



**POLYTECHNIQUE
MONTRÉAL**

INF4420A – SECURITE INFORMATIQUE

Travail Pratique 2– Hiver 2020

Jean Fikani Wajiha Bissola Badirou

1847428 | 1770039

Table of Contents

Partie A.....	2
Question 1 - Entropie [/0.75].....	2
Question 2 - Histogrammes [/0.75]	4
Question 3 - Masque jetable [/0.75]	8
Question 4 - Analyse de risque [/1.5].....	10
Partie B.....	13
Question 1 - Codage [/1.25].....	13
Question 2 - Certificats à clé publique, HTTPS et SSL [/1]	18
Question 3 - Chiffrement par bloc et modes d'opération [/0.5]	30
Question 4 - Organisation des mots de passe en UNIX/Linux [/1].....	34
Question 5 - Contrôle de qualité de choix de mot de passe [/1].....	41
Partie C.....	43
Question 1 - Échec du protocole RSA [/0.75].....	43
Question 2 - Déchiffrement "simple" [/0.75]	46

Partie A

Question 1 - Entropie [/0.75]

a)

```
parallels@wajihhas-linux-machine:~/Downloads/Utilitaires TP1/Source - Entropie - Chiffrement$ ./texte 200 | ./h-lettre
(space) = 36
A = 11
B = 4
C = 4
D = 9
E = 22
F = 5
G = 4
H = 12
I = 7
J = 0
K = 0
L = 7
M = 2
N = 6
O = 10
P = 5
Q = 0
R = 11
S = 8
T = 21
U = 8
V = 0
W = 2
X = 1
Y = 5
Z = 0
Nombre total de caracteres : 200
Entropie de l'entree : 4.028952
```

Comme on peut le voir, nous obtenons une entropie de 4.028952 bits par symbole

- b) En se servant de l'entropie de Shannon, on peut dire que chaque symbole est codé individuellement en utilisant $H(S)$ bits. De plus, le code compresseur possède une efficacité de $H(S) + 1$. Nous obtenons alors en moyenne une entropie de $H(S)$.

- c) La formule de l'entropie de Shannon est la suivante : $\sum P_i \log_2 \left(\frac{1}{P_i} \right)$ avec $P_i = \frac{1}{27}$.

Ainsi l'entropie par lettre serait de 4.75 bits par symbole.

- d) Le quotient de la valeur en a) sur la valeur en c) est de 0.848 (4.028952 / 4.75) et représente la dispersion des lettres.

e)

```
parallels@wajih-as-linux-machine:~/Downloads/Utilitaires TP1/Source - Entropie - Chiffrement$ ./lettre 200 | ./h-lettre
(space) = 24
A = 14
B = 4
C = 3
D = 4
E = 18
F = 6
G = 1
H = 17
I = 16
J = 1
K = 3
L = 4
M = 3
N = 7
O = 8
P = 3
Q = 0
R = 12
S = 20
T = 17
U = 7
V = 1
W = 5
X = 0
Y = 2
Z = 0
Nombre total de caracteres : 200
Entropie de l'entree : 4.112484
```

Comme nous pouvons le voir, nous obtenons une entropie de 4.112484 bits par symbole

- f) Le calcul de l'entropie ne tient pas compte des mots de la langue mais surtout de la fréquence de l'apparition de chaque lettre. Ainsi, lorsque le texte généré n'est pas long on peut se retrouver avec des entropies très proches malgré que ces différents textes ne soient pas dans la même langue, ce qui est notre cas ici.

Question 2 - Histogrammes [/0.75]

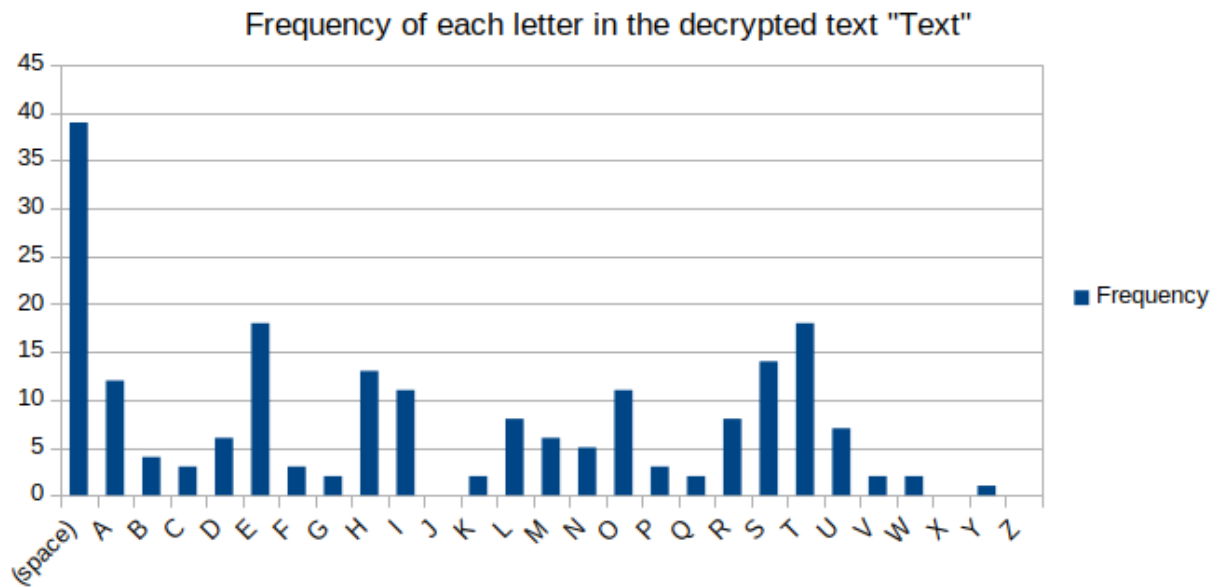
- a) Dans la première image, nous avons le chiffrement et le déchiffrement d'un texte de 200 caractères. Tandis que dans la deuxième image, nous avons le chiffrement et le déchiffrement d'une chaîne de 200 caractères.

```
parallels@wajihhas-linux-machine:~/Downloads/Utilitaires TP1/Source - Entropie - Chiffrement$ ./texte 200 > text
parallels@wajihhas-linux-machine:~/Downloads/Utilitaires TP1/Source - Entropie - Chiffrement$ cat text
IM AS HE REQUESTS BUT HOW NOT SERUILELY DISPOSD TO BEND BUT LIKE A CONQUERER TO
MAKE HIM BOWE HIS LAME VNPOLISHT SHIFTS ARE COME TO LIGHT AND TRUETH HATH PULD T
HE VISARD FROM HIS FACE THAT SETT A GLAS
parallels@wajihhas-linux-machine:~/Downlo
Chiffrement$ es TP1/Source - Entropie - C
parallels@wajihhas-linux-machine:~/Downloads/Utilitaires TP1/Source - Entropie - Chiffrement$ ./cesar < text > codedText
parallels@wajihhas-linux-machine:~/Downloads/Utilitaires TP1/Source - Entropie - Chiffrement$ cat codedText
LP DV KH UHTXHVWV EXW KRZ QRW VHUXLOHOB GLVSRVG WR EHQG EXW OLNH D FRQTXHUHU WR
PDNH KLP ERZH KLV ODPH YQSROLVKW VKLIWV DUH FRPH WR OLJKW DQG WUXHWK KDWK SXOG W
KH YLVDUG IURP KLV IDFH WKDW VHWW D JODV
parallels@wajihhas-linux-machine:~/Downlo
Chiffrement$ .s TP1/Source - Entropie - C
bash: .: filename argument required
.: usage: . filename [arguments]
parallels@wajihhas-linux-machine:~/Downloads/Utilitaires TP1/Source - Entropie - Chiffrement$ ./cesar-d < codedText
IM AS HE REQUESTS BUT HOW NOT SERUILELY DISPOSD TO BEND BUT LIKE A CONQUERER TO
MAKE HIM BOWE HIS LAME VNPOLISHT SHIFTS ARE COME TO LIGHT AND TRUETH HATH PULD T
HE VISARD FROM HIS FACE THAT SETT A GLAS
parallels@wajihhas-linux-machine:~/Downlo
Chiffrement$ .s TP1/Source - Entropie - C
```

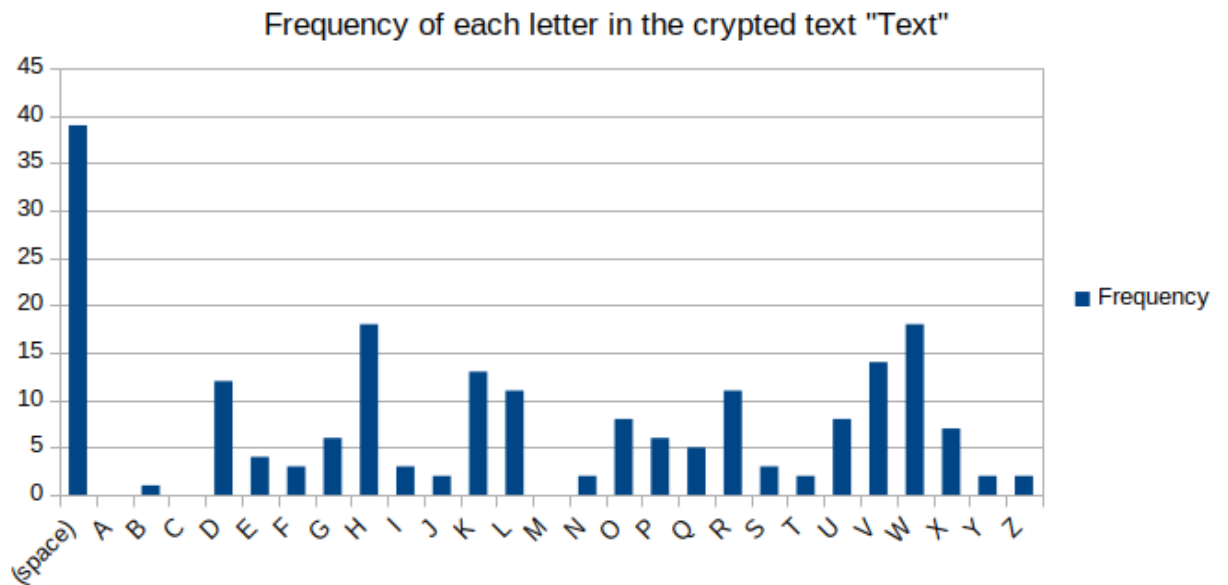
```
parallels@wajihhas-linux-machine:~/Downloads/Utilitaires TP1/Source - Entropie - Chiffrement$ ./lettre 200 > letter
parallels@wajihhas-linux-machine:~/Downloads/Utilitaires TP1/Source - Entropie - Chiffrement$ cat letter
WERH GSOI EIOTNAROCHFO T KWALB SEHEALHPRWECL O HJK TBEI U EII SSNCOWNLTOHCLUGTPLTEL
OVHOGWGRIKEHOJS OOOTVAU HPTHHEAT HMW SWMV BJVOEFDEII GUONEHEHTEITRI T TGSU HWEAHFWII
UYE UTOR BHFVE T CEASATHHLXRTEBW
parallels@wajihhas-linux-machine:~/Downloads/Utilitai
frement$ ource - Entropie - Chiff
parallels@wajihhas-linux-machine:~/Downloads/Utilitaires TP1/Source - Entropie - Chiffrement$ ./cesar < letter > codedLetter
parallels@wajihhas-linux-machine:~/Downloads/Utilitaires TP1/Source - Entropie - Chiffrement$ cat codedLetter
ZHUK JVRL HLRWQDURFKIR W NZDOE VHKHOKSUZHFO R KMN WEHL X HLL VVQFGZQOWRKFOXJWSOWHO
RYKRJZZJULNHKRMV RRRWYDX KSWKKHDW KPZ VZPY EMYRHIGHLL JXRQHKHKWHLWUL W WJVB KZHDKIZLL
XBH XWRU EKIYH W FHDVDWKKOAUWHEZ
parallels@wajihhas-linux-machine:~/Downloads/Utilitai
frement$ ource - Entropie - Chiff
parallels@wajihhas-linux-machine:~/Downloads/Utilitaires TP1/Source - Entropie - Chiffrement$ ./cesar-d < codedLetter
WERH GSOI EIOTNAROCHFO T KWALB SEHEALHPRWECL O HJK TBEI U EII SSNCOWNLTOHCLUGTPLTEL
OVHOGWGRIKEHOJS OOOTVAU HPTHHEAT HMW SWMV BJVOEFDEII GUONEHEHTEITRI T TGSU HWEAHFWII
UYE UTOR BHFVE T CEASATHHLXRTEBW
parallels@wajihhas-linux-machine:~/Downloads/Utilitai
frement$ ource - Entropie - Chiff
parallels@wajihhas-linux-machine:~/Downloads/Utilitaires TP1/Source - Entropie - Chiffrement$
```

b)

