisons de clés pour chiffrer et déchiffrer et comparer avec le texte chiffré "original".

Expérience de test des fonctions :

Clé1 et Clé2 aléatoire entre 0 et 2^10, texte clair toujours le même et 10 essais pour calculer la moyenne.

encode(), decode() : Quelques millisecondes d'exécution pour crypter et decrypter arsene ou lettres en SDES ou double SDES.

cassage_brutal () : Je n'ai pas pu laisser la fonction retrouver la clé mais elle prend au moins 2 heures (et surement beaucoup plus encore) pour retrouver arsene crypté en double SDES et moins de 10 secondes pour le simple SDES.

cassage_astucieux() : environ une cinquantaine de secondes pour casser le double SDES sur chacun des textes en 1100 essais environ mais le problème du cassage astucieux est le temps et le stockage pas le nombre d'essais.

Partie 2 :

1) Est-ce vraiment un problème ? Justifiez votre réponse.

Oui, c'est un problème car d'un nombre de clés possibles égale à 2^10 soit 1024 on passe à 2^256 soit 1.16x10^77 clés possibles ce qui rend (quasi) impossible le cassage brutal.

2) Nous allons tenter d'illustrer expérimentalement les différences entre les deux protocoles. Vous évaluerez :

A) Le temps d'exécution du chiffrement/déchiffrement d'un message avec chacun des deux protocoles. Ici vous devez le mesurer expérimentalement et donc fournir le code Python associé.

Ligne 100 environ du fichier SDES.py :

```
Temps moyen pour chiffrer et dechiffrer en double SDES la phrase b'Ce texte est en clair. Il est vraiment tres clair.': 3.680s
Temps moyen pour chiffrer et dechiffrer en AES la phrase b'Ce texte est en clair. Il est vraiment tres clair.': 0.347s
```

Les durées ont été multipliés par 1000 pour avoir des valeurs différentes de 0 mais l'AES est environ 10 fois plus rapide.

B) Le temps de cassage d'AES (même pour un cassage astucieux) si vous deviez l'exécuter sur votre ordinateur.

Sachant que je peux calculer 1000 clés en 0.3s pour AES, cela fait environ 3000 clés testées par seconde et sachant qu'il y a 2^256 clés possibles, je mettrai 2^256/(3000*60*60*24*365) = 1.22x10^66 années à cracker AES en brute force, et même en cassage astucieux, cela représente une durée beaucoup trop grande ainsi qu'un stockage quasi infini (pour une attaque meet in the middle).

Ici il faut uniquement estimer le temps nécessaire (sinon vous ne pourriez pas rendre votre rapport à temps !). Vous préciserez votre configuration et vous fournirez le détail des calculs

3) Il existe d'autres types d'attaques que de tester les différentes possibilités de clés. Lesquelles ? Vous donnerez une explication succincte de l'une d'elles

D'autres types d'attaques pourraient être par canal auxiliaire, évitant de s'attaquer à la complexité mathématique de ces fonctions mais nécessitant de trouver une faille dans la méthode de Alice et Bob et pas dans le protocole en lui-même mais sans accès à leurs machines, elle ne pourra pas utiliser d'autres méthodes.

Partie 3 :

Résolution de l'énigme

fichier python utilisé : programmeImages.py, programmeTraceRéseau.py

la deuxième image contient la clé de session sur 64 bits
Pour obtenir cette clé de session, j'ai parcouru l'image pixel par pixel j'ajoute le pixel à ma clé et je sors de la boucle une fois que la longueur de ma clé est égale à 64 bits

j'obtiens donc ma clé de session :

```
CLE_BINAIRE = "1110011101101101001100010011111111001001010111001100100000100110 0"
```

Ensuite, il faut ensuite mettre la clé en hexadécimale pour pouvoir l'utiliser sur nos paquets codés en hexa
J'utilise filtrer_paquets(trace) pour récuperer la liste des paquets UDP sur le port de destination 9999 et j'uilise charger_messages(trace) pour récuperer ces métadonnées puis j'utilise la méthode decrypter_message(cle, trace) avec la cle obtenu en amont et mis sous la bonne forme pour l'utiliser avec AES et Cipher

J'obtiens les messages suivants :

```
La crypto c'est trop bien!
Je suis complètement d'accord!
```

Partie 4 :

1. Alice et Bob utilisent toujours la même clé. Est-ce une bonne pratique?

   Non, car si l'on arrive à trouver la clé de session permettant de crypter l'échange entre Alice et Bob, nous pouvons déchiffrer tous leurs échanges. Cependant, si la clé change régulièrement il ne sera pas possible de déchiffrer les prochains échanges ou du moins difficilement

2. Le protocole PlutotBonneConfidentialité est inspiré d'un vrai protocole réseau. Lequel? Décrivez la partie associé à la certification des clés qui est absente de PlutotBonneConfidentialité.

Le protocole PlutotBonneConfidentalité est inspiré du protocole SSL

3. Il n'y a pas que pour l'échange de mots doux q'un tel protocole peut se révéler utile. .Donnez au moins deux autres exemples de contexte où cela peut se révéler utile.

   Il y a les transactions en lignes où ce protocole peut être utilié tel que Amazon pour crypter les données sensibles, telles que les informations de paiement, lors de la transmission entre le client et les serveurs d'Amazon. Également, les accès sécurisés aux réseaux sociaux par exemple afin de sécuriser les connexions entre les utilisateurs et leurs serveurs.

4. Connaissez-vous des applications de messagerie utilisant des mécanismes chiffrement similaires? (on parle parfois de chiffrement de bout en bout)? Citez-en au moins deux et décrivez brièvement les mécanismes cryptographiques sous-jacent.

   Il y a SnapChat ou encore Messenger qui utilise le chiffrement de bout en bout pour sécuriser les conversations entre les utilisateurs. Ce protocole garantit que seuls les participants à une conversation peuvent lire les messages échangés. Lorsqu'un message est envoyé, il est chiffré sur l'appareil de l'émetteur à l'aide de la clé publique du destinataire. Une fois le message reçu, il est déchiffré à l'aide de la clé privée du destinataire

5. Récemment, différents projets de loi et règlements (CSAR, EARN IT Act) visent à inciter voir obliger les fournisseurs de services numériques à pouvoir déchiffrer (et donc analyser) les communications de leur.e.s utilisateur.rices. Discutez des arguments en faveur ou contre ces législations, notamment en matière de vie privée.

   Cela permet de renforcer la sécurité de tout le monde en permettant aux autorités d'accéder aux communications chiffrées lorsqu'elles sont liées à des activités criminelles ou malveillantes. Cependant, cela pourrait constituer une violation de la vie privée et avoir des conséquences néfastes pour les droits individuels.

Répartitions des tâches :

Sullivan DEMARET : Partie 2 (Images) + Partie 3 + Partie 4
Titouan COULON : Partie 1 + Partie 2 (AES)