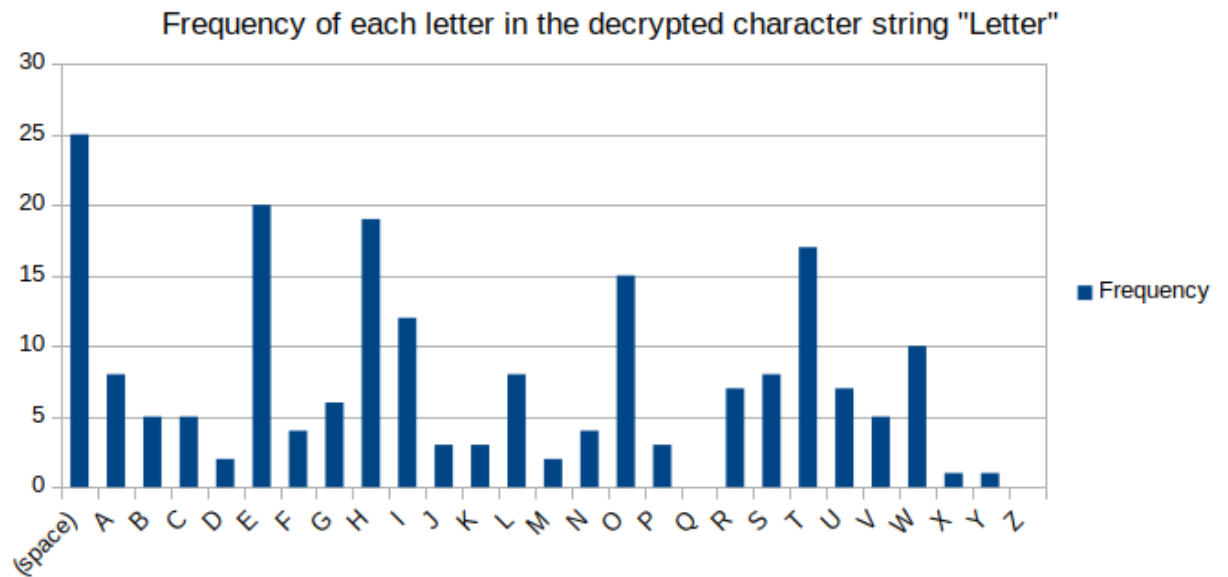
Graphique 1 : Histogramme texte déchiffré



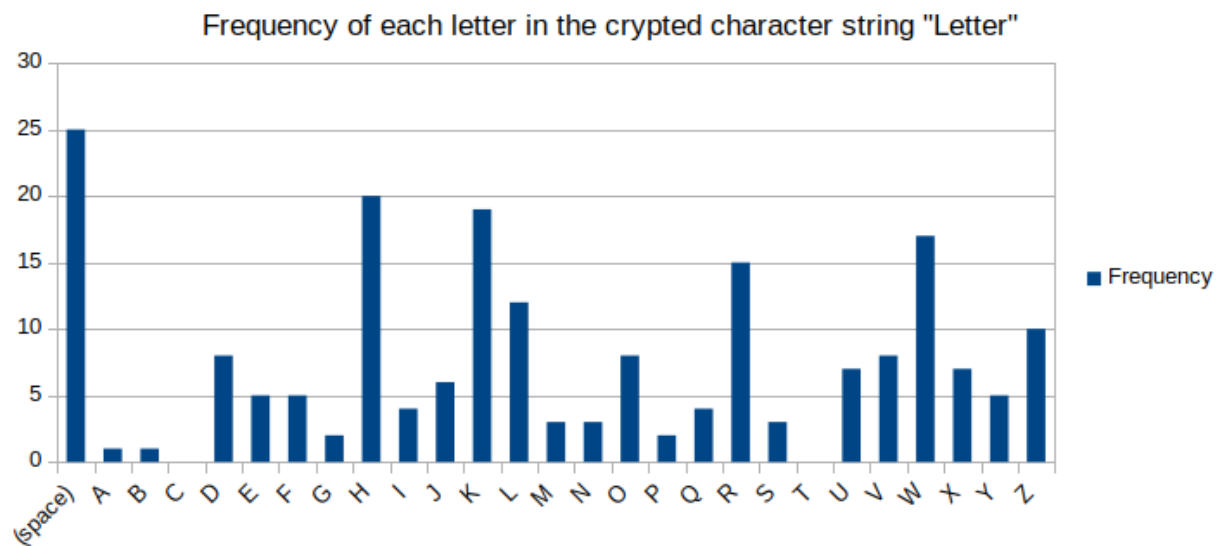
Graphique 2 : Histogramme texte chiffré



Graphique 3 : Histogramme chaîne de caractères déchiffrée



Graphique 4 : Histogramme chaîne de caractères chiffrée



- c) On remarque que les histogrammes de chaque texte sont assez similaires. Si les fréquences étaient comptabilisées deux lettres à la fois, les histogrammes seraient plus différents car on aurait des couples plus fréquents dans l'un ou dans l'autre. Or, avec les fréquences des lettres, on ne fait pas trop la différence entre un chiffrement et un déchiffrement. Les fréquences du *ee* ou *th* seraient surement les plus fréquentes.
- d) Cette méthode facilite le déchiffrement dans le cas de *texte* mais pas forcément dans *lettre*. Dans *texte* qui est un texte de la langue anglais, il y a certains doublets de lettres qui sont plus fréquents donc facilite le déchiffrement. Or, dans le 2e cas, les lettres sont choisies aléatoirement et l'ordre n'est pas conservée. Pour *lettre*, il faudrait essayer avec la fréquence de chaque lettre dans la langue anglaise.

Question 3 - Masque jetable [/0.75]

a)

```
wabad@ubuntu:~/Documents/INF4420A/Utilitaires TP1/Source - Entropie - Chiffrement$ ./monnaie 1024 > monnaie1024
wabad@ubuntu:~/Documents/INF4420A/Utilitaires TP1/Source - Entropie - Chiffrement$ ./h-bit < monnaie1024
0 = 4076
1 = 4116
Nombre total de bits : 8192
Entropie du texte entre : 0.999983
wabad@ubuntu:~/Documents/INF4420A/Utilitaires TP1/Source - Entropie - Chiffrement$ ./h-ascii < monnaie1024
Nombre total d'octets : 1024
Entropie de l'entree : 7.834079
wabad@ubuntu:~/Documents/INF4420A/Utilitaires TP1/Source - Entropie - Chiffrement$ ./binaire 1024 > binaire1024
wabad@ubuntu:~/Documents/INF4420A/Utilitaires TP1/Source - Entropie - Chiffrement$ ./h-bit < binaire1024
0 = 5120
1 = 3072
Nombre total de bits : 8192
Entropie du texte entre : 0.954434
wabad@ubuntu:~/Documents/INF4420A/Utilitaires TP1/Source - Entropie - Chiffrement$ ./h-ascii < binaire1024
Nombre total d'octets : 1024
Entropie de l'entree : 0.811278
```

Avec le fichier créé à partir de *monnaie*, on obtient une entropie par bit de **0.999983** et une entropie par octet de **7.834079**. Tandis que pour le fichier créé à partir de *binaire*, on obtient une entropie de **0.954434** et une entropie par octet de **0.811278**.

b)

```
wabad@ubuntu:~/Documents/INF4420A/Utilitaires TP1/Source - Entropie - Chiffrement$ ./monnaie 1024 > cle
wabad@ubuntu:~/Documents/INF4420A/Utilitaires TP1/Source - Entropie - Chiffrement$ ./masque cle 1024 monnaie1024 monnaie1024C
hiffree
wabad@ubuntu:~/Documents/INF4420A/Utilitaires TP1/Source - Entropie - Chiffrement$ ./h-bit < monnaie1024Chiffree
0 = 4033
1 = 4159
Nombre total de bits : 8192
Entropie du texte entre : 0.999829
wabad@ubuntu:~/Documents/INF4420A/Utilitaires TP1/Source - Entropie - Chiffrement$ ./h-ascii < monnaie1024Chiffree
Nombre total d'octets : 1024
Entropie de l'entree : 7.809044
wabad@ubuntu:~/Documents/INF4420A/Utilitaires TP1/Source - Entropie - Chiffrement$ ./binaire 1024 cleBinaireUsage : binaire [
nb_octets]
wabad@ubuntu:~/Documents/INF4420A/Utilitaires TP1/Source - Entropie - Chiffrement$ ./masque cle 1024 binaire1024 binaire1024C
hiffree
wabad@ubuntu:~/Documents/INF4420A/Utilitaires TP1/Source - Entropie - Chiffrement$ ./h-bit < binaire1024Chiffree
0 = 4039
1 = 4153
Nombre total de bits : 8192
Entropie du texte entre : 0.999860
wabad@ubuntu:~/Documents/INF4420A/Utilitaires TP1/Source - Entropie - Chiffrement$ ./h-ascii < binaire1024Chiffree
Nombre total d'octets : 1024
Entropie de l'entree : 7.825459
wabad@ubuntu:~/Documents/INF4420A/Utilitaires TP1/Source - Entropie - Chiffrement$
```

Pour le fichier *monnaie chiffrée*, on obtient une entropie par bit de **0.99829** et une entropie par octet de **7.809044** après l'application du masque. Tandis que pour le fichier *binaire chiffrée*, on obtient une entropie par bit de **0.99860** et une entropie par octet de **7.82459**.

On remarque pour ces deux fichiers que leurs entropies par bit et par octets sont presque entièrement identiques. On peut tirer comme conclusion que ce n'est pas le nombre symboles dans l'alphabet utilisé pour chiffrer chaque message qui compte mais celui utilisée pour la clé.

- c) Chaque terme est chiffré aléatoirement et, de plus, on peut voir que l'entropie par octets augmente donc on peut dire que la méthode de chiffrement est assez sécuritaire.

Question 4 - Analyse de risque [/1.5]

a) A vu d'œil, le site B semble plus avantageux car nous sauverons 400,000.00\$ à s'installer sur cette île. Certes, elle reste une île avec 25% de risque que nous perdions notre investissement. Pour être plus précis, nous perdrons plus que notre investissement initial, car il y aura des dommages collatéraux et humains, et ceci peut arriver à n'importe quel moment. Il est préférable d'investir plus et d'assurer un environnement sécuritaire aux employés, aux matériels et faire un placement fiable que prendre un risque pareil juste pour sauver de l'argent. Dans cette logique, nous préférons l'île A.

b) Le scénario **i** touche l'intégrité, tandis que, le scénario **ii** touche à la disponibilité. Pour finir, le scénario **iii** touche à la confidentialité.

c) **PROBABILITE = (CAPACITE + OPPORTUNITE + MOTIVATION) / 3**

RISQUE = (PROBABILITE + IMPACT) / 2

On surligne en jaune l'acteur qui constitue la plus grande menace pour l'entreprise

	Acteur	Capacité	Opportunité	Motivation	Probabilité	Impact	Risque
Scenario i	Tricheur	4	4	4	4	2	3
	C.O.	1	4	1	2	2	2
	Concurrents	2	4	2	2.67	2	2.335

	Acteur	Capacité	Opportunité	Motivation	Probabilité	Impact	Risque
Scenario ii	Tricheur	1	4	1	2	4	3
	C.O.	4	4	1	3	4	3.5
	Concurrents	2	4	4	3.3	4	3.65

	Acteur	Capacité	Opportunité	Motivation	Probabilité	Impact	Risque
Scenario iii	Tricheur	1	3	1	1.67	3	2.34
	C.O.	4	3	4	3.67	3	3.17
	Concurrents	1	3	2	2	3	2.5

- d) 1. Cette situation entraînerait une hausse de la motivation des concurrents, ce qui augmenterait le risque lié à cet acteur et le faire devenir un risque prioritaire. On devient alors plus enclin à recevoir une attaque de leur part
2. Cette situation entraînerait une hausse de la motivation des crimes locaux, ce qui augmenterait le risque lié à cet acteur et le faire devenir un risque prioritaire.
3. Dans cette situation, l'opportunité des tricheurs diminue, ce qui diminuerait le risque lié à cet acteur, et possiblement le faire descendre dans les priorités des risques à gérer.

e)

	Acteur	Capacité	Opportunité	Motivation	Probabilité	Impact	Risque
Scenario iii	Tricheur	1	3	1	1.67	3	2.34
	C.O.	4	2	4	3.67	3	3.17
	Concurrents	1	3	2	2	3	2.5

Avec l'offre que l'on nous propose, nous n'avons une possible meilleure protection que contre le crime organisé. Ce sont eux qui voient leur opportunité diminuer.

D'un autre côté, délocaliser sa sécurité peut s'avérer être un grand risque, surtout dans un pays avec la main d'œuvre moins chère. Nous ne connaissons pas les méthodes de travail, leur intégrité en ce qui concerne les informations. Notre entreprise est dans le domaine du poker en ligne, donc nous travaillons avec des données très sensibles, les exposer de la sorte peut s'avérer catastrophique. Nous nous créons ainsi une nouvelle faille potentielle. Il est préférable d'attribuer un budget conséquent et être rassuré du travail qui nous sera fourni, donc nous ne recommandons pas cette offre. Cette recommandation se limite à du cas par cas, certaines entreprises dans d'autres domaines n'y verront pas d'inconvénient et au contraire y verront un avantage d'avoir une surveillance à distance.

Partie B

Question 1 - Codage [/1.25]

a) Cas où le codage ne change pas:

Alors les différents alphabets sont des ensembles comme : $s = \{0-9\}$; $t = \{0-9\}$;
 $t' = \{0,1\}$.

b)

- Langue L1: Disposition de quatre symboles de l'alphabet s .
- Langue L2: $m.m$ avec m mots de la première langue.
- Langue L3: bloc de taille 64 bits composé de 0 et 1, t' .

c) Interception d'un message crypté sur le réseau:

- Attaque par force brute du type "texte chiffré uniquement" sans pouvoir chiffrer les messages. Nous avons testé toutes les chances que les clés soient de 2^{56} pour DES.
- Attaque de texte connue, accès au texte brut et au texte crypté alors n'oubliez pas 10000 combinaisons pour un nouveau NIP du client.

Accès à une boîte noire :

- Choix d'attaque en texte clair: possibilité de choisir du texte clair et d'accéder à une boîte noire qui crypte chaque message avec la clé que vous recherchez.
- Attaque par texte chiffré choisi: accès à plusieurs textes chiffrés et à une boîte noire qui déchiffre chaque message avec la clé qui a rendu possible le chiffrement donc un Cas symétrique vers DES

Attaque par rejeu:

De plus, la déclaration nous dit qu'un attaquant peut intercepter et même modifier des messages chiffrés. Dans une telle situation, un attaquant en profitera pour saisir son propre code NIP afin d'accéder au compte de la victime, sans avoir à décrypter aucun message.

On note également l'absence d'application d'une fonction de hachage dans le chiffrement.

- d) La boîte noire nous permettrait de mener les attaques décrites ci-dessus en exploitant un grand nombre d'entrées et en observant les sorties résultantes. Par essais et erreurs, nous trouvons éventuellement des modèles utiles pour reconstruire la clé.

```
[jefik@l4712-18 Codage] $ ./transBase.py 1234
0!S04yE[jefik@l4712-18 Codage] $ ./transBase.py 1234
0!S04yE[jefik@l4712-18 Codage] $
```

```
0!S04yE[jefik@l4712-18 Codage] $ ./transBase.py 1234 > nipB
[jefik@l4712-18 Codage] $ ./recepBase.py < nipB
1234[jefik@l4712-18 Codage] $
```

Cas où le codage change

e)

```
1234[jefik@l4712-18 Codage] $ ./trans1.py 1234 > nip1
[jefik@l4712-18 Codage] $ ./recep1.py < nip1
1234[jefik@l4712-18 Codage] $ ./trans2.py 1234 > nip2
[jefik@l4712-18 Codage] $ ./recep2.py < nip2
Delaï de transmission suspect, operation annulee
[jefik@l4712-18 Codage] $ ./trans3.py 1234 > nip3
[jefik@l4712-18 Codage] $ ./recep3.py < nip3
Delaï de transmission suspect, operation annulee
[jefik@l4712-18 Codage] $
```

- f) Ainsi, le premier et commun dans les trois codages. Après analyse des programmes trans1.py, trans2.py et trans3.py, les bits de remplissage sont générés pour chacun des trois codages selon un algorithme déterministe et potentiellement réversible. Donc, pour les deux premiers encodages, cela

représente 2 bits à 64 bits. Ratio théorique $2/64 = 3,125\%$. Dans le cas du codage 3, par conséquent, le bit 4 bits de 64 est connu, ce qui représente un rapport théorique de $4/64 = 6,25\%$. Cependant, cela a déjà une faille de sécurité qui permet des attaques fréquentes et/ou des attaques connues en texte brut. Ces bits de remplissage auraient dû être produites au hasard.

L'algorithme commence par convertir le code NIP décimal en quatorze binaires. Ensuite, convertissez-le en une chaîne de caractères et coupez-le en deux parties égales à 7. Mesurez ensuite le nombre de «1» pour chacune de ces deux parties et créez un module 2. Si le nombre de «1» est pair, un peu sera ajouté à «0», sinon si le nombre de «1» est impair, il ajoutera un peu à «1». Nous savons rapidement qu'avec la connaissance de la boîte noire et de l'algorithme utilisé, il est très facile de créer 10000 combinaisons, d'appliquer le même algorithme de remplissage, d'observer les nombres qui répondent à ces critères et de filtrer en raison de la réduction de l'espace numérique candidats. Tout cela nous fournit des informations utiles pour notre cryptanalyse. En fait, ces deux bits de remplissage nous permettent d'isoler le code NIP d'un ensemble de petits nombres inférieurs à 10000. Nous savons désormais qu'un algorithme déterministe fournit essentiellement un certain nombre d'informations, ce qui signifie en d'autres termes, que nous réduisons l'entropie du système, et donc sa sécurité globale. L'algorithme utilisé ici à une logique pour ajouter et créer ces bits de remplissage et cette logique sous-jacente génère de manière inhérente beaucoup d'informations, ce qui réduit l'entropie et est douloureux de bénir un attaquant.

Dans le cas du cryptage 2, selon le nombre de clients changeant leur code NIP, le risque de collision n'est pas si faible. En effet, compte tenu du nombre de bits alloués aux deux premiers champs, le soudage de ces deux éléments produit une chaîne binaire de 32 bits de taille théorique. Ainsi, au mieux, le risque de collision est d'environ 2^{32} pour deux clients actifs. Étant donné que l'unicité est basée uniquement sur un nombre 32 bits, la somme du nouveau code NIP avec un nombre aléatoire, alors le timestamp est décisif, si deux messages sont envoyés à la même milliseconde, les chances de collision pendant ce millimètre. Les secondes sont de 2^{32} , c'est-à-dire qu'elles sont similaires ou presque le double du risque total de collision si 2^{32} messages sont envoyés avec des informations aléatoires à 96 bits sans timestamp. En fait, les

chances de collision sont plus importantes dans la pratique car, comme indiqué dans la déclaration, un attaquant peut très bien savoir comment fonctionnent les boîtes de code.

Vous pouvez connaître les bits de parité car ils ne sont pas a priori aléatoires. Bien sûr, si la probabilité que deux messages soient envoyés à la même milliseconde est suffisamment faible et peut être garantie, même en cas d'attaque, l'ajout d'un timestamp rendrait le travail d'un attaquant encore plus difficile. Je peux imaginer, mais je ne pense pas. Vous pouvez penser à un scénario: attaquant peut forcer les clients à utiliser le même timestamp pour augmenter le risque de collision, pour obtenir un avantage. Dans ce cas, un nonce purement aléatoire serait la meilleure option.

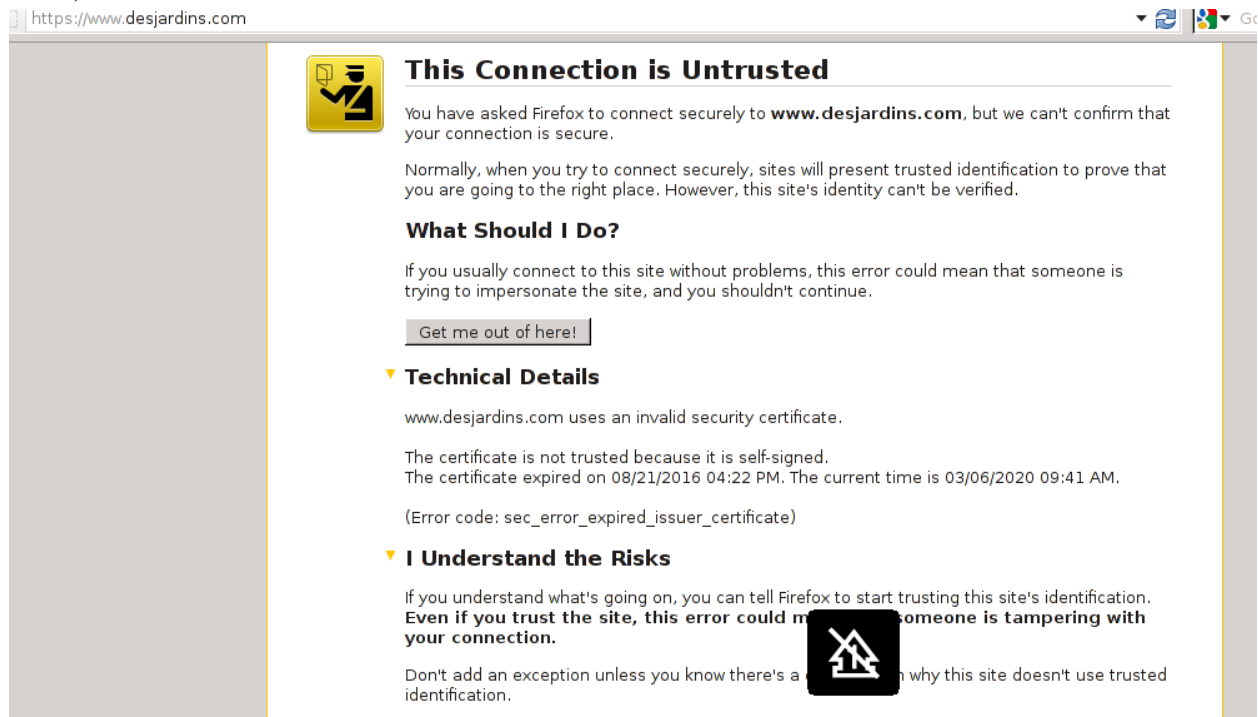
De plus, j'ai l'impression que l'encodage 2 est en fait encore plus enclin à répéter les attaques que l'encodage 1, qui utilise un nonce aléatoire de 48 bits. En fait, le codage 2 utilise seulement 16 bits pour produire un nombre aléatoire représentant deux caractères. En prenant la table ASCII comme source d'entrée, cela représente $2^{16} = 256^2 = 65536$ combinaisons. La génération d'un code NIP est très simple à calculer et représente environ $10^4 = 10000$ combinaisons. Nous avons donc $65\,536 + 10\,000 = 75\,536$ combinaisons au total. Cependant, selon notre déclaration et notre analyse, les deux bits de parité ne sont pas aléatoires et peuvent donc être connus à l'avance, tout comme l'estampille temporelle prévisible. En outre, selon le communiqué de presse, la banque utilise le timestamp pour annuler les messages dont le timestamp est trop ancien. Mais est-il possible pour un attaquant de produire ces 75 536 combinaisons avant l'expiration de la banque? Tout dépend du temps!

Le codage 1 a un nombre aléatoire de 48 bits. Nous calculons le nombre de combinaisons: $2^{48} = 281474976710656$ combinaisons, ici 10000 combinaisons de codes PIN sont insignifiantes, ce qui représente un nombre de combinaisons beaucoup plus élevé que dans le cas des deux codages précédents et offrent, à mon avis, la sécurité.

Le chiffrement 3 ne semble pas crédible car si le système de transmission est initialement vulnérable au communiqué de presse et peut donc être compromis, l'attaquant peut connaître l'ancien code PIN utilisé. De plus, comme vous savez potentiellement comment fonctionne la boîte de chiffrement, vous savez qu'un timestamp de 32 bits est utilisé de façon causale, que les anciens bits + 2 bits sont potentiellement connus comme déterministes et que le nouveau PIN + 2 bits est utilisé, seulement 10000 combinaisons. La codification 3 est, à mon avis, très opportuniste, ce qui est contraire au principe de confusion et de chaos dans le cryptage.

Question 2 - Certificats à clé publique, HTTPS et SSL [/1]

a)



Un écran apparaît avec le message que cette connexion n'est pas fiable. La connexion est considérée comme peu sûr. Nous ne pouvons pas nous y connecter car Firefox génère une alerte de sécurité. En fait, le serveur utilise un certificat auto-signé et non un certificat émis et vérifié par une autorité de certification de confiance.

- b) C'est un site bancaire, donc l'authenticité devrait être un objectif important, mais le certificat ce n'est pas vérifié, alors ce qui est suspect. Le site Web d'une société bancaire comme Desjardins doit utiliser des certificats émis et vérifiés par une autorité de certification de confiance (un tiers). Par conséquent, il est paradoxal que le site de ladite institution utilise un certificat auto-signé qui porte atteinte à sa réputation et affecte la confiance de ses clients.

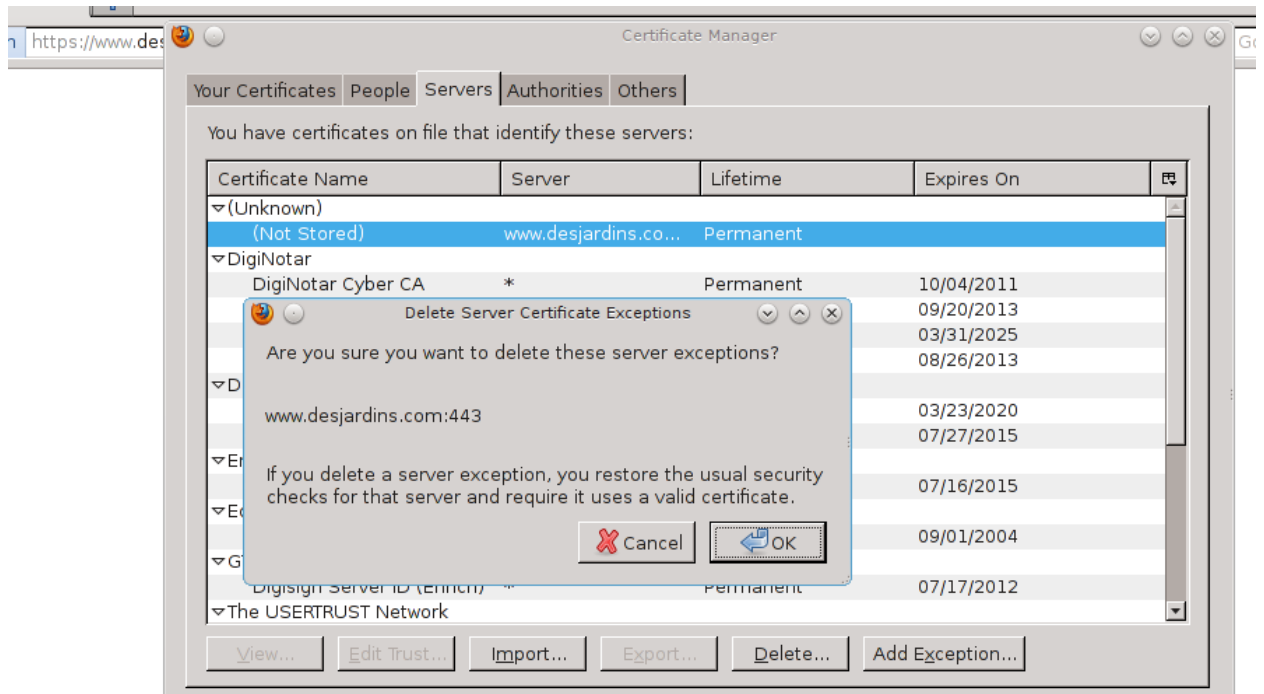
c)



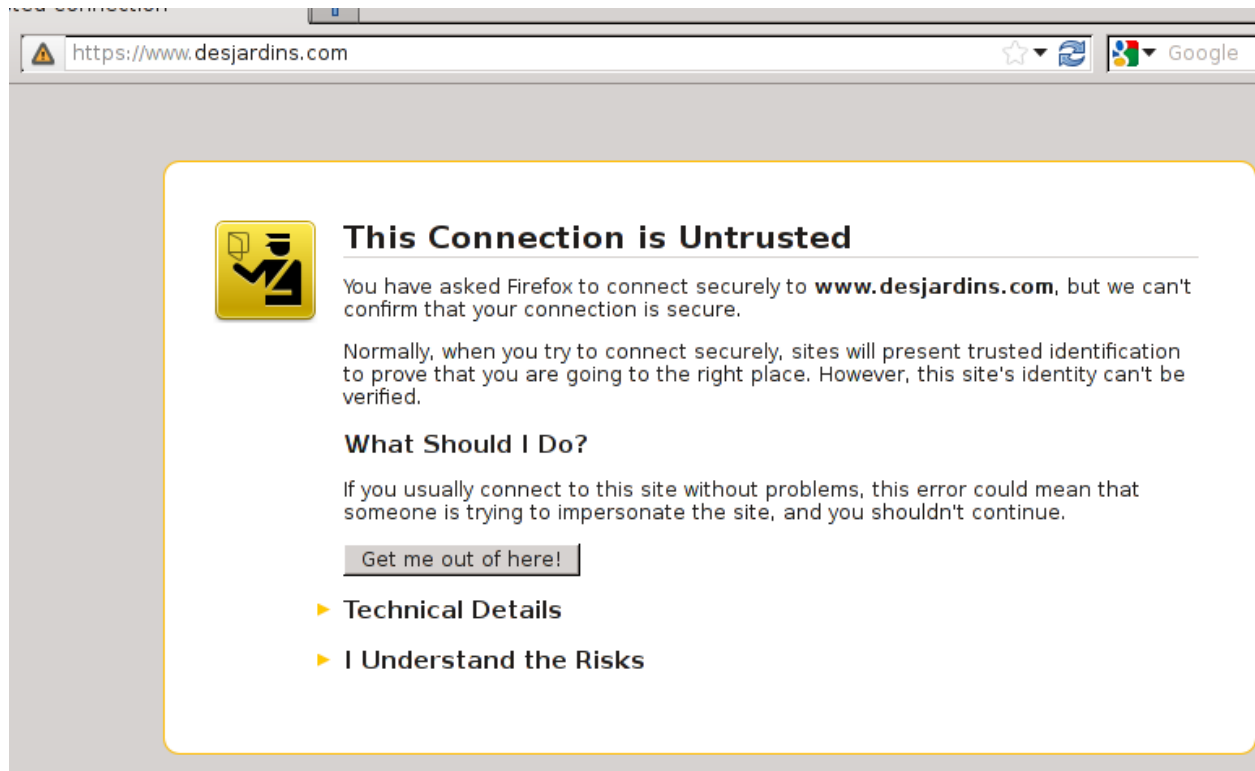
Maintenant, nous avons accès au site, le certificat est ajouté au navigateur, le site est considéré comme fiable par Firefox.

Désormais, Firefox affiche le site sans alerte de sécurité car nous ajoutons une exception. Par conséquent, le certificat figure désormais dans la liste des certificats approuvés.

d)

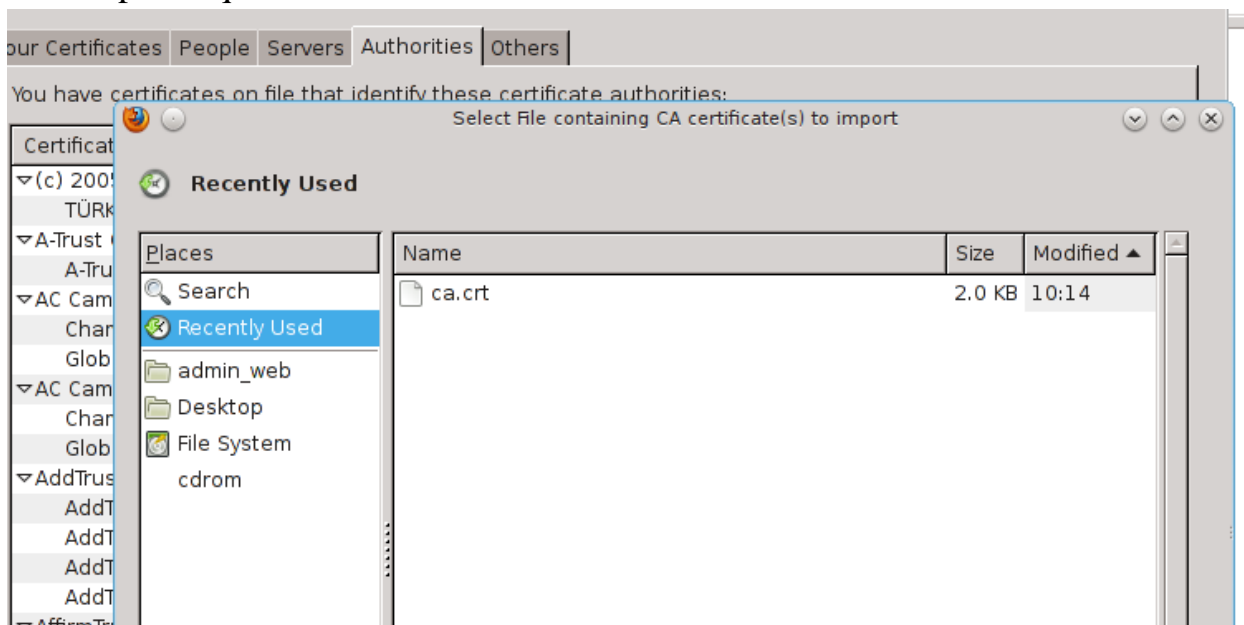


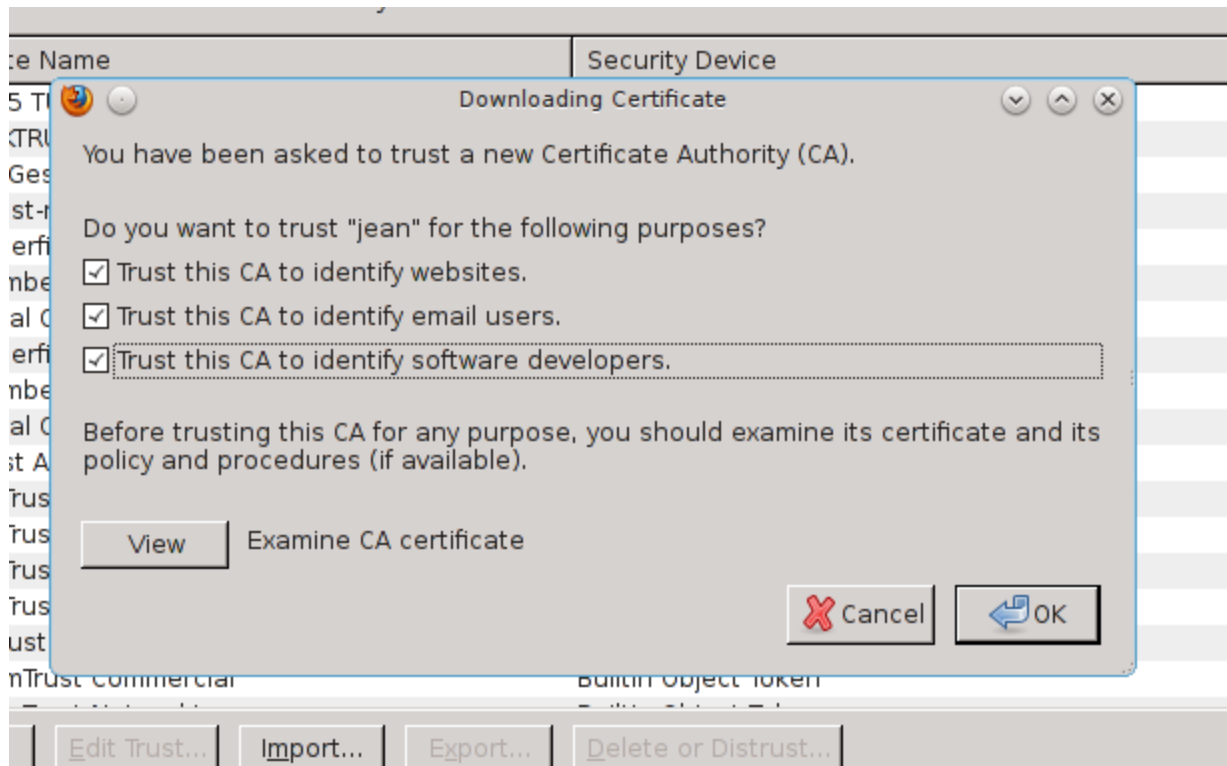
e)



```
~ : bash
File Edit View Bookmarks Settings Help
admin_web@certificates ~ $ openssl genrsa -des3 -out ca.key 4096
Generating RSA private key, 4096 bit long modulus
.....++
.....++
e is 65537 (0x10001)
Enter pass phrase for ca.key:
Verifying - Enter pass phrase for ca.key:
admin_web@certificates ~ $ openssl req -new -x509 -days 365 -key ca.key -out ca.crt
Enter pass phrase for ca.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:ca
State or Province Name (full name) [Some-State]:quebec
Locality Name (eg, city) []:laval
Organization Name (eg, company) [Internet Widgits Pty Ltd]:poly
Organizational Unit Name (eg, section) []:ing
Common Name (e.g. server FQDN or YOUR name) []:jean
Email Address []:jeanfikani@gmail.com
admin_web@certificates ~ $
```

Nous avons accès au site Desjardins, car Firefox trouve un certificat pour ce site, donc il pense que c'est fiable.





Création du certificat de l'autorité de certification

Une clé d'une taille de 4096bits est générée par un cryptage triple DES. Ce dernier nécessite une phrase secrète qui vous sera demandée à chaque fois que nous utiliserons notre clé privée et signera tous les certificats que nous émettrons. Ensuite, nous générons le certificat de type x509 auto-signé à l'aide de la clé précédemment construite. Le certificat a une durée de 365 jours ou un an. Étant donné que le certificat n'est pas signé par une autorité de certification de confiance, le navigateur affichera une anomalie comme celle vue ci-dessus. C'est notre certificat d'autorité de certification qui nous permettra de signer les certificats créés.

```
openssl genrsa -des3 -out ca.key 4096
openssl req -new -x509 -days 365 -key ca.key -out ca.crt
```

```
admin_web@certificates ~ $ sudo /etc/init.d/apache2 stop
Password:
Sorry, try again.
Password:
Sorry, try again.
Password:
* Stopping apache2 ... [ ok ]
admin_web@certificates ~ $
```

```
admin_web@certificates ~ $ openssl genrsa -des3 -out desjardins.key 4096
Generating RSA private key, 4096 bit long modulus
.....++
.....++
e is 65537 (0x10001)
Enter pass phrase for desjardins.key:
Verifying - Enter pass phrase for desjardins.key:
admin_web@certificates ~ $ openssl req -new -key desjardins.key -out desjardins.csr
Enter pass phrase for desjardins.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:ca
State or Province Name (full name) [Some-State]:quebec
Locality Name (eg, city) []:montreal
Organization Name (eg, company) [Internet Widgits Pty Ltd]:poly
Organizational Unit Name (eg, section) []:ing
Common Name (e.g. server FQDN or YOUR name) []:www.desjardins.com
Email Address []:jeanfikani@gmail.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
admin_web@certificates ~ $
```

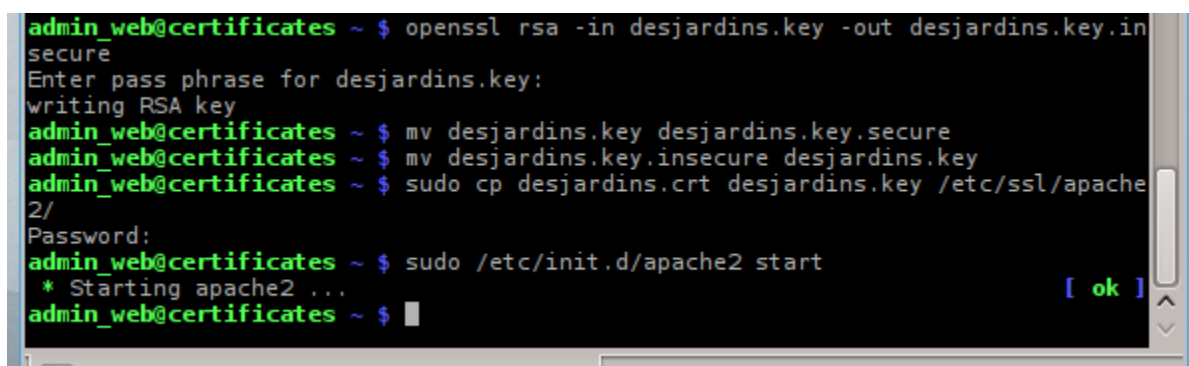
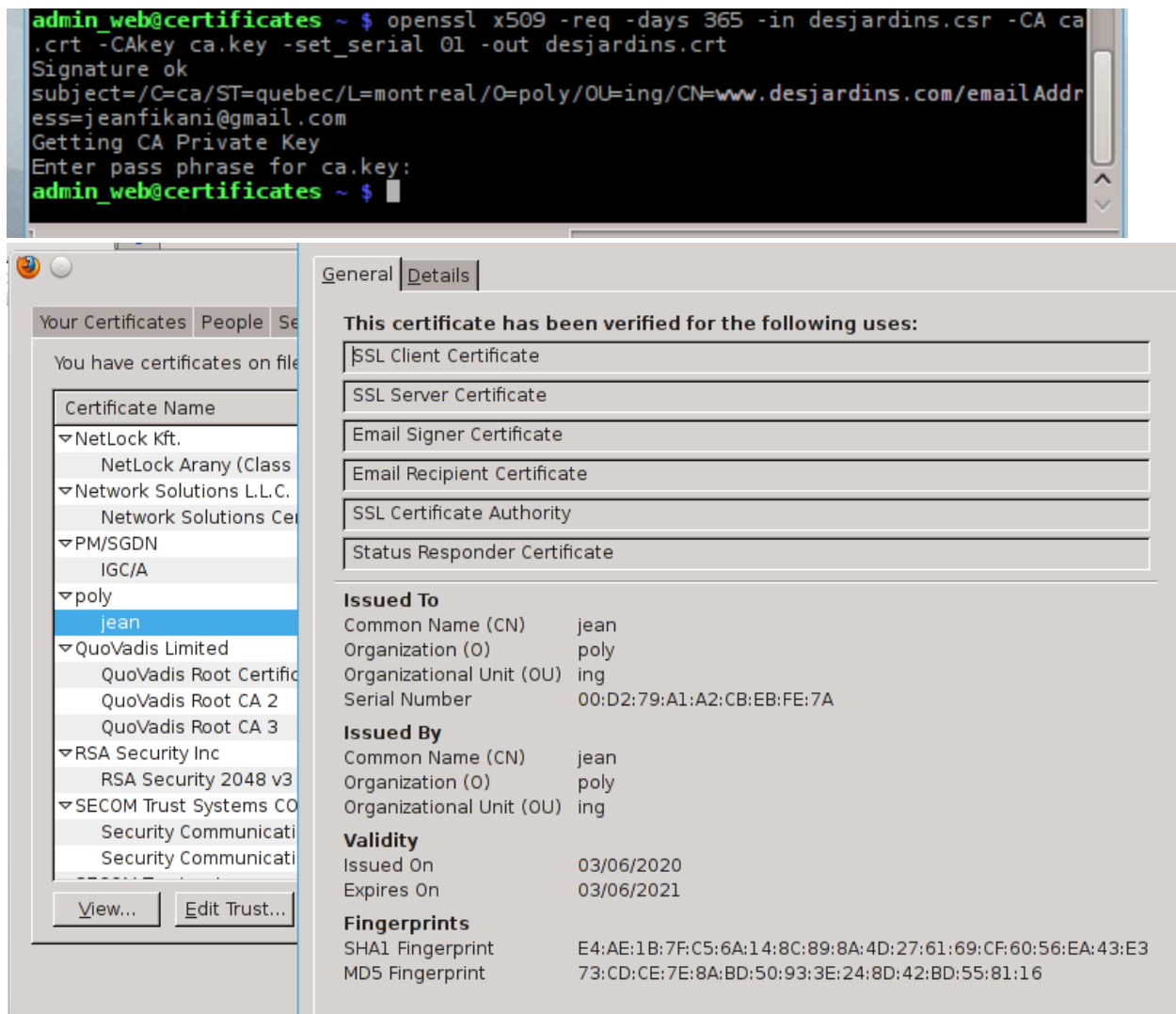
La signature du certificat serveur par le CA (Certificate Authority)

Signature de la requête :

openssl x509 -req -days 365 -in desjardins.csr -CA ca.crt -CAkey ca.key -set_serial 01 -out desjardins.crt

Certificat signé est le fichier "desjardins.crt".

Alors, sortie de la commande est la suivante :



Firefox affiche la page sans alerte. Car nous avons indiqué www.desjardins.com au champ (Common Name)

e)


Untrusted Connection

+

https://www.rbc.com

☆

Google



This Connection is Untrusted

You have asked Firefox to connect securely to **www.rbc.com**, but we can't confirm that your connection is secure.

Normally, when you try to connect securely, sites will present trusted identification to prove that you are going to the right place. However, this site's identity can't be verified.

What Should I Do?

If you usually connect to this site without problems, this error could mean that someone is trying to impersonate the site, and you shouldn't continue.

Get me out of here!

Technical Details

www.rbc.com uses an invalid security certificate.

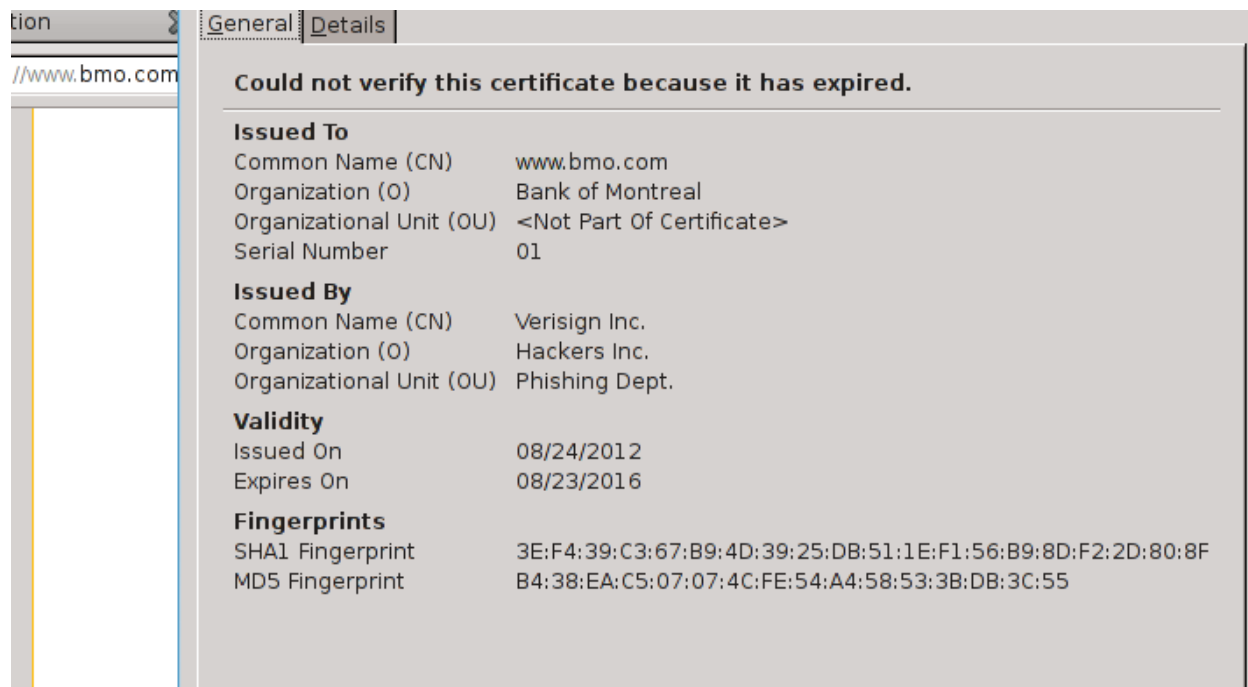
The certificate is not trusted because no issuer chain was provided.

(Error code: sec_error_unknown_issuer)

I Understand the Risks

If you understand what's going on, you can tell Firefox to start trusting this site's identification. **Even if you trust the site, this error could mean that someone is tampering with your connection.**

Don't add an exception unless you know there's a good reason why this site doesn't use



Alors, “This connection is Untrusted” qui apparaît Pour les deux sites. Firefox bloque la page et affiche une alerte pour les deux sites. En examinant le certificat plus en détail, nous apprenons des informations intéressantes telles que l'organisation (Hacker Inc.) et l'unité organisationnelle (Phishing Department). Bref, rien de bon augure.

g)

The screenshot shows a Firefox browser window with the BMO Financial Group website loaded. The address bar shows 'https://www.bmo.com'. The website has a blue header with the BMO logo and navigation links. A 'Certificate Manager' dialog box is open in the foreground, displaying a list of certificates. The 'Servers' tab is selected, showing a table of certificates for various servers.

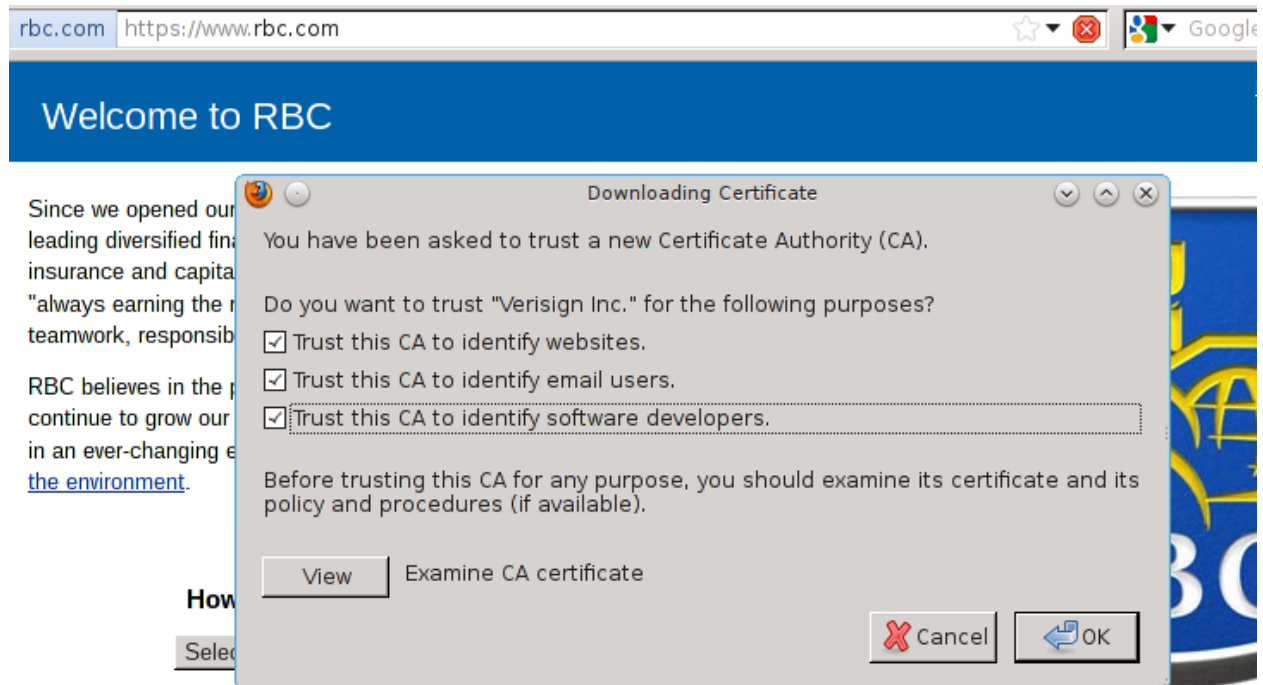
Certificate Name	Server	Lifetime	Expires On
▼ (Unknown)			
www.bmo.com	www.bmo.com:443	Temporary	08/23/2016
▼ DigiNotar			
DigiNotar Cyber CA	*	Permanent	10/04/2011
DigiNotar Cyber CA	*	Permanent	09/20/2013
DigiNotar Root CA	*	Permanent	03/31/2025
DigiNotar Services 1024 CA	*	Permanent	08/26/2013
▼ DigiNotar B.V.			
DigiNotar PKIoverheid CA ...	*	Permanent	03/23/2020
DigiNotar PKIoverheid CA ...	*	Permanent	07/27/2015

Étant donné que l'exception n'était que permanente, lors du nettoyage du cache de Firefox, les exceptions ont été supprimés

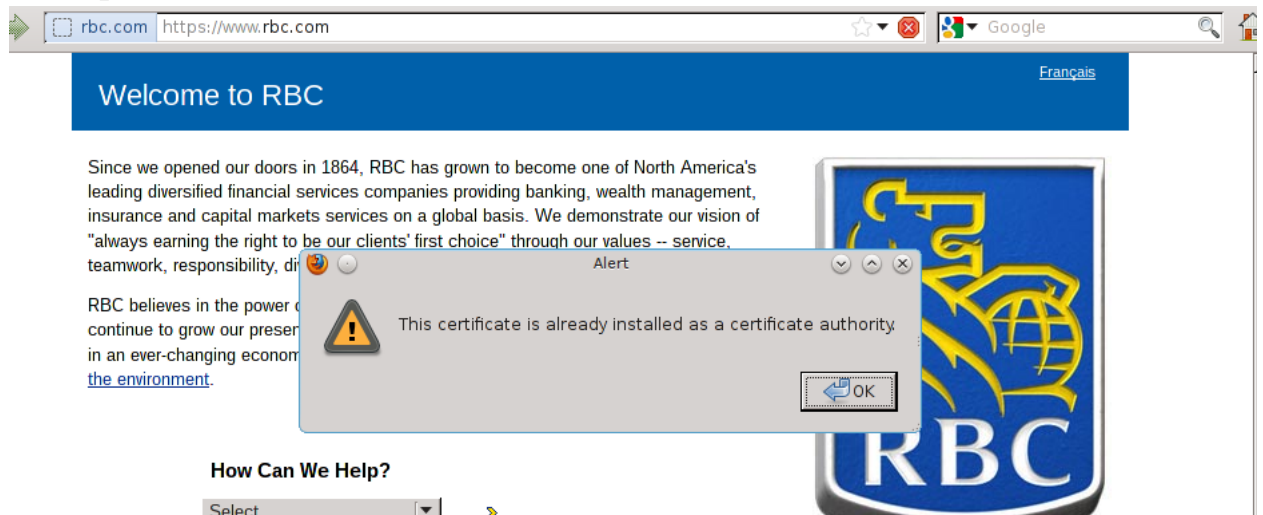
Après avoir effacé le cache, l'exception ajoutée ci-dessus est supprimée et n'apparaît plus dans la liste des exceptions dans la fenêtre Gestionnaire de certificats.

En fait, l'exception est temporaire et non permanente. Nous n'avons plus accès au site et Firefox affiche une alerte.

h)



Un message s'affichait qu'un certificat d'autorité est déjà installé.
On a toujours accès au site car Firefox trouvait un certificat d'autorité correspondant.



Since we opened our doors in 1864, RBC has grown to become one of North America's leading diversified financial services companies providing banking, wealth management, insurance and capital markets services on a global basis. We demonstrate our vision of "always earning the right to be our clients' first choice" through our values -- service, teamwork, responsibility, diversity and integrity.

RBC believes in the power of communities and the individuals who live in them. As we continue to grow our presence globally, we offer the right advice and solutions to our clients in an ever-changing economic environment and do our part to help in the [community and the environment](#).

How Can We Help?

Select...



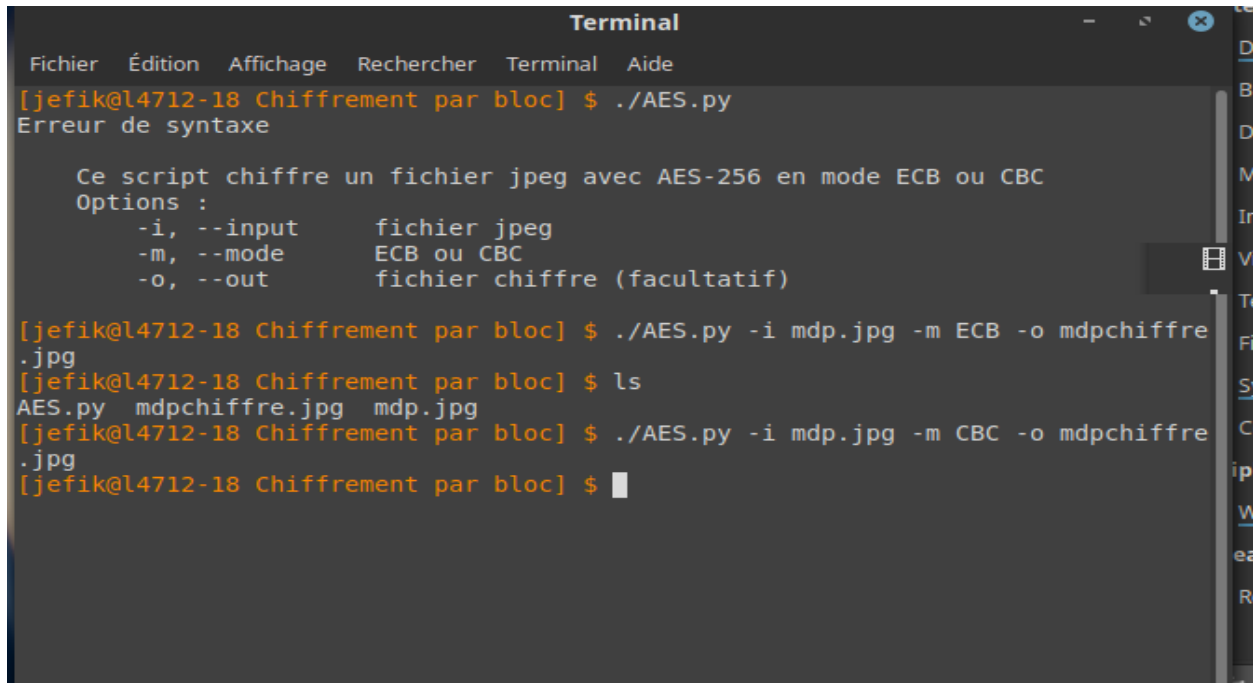
- i) Nous avons accès au site. Le certificat d'autorité BMO a dû être ajouté à Firefox au cours de la question précédente. Le site bmo.com maintient également un certificat publié par la même autorité que rbc.com, "Verisign inc". Ensuite, nous pouvons également y accéder, car nous avons pris la décision par le passé de faire confiance à cette autorité.
- j) Accepter un certificat auto-signé est dangereux car il n'y a aucun moyen de vérifier l'intégrité et la bonne foi du site. Ils sont généralement utilisés dans les entreprises et les employés doivent ignorer les avertissements. Du point de vue d'une éducation saine des utilisateurs, cette pratique est déplorable et conduit à de mauvaises habitudes. Les autorités de certification CA sont dangereuses car le navigateur peut également accepter automatiquement d'autres certificats avec la même autorité. Ils sont plus ou moins sur une liste blanche de confiance aveugle. Augmente la prolifération des attaques. Un certificat auto-signé est, comme son nom l'indique signé par l'entité du site, qui ce n'est pas très sûr, car l'entité du site peut être malveillante. Si nous acceptons un certificat d'autorité douteux, votre navigateur peut compter sur plusieurs sites douteux.

Question 3 - Chiffrement par bloc et modes d'opération

[/0.5]

- a) Nous pouvons clairement voir les formes des lettres qui composent le mot de passe, ainsi que les séparations entre les blocs.

Lancement du programme AES.py



```
Terminal
Fichier  Édition  Affichage  Rechercher  Terminal  Aide
[jefik@l4712-18 Chiffrement par bloc] $ ./AES.py
Erreur de syntaxe

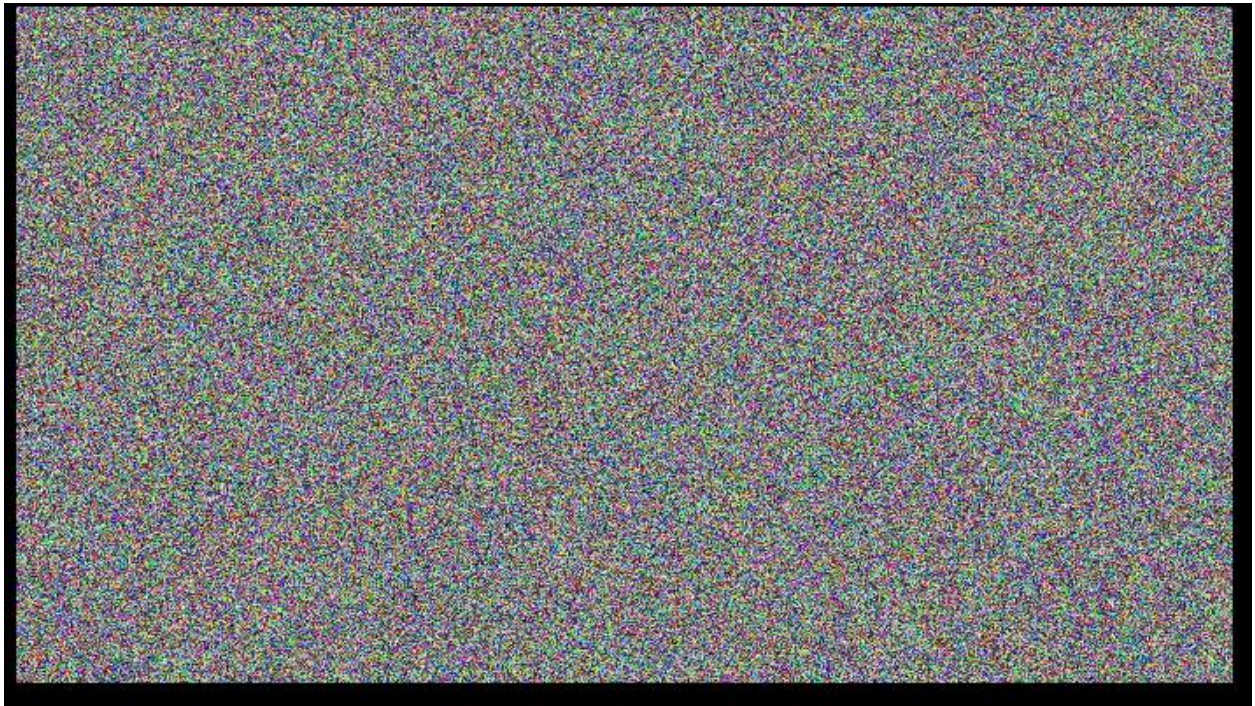
Ce script chiffre un fichier jpeg avec AES-256 en mode ECB ou CBC
Options :
  -i, --input      fichier jpeg
  -m, --mode       ECB ou CBC
  -o, --out        fichier chiffré (facultatif)

[jefik@l4712-18 Chiffrement par bloc] $ ./AES.py -i mdp.jpg -m ECB -o mdpchiffre.jpg
[jefik@l4712-18 Chiffrement par bloc] $ ls
AES.py  mdpchiffre.jpg  mdp.jpg
[jefik@l4712-18 Chiffrement par bloc] $ ./AES.py -i mdp.jpg -m CBC -o mdpchiffre.jpg
[jefik@l4712-18 Chiffrement par bloc] $
```


Chiffrement AES avec mode ECB



b) Chiffrement AES mode CBC



C'est à privilégier, le fichier généré ici présente un bruit aléatoire. Il ne permet aucune reconnaissance visuelle de l'image d'origine. Le mode CBC respecte bien mieux le principe d'incertitude et de chaos que le mode ECB et offre donc une plus grande sécurité.

c) Chiffrement AES avec mode ECB

Le principe du mode ECB est basé sur la découpe des données qui seront chiffrées dans un bloc de N bits, puis effectuer un "ou exclusif" ou XOR dans chaque bloc avec une clé de chiffrement de taille N bits:

$$C_i = K \oplus P_i$$

Où C représente le Chiffré (Cipher), K la clef et P le clair (Plain)

Le problème est que si plusieurs blocs contiennent les mêmes données, les blocs de sortie seront cryptés de la même manière. On comprend tout de suite que l'entropie calculée dans le fichier généré précédemment serait plus faible par rapport au fichier généré avec le mode CBC qui, visuellement, indique un meilleur respect du principe d'incertitude. Par conséquent, le mode ECB présente deux problèmes principaux:

1. Le premier est que deux blocs identiques, c'est-à-dire avec le même contenu, seront chiffrés de la même manière, ce qui facilitera les attaques des statistiques. Nous appelons cela la cryptanalyse scientifique. Ensuite, par essais et erreurs, nous pouvons extraire des informations précieuses, en particulier reconstruire des séquences claires du texte chiffré. Par conséquent, nous obtenons une Code dictionnaire avec les correspondances entre clair et crypté, d'où le terme de codebook.
2. La seconde est que le mode ECB ne respecte pas l'intégrité des données. Un attaquant peut remplacer complètement un bloc chiffré par d'autres blocs chiffrés du message, échangé deux blocs, sans que le destinataire ne s'en aperçoive. Il n'y a pas de "chaîne" entre les différents blocs et permet donc des attaques comme les suivantes: Imaginez une transaction bancaire d'un montant de 1999 euros. Un attaquant, amoureux de l'argent, changera le premier et le dernier chiffre: 9991 euros.

En utilisant l'utilitaire ./h-bit pour chacun des fichiers générés, nous observons que l'image générée avec le mode CBC a une meilleure distribution de 0 et 1 et, par conséquent, offre une meilleure entropie, comme on le suppose alors. L'utilitaire ./h-ascii confirme qu'il existe une plus grande incertitude pour l'image chiffrée avec le mode CBC.

Conclusion :

Bien qu'AES offre une sécurité reconnue, son mode doit être choisi avec soin. Et même si CBC présente également des défaillances, qui ne seront pas discutées ici, car cela dépasserait le cadre de ce laboratoire, il est préférable au mode ECB pour les raisons évidentes évoquées ci-dessus.

Question 4 - Organisation des mots de passe en UNIX/Linux [/1]

a)

```
Mdp login: root
Password:
Last login: Mon Mar  2 11:21:59 EST 2020 on tty1
Mdp ~ # cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/bin/false
daemon:x:2:2:daemon:/sbin:/bin/false
adm:x:3:4:adm:/var/adm:/bin/false
lp:x:4:7:lp:/var/spool/lpd:/bin/false
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
news:x:9:13:news:/var/spool/news:/bin/false
uucp:x:10:14:uucp:/var/spool/uucp:/bin/false
operator:x:11:0:operator:/root:/bin/bash
portage:x:250:250:portage:/var/tmp/portage:/bin/false
nobody:x:65534:65534:nobody:/var/empty:/bin/false
man:x:13:15:added by portage for man:/usr/share/man:/sbin/nologin
sshd:x:22:22:added by portage for openssh:/var/empty:/sbin/nologin
Mdp ~ #
```

Les permissions d'accès du fichier /etc/passwd sont :

- rw- : indique que le propriétaire du fichier, root en l'occurrence, peut lire et modifier (et donc supprimer) le fichier. En revanche, il ne peut pas l'exécuter car il n'a pas de x à la fin.
- r-- : tous les utilisateurs qui font partie du groupe root mais qui ne sont pas root peuvent seulement lire le fichier. Ils ne peuvent ni le modifier, ni l'exécuter.
- r-- : tous les autres utilisateurs peuvent seulement lire le fichier

Le fichier ne contient pas de mots de passe car il est historiquement lisible par tous. Seul le fichier /etc/shadow contient des mots de passe et n'est accessible que par root pour des raisons de sécurité évidentes. Le champ qui contenait autrefois un

mot de passe dans /etc/passwd contient désormais le caractère "x", comme on peut le voir sur l'image ci-dessus.

En Linux, toutes les données de connexion des utilisateurs sont stockées dans le fichier

/etc/passwd. Il s'agit d'un simple fichier texte utilisé lors de la connexion à un utilisateur. Informations utilisateur telles que nom d'utilisateur, UID, GID, répertoire personnel, shell, etc. Ils sont stockés dans le fichier /etc/passwd. Tous les utilisateurs peuvent lire ce fichier, mais seul l'utilisateur root peut le modifier.

Les autorisations d'accès du fichier /etc/passwd sont:

- rw-: indique que le propriétaire du fichier, root dans ce cas, peut lire et modifier (et donc supprimer) le fichier. Cependant, vous ne pouvez pas l'exécuter car il n'a pas x à la fin.
- r--: tous les utilisateurs qui font partie du groupe root mais qui ne sont pas root peuvent uniquement lire le fichier. Ils ne peuvent ni le modifier ni l'exécuter.
- r--: tous les autres utilisateurs ne peuvent lire que le fichier

Le fichier ne contient pas de mots de passe car historiquement tout le monde peut le lire. Seul le fichier /etc/shadow contient des mots de passe et n'est accessible que par root pour des raisons de sécurité évidentes. Le champ qui contenait autrefois un mot de passe dans /etc/passwd contient maintenant le caractère "x", comme on peut le voir dans l'image ci-dessous.

b)



```
Mdp ~ # cat /etc/shadow
root:$6$ng.JEjcm$7S11KCSLXahz.Am1u6kug1F.j4U13jdg1CDNBIRIi0jr.jmmu9UFQLasRjIqGYp0P7KbgLWMxi i .8XycKotUWMM0:15580:0:0:
halt:!:9797:0:0:
operator:!:9797:0:0:
shutdown:!:9797:0:0:
sync:!:9797:0:0:
bin:!:9797:0:0:
daemon:!:9797:0:0:
adm:!:9797:0:0:
lp:!:9797:0:0:
news:!:9797:0:0:
uucp:!:9797:0:0:
nobody:!:9797:0:0:
man:!:15513:0:0:
sshd:!:15513:0:0:
Mdp ~ #
```

Contenu du fichier /etc/shadow avant ajouter un nouvel utilisateur

```
Mdp ~ # useradd -g users -s/bin/bash -m bob
Mdp ~ # cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/bin/false
daemon:x:2:2:daemon:/sbin:/bin/false
adm:x:3:4:adm:/var/adm:/bin/false
lp:x:4:7:lp:/var/spool/lpd:/bin/false
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
news:x:9:13:news:/var/spool/news:/bin/false
uucp:x:10:14:uucp:/var/spool/uucp:/bin/false
operator:x:11:0:operator:/root:/bin/bash
portage:x:250:250:portage:/var/tmp/portage:/bin/false
nobody:x:65534:65534:nobody:/var/empty:/bin/false
man:x:13:15:added by portage for man:/usr/share/man:/sbin/nologin
sshd:x:22:22:added by portage for openssh:/var/empty:/sbin/nologin
bob:x:1000:100::/home/bob:/bin/bash
Mdp ~ #
```

```
Mdp ~ # passwd bob
New password:
BAD PASSWORD: it is too short
BAD PASSWORD: is too simple
Retype new password:
passwd: password updated successfully
Mdp ~ # cat /etc/shadow
root:$6$mq.JEjcm$7SI1KCSLXahz.Am1u6kug1Fj4V13jdg1CDNBIRIi0jrjmmv9UFQLasRj1q6Yp0P7KbgLWMxi.8XycKotVWMM0:15580:0:0:0:
halt:!:9797:0:0:0:
operator:!:9797:0:0:0:
shutdown:!:9797:0:0:0:
sync:!:9797:0:0:0:
bin:!:9797:0:0:0:
daemon:!:9797:0:0:0:
adm:!:9797:0:0:0:
lp:!:9797:0:0:0:
news:!:9797:0:0:0:
uucp:!:9797:0:0:0:
nobody:!:9797:0:0:0:
man:!:15513:0:0:0:
sshd:!:15513:0:0:0:
bob:$6$z00vMuaS$xC1WTeZTsr1WZEmapi4F7zzU20xW/vUS6cX16rEzAzBXfCSzt1M2T2Gp81gRdso33Zd.jxju4HmSOqyDLopZW6.:18326:0:99999:7:0:
Mdp ~ #
```

```

Mdp ~ # useradd -g users -s/bin/bash -m alex
Mdp ~ # cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/bin/false
daemon:x:2:2:daemon:/sbin:/bin/false
adm:x:3:4:adm:/var/adm:/bin/false
lp:x:4:7:lp:/var/spool/lpd:/bin/false
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
news:x:9:13:news:/var/spool/news:/bin/false
uucp:x:10:14:uucp:/var/spool/uucp:/bin/false
operator:x:11:0:operator:/root:/bin/bash
portage:x:250:250:portage:/var/tmp/portage:/bin/false
nobody:x:65534:65534:nobody:/var/empty:/bin/false
man:x:13:15:added by portage for man:/usr/share/man:/sbin/nologin
sshd:x:22:22:added by portage for openssh:/var/empty:/sbin/nologin
bob:x:1000:100::/home/bob:/bin/bash
alex:x:1001:100::/home/alex:/bin/bash
Mdp ~ # cat /etc/shadow
root:$6$ng.JE.jcn$7SI1KCSLXahz.Am1u6kug1Fj4U13jdg1CDNBIRIi0jr.jmmv9UFQLasRjIqGYp0P7KbgLWMxi.i.8XycKotVWMM0:15580:0:0:0:
halt:!:9797:0:0:0:
operator:!:9797:0:0:0:
shutdown:!:9797:0:0:0:
sync:!:9797:0:0:0:
bin:!:9797:0:0:0:
daemon:!:9797:0:0:0:
adm:!:9797:0:0:0:
lp:!:9797:0:0:0:
news:!:9797:0:0:0:
uucp:!:9797:0:0:0:
nobody:!:9797:0:0:0:
man:!:15513:0:0:0:
sshd:!:15513:0:0:0:
bob:$6$z00vMuaS$xCIWTeZTsrIWZEnap14F7zzU20xW/uUS6cX16rEzAzBXfCSzt1M2T2Gp81gRdso33Zd.jx.ju4HmSOgyDLopZW6.:18326:0:99999:7:0:
alex:!:18326:0:99999:7:0:
Mdp ~ #

```

Contenu des fichiers passwd et shadow après la création d'utilisateur BOB. Une ligne a été ajoutée au contenu du fichier passwd après la création de l'utilisateur BOB avec son mot de passe.

Cependant, qu'aucun mot de passe n'a été associé à ce compte dans le fichier shadow qu'on a aussi créé qui s'appelle ALEX, quel qu'il soit, BOB s'est vu attribuer un mot de passe, comme on peut le voir très clairement dans le contenu du fichier Shadow. Celui-ci est également haché.

- c) Notez que seul le fichier shadow est modifié. En fait, il est très clair que le mot de passe de hachage a changé. Ce qui semble logique puisque nous ne modifions le mot de passe que manuellement. Nous déduisons et confirmons que les mots de passe sont gérés par le fichier shadow, et non par le fichier passwd comme son nom pourrait le suggérer.

```
bob:$6$z00vMuaS$xClWTeZTsrlWZEapi4F7zzU20xW/vUS6cX16rEzAzBXfCSzt1M2T2Gp8lgRdso332d.jx.ju4HmS0qyDLopZW6.:18326:0:99999:7:::
Mdp ~ #

Mdp ~ # ls -l /etc/shadow
-rw-r----- 1 root root 493 Mar  5 10:14 /etc/shadow
Mdp ~ #
```

Comme indiqué dans l'image ci-dessus, seul l'utilisateur root peut lire et modifier le fichier Shadow. Le groupe root ne peut que lire. Tous les autres utilisateurs n'ont pas de droits sur ce fichier. Ces autorisations restrictives sont essentielles pour la gestion des données confidentielles, telles que les mots de passe.

- d) Le mot de passe crypté diffère du message crypté d'antan. Pour comprendre pourquoi les informations de mot de passe ont changé, vous devez comprendre comment elles sont générées. Les mots de passe ne sont pas chiffrés, mais hachés. Ces mots de passe sont stockés au format suivant: \$ID\$SALT\$HASH.

L'ID indique la fonction de hachage utilisée, HASH le hachage du mot de passe. Enfin, SALT ou SEL ajouté de la complexité d'entropie et est utilisé pour générer des mots de passe.

- ✓ Type de Hachage est SHA-512
- ✓ ID est : \$6\$
- ✓ Taille du Hash est : 86 caractères

Alors, Le mot de passe généré diffère également à cause du sel qui est généré aléatoirement et change à chaque nouvelle utilisation.

e)

```

Mdp ~ # useradd -g users -s/bin/bash -m alex
Mdp ~ # cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/bin/false
daemon:x:2:2:daemon:/sbin:/bin/false
adm:x:3:4:adm:/var/adm:/bin/false
lp:x:4:7:lp:/var/spool/lpd:/bin/false
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
news:x:9:13:news:/var/spool/news:/bin/false
uucp:x:10:14:uucp:/var/spool/uucp:/bin/false
operator:x:11:0:operator:/root:/bin/bash
portage:x:250:250:portage:/var/tmp/portage:/bin/false
nobody:x:65534:65534:nobody:/var/empty:/bin/false
man:x:13:15:added by portage for man:/usr/share/man:/sbin/nologin
sshd:x:22:22:added by portage for openssh:/var/empty:/sbin/nologin
bob:x:1000:100::/home/bob:/bin/bash
alex:x:1001:100::/home/alex:/bin/bash
Mdp ~ # cat /etc/shadow
root:$6$ng.JE.jcn$7SI1KCSLXahz.Am1u6kug1Fj4U13jdg1CDNBIRIi0.jr.jmmv9UFQLasRjIqGYp0P7KbgLWMxii.8XycKotUWMM0:15580:0:::::
halt:!:9797:0:::::
operator:!:9797:0:::::
shutdown:!:9797:0:::::
sync:!:9797:0:::::
bin:!:9797:0:::::
daemon:!:9797:0:::::
adm:!:9797:0:::::
lp:!:9797:0:::::
news:!:9797:0:::::
uucp:!:9797:0:::::
nobody:!:9797:0:::::
man:!:15513:0:::::
sshd:!:15513:0:::::
bob:$6$z00vMuaS$xCIWTeZTsrIWZEmap14F7zzU20xW/uUS6cX16rEzAzBXfCSzt1M2T2Gp8lgRdso33Zd.jx.ju4HmS0qyDLop2W6.:18326:0:99999:7:::
alex:$6$z00vMuaS$xCIWTeZTsrIWZEmap14F7zzU20xW/uUS6cX16rEzAzBXfCSzt1M2T2Gp8lgRdso33Zd.jx.ju4HmS0qyDLop2W6.:18326:0:99999:7:::
Mdp ~ #

Mdp ~ # cat /etc/shadow
root:$6$ng.JE.jcn$7SI1KCSLXahz.Am1u6kug1Fj4U13jdg1CDNBIRIi0.jr.jmmv9UFQLasRjIqGYp0P7KbgLWMxii.8XycKotUWMM0:15580:0:::::
halt:!:9797:0:::::
operator:!:9797:0:::::
shutdown:!:9797:0:::::
sync:!:9797:0:::::
bin:!:9797:0:::::
daemon:!:9797:0:::::
adm:!:9797:0:::::
lp:!:9797:0:::::
news:!:9797:0:::::
uucp:!:9797:0:::::
nobody:!:9797:0:::::
man:!:15513:0:::::
sshd:!:15513:0:::::
bob:$6$z00vMuaS$xCIWTeZTsrIWZEmap14F7zzU20xW/uUS6cX16rEzAzBXfCSzt1M2T2Gp8lgRdso33Zd.jx.ju4HmS0qyDLop2W6.:18326:0:99999:7:::
alex:$6$z00vMuaS$xCIWTeZTsrIWZEmap14F7zzU20xW/uUS6cX16rEzAzBXfCSzt1M2T2Gp8lgRdso33Zd.jx.ju4HmS0qyDLop2W6.:18326:0:99999:7:::
Mdp ~ #

```

Il est très possible de se connecter avec le compte du deuxième usager en utilisant le hachage du mot de passe du premier BOB. Ce que nous venons de réaliser, c'est absolument interdit car il est clair que deux ou plusieurs usagers partagent le même mot de passe. Si l'un des deux comptes est compromis, l'autre le sera aussi. Avec un SALT séparé, ce qui est normalement le cas, il serait impossible de dire que deux comptes ont le même mot de passe car la probabilité que deux hachages soient identiques est extrêmement faible, voire presque nulle.

f)


```

Mdp ~ # userdel -r alex
Mdp ~ # cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/bin/false
daemon:x:2:2:daemon:/sbin:/bin/false
adm:x:3:4:adm:/var/adm:/bin/false
lp:x:4:7:lp:/var/spool/lpd:/bin/false
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
news:x:9:13:news:/var/spool/news:/bin/false
uucp:x:10:14:uucp:/var/spool/uucp:/bin/false
operator:x:11:0:operator:/root:/bin/bash
portage:x:250:250:portage:/var/tmp/portage:/bin/false
nobody:x:65534:65534:nobody:/var/empty:/bin/false
man:x:13:15:added by portage for man:/usr/share/man:/sbin/nologin
sshd:x:22:22:added by portage for openssh:/var/empty:/sbin/nologin
bob:x:1000:100:/home/bob:/bin/bash
Mdp ~ # cat /etc/shadow
root:$6$ng.JE.jcm$7SI1KCSLXahz.Am1u6kug1F.j4UI3.jdg1CDNBIRIi0.jr.jmmv9UUFQLasR.jIqGYp0P7KbgLWMxii.8XycKotVUWMM0:15580:0:0:0:
halt:!:9797:0:0:0:0:
operator:!:9797:0:0:0:0:
shutdown:!:9797:0:0:0:0:
sync:!:9797:0:0:0:0:
bin:!:9797:0:0:0:0:
daemon:!:9797:0:0:0:0:
adm:!:9797:0:0:0:0:
lp:!:9797:0:0:0:0:
news:!:9797:0:0:0:0:
uucp:!:9797:0:0:0:0:
nobody:!:9797:0:0:0:0:
man:!:15513:0:0:0:0:
sshd:!:15513:0:0:0:0:
bob:$6$z00vMuaS$xC1WTeZTsrIWZEmapI4F7zzUz0xW/vUS6cX16rEzAzBXfCSzt1M2T2Gp81gRdso33Zd.jx.ju4HmS0qyDLopZW6.:18326:0:99999:7:0:
Mdp ~ #

```

Les entrées dans les deux fichiers - shadow et passwd - ont été effacées. Ce qui est tout normal suite à la suppression de l'utilisateur Alex. Alors, il ne reste plus que BOB.

Question 5 - Contrôle de qualité de choix de mot de passe

[/1]

a)

```
Mdp john # john /root/password1 /root/password2
Loaded 14 password hashes with 12 different salts (FreeBSD MD5 [32/64 X2])
0244fni      (inf4420)
0244fni      (inf4420)
john1       (john)
claudia     (david)
security    (admin)
nientel     (lola)
guesses: 6   time: 0:00:00:35 52.37% (2) (ETA: Thu Mar  5 10:51:18 2020) c/s: 16684 trying: testeR - testI
Tigers5     (andre)
3sunshine   (morning)
guesses: 8   time: 0:00:02:25 (3)   c/s: 17317 trying: chipit - chipia
guesses: 8   time: 0:00:04:14 (3)   c/s: 17384 trying: 0871941 - 0871949
guesses: 8   time: 0:00:04:15 (3)   c/s: 17382 trying: 0843078 - 0843074
guesses: 8   time: 0:00:04:16 (3)   c/s: 17383 trying: sping22 - sping23
guesses: 8   time: 0:00:04:17 (3)   c/s: 17384 trying: spracle - spracla
guesses: 8   time: 0:00:04:18 (3)   c/s: 17385 trying: mcgrick - mcgrica
guesses: 8   time: 0:00:04:19 (3)   c/s: 17385 trying: mcbuaid - mcbuair
guesses: 8   time: 0:00:04:20 (3)   c/s: 17386 trying: mc28630 - mc28633
guesses: 8   time: 0:00:04:21 (3)   c/s: 17388 trying: cupitcs - cupitct
guesses: 8   time: 0:00:04:24 (3)   c/s: 17390 trying: buttage - buttago
guesses: 8   time: 0:00:04:26 (3)   c/s: 17390 trying: jery - jerd
Session aborted
```

```
Mdp john # cat john.pot
$1$Wila6SGN$LPLfCWuikEZkOb7CPT01p.:0244fni
$1$n/P09Tgu$CAs0ZntIFmZk3tAfr2Y2B0:john1
$1$Aw/cHolc$laW8KVkQeJAernWE1TL3B/:claudia
$1$arMaK13M$PMYZT2poiPR4pdGW26rlw0:security
$1$S2uBDM/D$C8dXktTJAjxUndXThMboX/:nientel
$1$fV99GiZo$Uay3oILYbUvYsdiahaBMf1:Tigers5
$1$hLGAa7.R$FbMLS3T/XJIrSkUcWnHv.1:3sunshine
Mdp john #
```

Lors de la première phase de test, 8 mots de passe ont été trouvés. Notez, que le compte inf4420 a le même hachage, par conséquent, le même mot de passe dans les deux fichiers password1 et password2

- b) Soit un alphabet X de N lettres, l'entropie de cet alphabet, noté $H(X)$, est l'opposé de la somme du produit des probabilités multiplié par son logarithme en base2. L'entropie est maximale lorsque la probabilité de l'apparence de chaque élément est équiprobable.

Un alphabet X de N lettres, l'entropie de cet alphabet, noté H(X), est l'opposé de la somme du produit des probabilités multipliées par leur logarithme en base 2. L'entropie est maximale quand la probabilité d'apparition de chaque élément est équiprobable.

$$H(X) = - \sum_{n=1}^N p(x) * \log_2 p(x) = -\log_2(p) = \log_2(N) = \log_2(52) = 5.70$$

On vérifie $2^{5.70} = 52$

$H(x) = \log_2(62) = 5.95$ On vérifie : $2^{5.95} = 62$

$H(x) = \log_2(256) = 8$ On vérifie : $2^8 = 256$ pour une table ASCII étendu.

- c) Un mot de passe sera d'autant plus sécurisé que le plus de caractères dans l'alphabet utilisé. En supposant que les caractères ont une fréquence d'occurrence équiprobable. Plus d'options dans l'alphabet conduisent à une plus grande incertitude sur les caractères de sortie.
- d) Alors, les critères sont :
- Les mots de passe ne doivent pas être des mots du dictionnaire,
 - Les mots de passe ne doivent pas être sémantiquement proches de la connexion. **Ex: john -> john1 ou inf4420 -> 0244fni,**
 - Les mots de passe ne doivent pas utiliser un alphabet tel que *leetSpeak*, qui consiste à faire correspondre les caractères de l'alphabet habituel avec l'alphabet *leetSpeak*. **Ex: Soleil -> 50l31l.** Nous augmentons un peu l'entropie, mais un outil bien configuré comme *John TheRipper* surmontera ce type de mot de passe,
 - Les mots de passe longs sont préférés comme mot de passe,
 - N'utilisez jamais votre nom, votre date de naissance ou tout ce qui peut être directement ou indirectement lié à vous-même et, par conséquent, facile à deviner,
 - Le mot de passe à 28 bits d'entropie, donc 2^{28} combinaisons. Alors que le mot de passe: à 44 bits d'entropie, puis 2^{44} combinaisons. Le premier est sophistiqué, donc ce mot de passe est facile à déchiffrer par rapport au second

Partie C

Question 1 - Échec du protocole RSA [/0.75]

- a) Il s'agit d'appliquer une attaque sur RSA, communément appelé textbook RSA ou bien Raw RSA, où le chiffrement est appliqué directement au message. Cela rend le chiffrement déterministe et permet à un attaquant, étant donné un texte chiffré, de rechercher le texte clair correspondant et d'établir une table de correspondance. En général, pour chiffrer correctement un message à l'aide de RSA, vous devez d'abord appliquer un schéma de remplissage (OAEP/ PKCS) qui vous permet d'ajouter un caractère aléatoire au schéma déterministe qui est intrinsèquement RSA. Sans cela, RSA n'est pas sûr sécurisé, on va voir de nombreuses attaques en détail dans la question suivante.

b)

Matricule = 1847428 ; $e = 461$; $n = 198931$; Texte Chiffré = { 140147, 94012, 33456, 126824, 16753 }

La liste ci-dessous est obtenue en calculant le chiffrement RSA de chaque caractère, avec $A = 0$, $B = 1 \dots$

La formule utilisée est $C = M^e \bmod n$ avec C la valeur affectée au caractère chiffré en RSA

M la valeur affectée au caractère en claire et toujours $e = 461$, $n = 198931$

<u>#</u>	<u>Lettre</u>	<u>Chiffrement RSA</u>
0	A	0
1	B	1
2	C	2048
3	D	184960
4	E	16753
5	F	65629
6	G	33456
7	H	64941
8	I	94012
9	J	37530
10	K	129767
11	L	140147
12	M	85624
13	N	126824
14	O	113260
15	P	169151

16	Q	170299
17	R	121827
18	S	74074
19	T	151284
20	U	189931
21	V	33580
22	W	162554
23	X	18939
24	Y	99741
25	Z	110560

En comparant chaque symbole du texte chiffré avec la liste alors le texte déchiffré de matricule mentionnée ci-dessus 1847428 est : “ **LIGNE** ”.

- c) La fonction de cryptage code les lettres A et B pour 0 et 1, car les valeurs non cryptées de A et B sont simplement 0 et 1. c'est la raison pour laquelle pour chiffrer correctement un message à l'aide de RSA, vous devez d'abord appliquer un schéma de remplissage (OAEP/PKCS) qui vous permet d'ajouter un caractère aléatoire au schéma déterministe qui est intrinsèquement RSA. Sans cela, RSA n'est pas sécurisé et autorise de nombreuses attaques.

Question 2 - Déchiffrement "simple" [/0.75]

Le chiffrement utilisé ici est un chiffrement de type substitution mono-alphabétique qui utilise un alphabet désordonné, ce qui rend le déchiffrement plus complexe. En fait, l'alphabet a 26 caractères, nous en avons $26!$ combinaisons. Par conséquent, cette clé a une entropie de $\log_2(26!) = \log_2(4.03 \cdot 10^{26}) = \log_2(4) + \log_2(10^{26}) = 2 + 26 \cdot \log_2(10) = 2 + 78 = 80$ bits. Le résultat exact est de 88 bits. Il s'agit d'une taille de clé relativement importante. Le caractère @ (espace) est intentionnellement omis.

Donc, Algorithme de substitution utilisé :

Source

- Texte en caractères latin

Codage

- Aucun

Chiffrement

- $x \rightarrow \pi(x)$

Clé

- π (une table de substitution)

L'énoncé nous dit que le texte clair est en anglais, mais pour nous en assurer nous avons calculé l'indice de coïncidence du texte chiffré.

L'indice de coïncidence correspond à l'IC moyen de la langue anglaise (0.0667).

On confirme donc que le texte est en anglais et nous pouvons commencer notre cryptanalyse.

- L'analyse fréquentielle des lettres.
- L'analyse fréquentielle des digrammes.
- L'analyse fréquentielle des trigrammes.

La déclaration nous dit que le texte clair est en anglais, mais pour nous assurer que nous avons calculé la correspondance du texte chiffré.

L'indice de coïncidence correspond à l'IC moyen en anglais (0,0667). Ensuite, nous confirmons que le texte est en anglais et nous pouvons commencer notre cryptanalyse.

1. Analyse de fréquence des lettres
2. Analyse de fréquence des Digrammes
3. Analyse de fréquence de trigramme

Lorsqu'un texte est relativement court comme celui-ci, il est risqué de se concentrer uniquement sur les mots de grande taille et/ou sur les fréquences individuelles des lettres car la correspondance avec les lettres claires et les lettres du texte chiffré peut être extrêmement hasardeuse. Nous nous sommes donc concentrés sur de petits mots - les mots de deux ou trois lettres - car plus facile à cryptanalyser. L'entropie d'un digramme ou trigramme est effectivement plus basse que l'entropie d'une lettre seule car nous effectuons le calcul suivant : $H(S)/n\text{-gramme}$. On en déduit que plus n est grand, plus on se rapproche de l'entropie du langage. Enfin, une fois ces mots déchiffrés, nous pouvons plus facilement attaquer les mots de plus grande taille. Le reste est trivial et machinale.

Lorsqu'un texte est relativement court comme celui-ci, il est risqué de se concentrer uniquement sur les gros mots et/ou les fréquences des lettres individuelles, car la correspondance avec des lettres claires et des lettres dans du texte crypté peut être extrêmement risquée. Par conséquent, nous nous concentrons sur les petits mots, les mots de deux ou trois lettres, car ils sont plus faciles à cryptanalyser. L'entropie d'un digramme ou d'un trigramme est en fait inférieur à l'entropie d'une seule lettre car nous effectuons le calcul suivant: $H(S)/n\text{-gramme}$. On en déduit que plus n est élevé, plus on se rapproche de l'entropie du langage. Enfin, une fois ces mots déchiffrés, nous pouvons plus facilement attaquer des mots plus gros.

L'alphabet utilisé est le suivant :

XYZABCDEFGHJKLMNOPQRSTVW

ABCDEFGHIJKLMNOPQRSTUVWXYZ

@ -- Q

__ FG

Alors, selon plusieurs scénarios qu'on a fait et simulé d'après les outils qui se trouvent dans UTILITAIRES TP1, on pourrait conclure que la synthèse précédente et ce qu'on avait mentionné ci-dessus.

Donc, comment les alphabets sont convertis et déchiffrés vers quelle substitution et en quel pourcentage pour chaque lettre, comme on a trouvé que plusieurs occurrences du mot CAE apparaissent ce qui nous fait immédiatement penser au trigramme de THE, ainsi que DF pour un digramme nt et l'Espace ou bien @ par un Q.

Texte chiffré:

1847428;

SAXYKHJKAIXPFHJKAJDREXHNJVYYXHRIXBARKXSDOYA@CXYKVF
K@UXKVXSDYXWVJTXXXSAXSDKXKSAX@HRIDREXVRXKSAXVKSA
JXYDIAXRAHK@UXCVJXKSDYXWHYXHXCHOD@DHJXBDKXVCXWVJ
TXKVXSDOXXXSAXWHYXOVQAIXKVXNHMKFJAXKSAXYTDCCXHJE
FDREXX

En découle le texte déchiffré :

**The only person who succeeds is the person who is progressively
realizing a worthy ideal**

Sources :

Q1 :

- https://fr.wikipedia.org/wiki/Entropie_de_Shannon
- <http://benhur.telug.ca/SPIP/inf6460/spip.php?article110>
- <http://nomis80.org/cryptographie/node23.html>

Q2 :

- <http://math.pc.vh.free.fr/divers/crypto/cryptanalyse.htm>
- http://www.bibmath.net/crypto/index.php?action=affiche&quoi=chasseur/frequences_english

Q3 :

- https://fr.wikipedia.org/wiki/Sécurité_inconditionnelle
-
- <https://www.shellhacks.com/linux-generate-password-hash/>
 - <https://unix.stackexchange.com/questions/209231/does-the-shadow-file-have-encrypted-passwords>
 - <https://unix.stackexchange.com/questions/219933/same-salt-hash-value-in-etc-shadow>
 - <https://www.tutorialsandyou.com/linux/linux-passwd-file-5.html>
 - <https://crypto.stackexchange.com/questions/41170/what-advantage-is-there-for-using-a-nonce-and-a-timestamp>