



INF4420A - Sécurité informatique

Automne 2018

TP No. [1]

Groupe [2]

[1972528] – [Manuel Rodriguez]

[1748125] – [David Tremblay]

Soumis à : Mikaela Stéphanie Ngamboe Mvogo

[15-10-2018]

Partie A

Question 1 - Entropie

a) L'entropie de l'entrée est **4.028952**.

```
[marod1@l4712-15 Source - Entropie - Chiffrement] $ ./texte 200 > text.bin
[marod1@l4712-15 Source - Entropie - Chiffrement] $ ./h-lettre < text.bin
(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
```

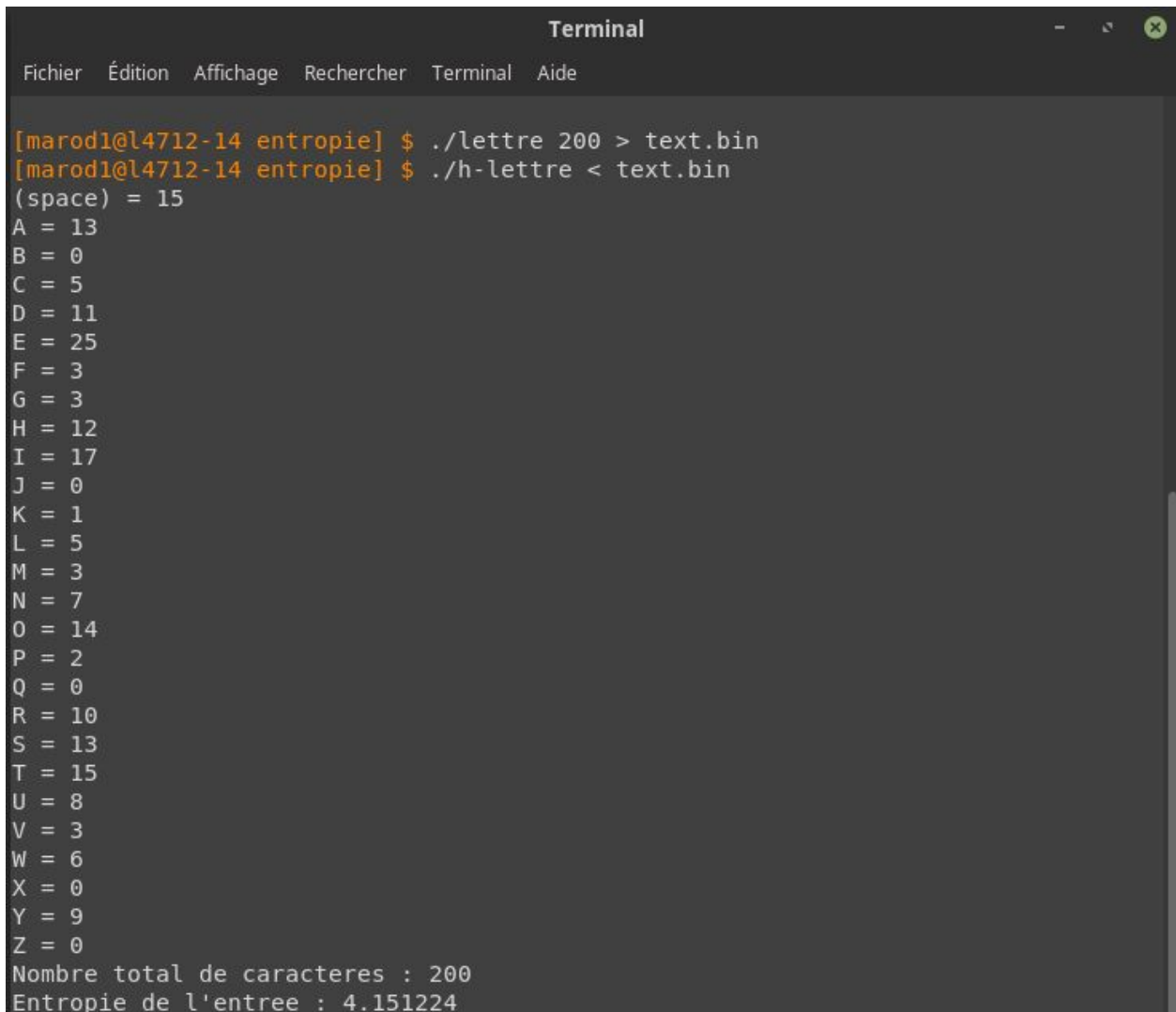
Figure 1: Question 1a)

b) L'entropie mesure l'incertitude de la chaîne de texte et pour le cas de h-lettre, une entropie de 4 veut dire que il faut 4 bits pour transmettre le text d'entrée sans perdre d'information.

c) L'entropie serait maximale parce que toutes les lettres ont la même probabilité d'apparaître dans le texte donc on ne pourrait pas compresser l'information parce que tous les composants sont d'importance égale. Comme $H(S) = \sum p_i \log_2 \frac{1}{p_i}$ et $p_i = 1/27$, $H(S) = 1/27 \cdot \log_2 27 + 1/27 \cdot \log_2 27 + \dots + 1/27 \cdot \log_2 27$ 27 fois. Finalement, $27/27 = 1 \cdot \log_2 27 = 4.754 = H(S)$

d) Le quotient $4.028/4.754 = 0.847$ est la relation entre une entropie d'une chaîne de texte, dans ce cas 4.028 et l'entropie maximale, 4.754. Cela peut nous donner une idée de la quantité d'informations qu'une chaîne de texte contient, en tenant compte de l'entropie maximale.

e)

A terminal window titled "Terminal" with a menu bar containing "Fichier", "Édition", "Affichage", "Rechercher", "Terminal", and "Aide". The terminal shows the execution of two commands: `./lettre 200 > text.bin` and `./h-lettre < text.bin`. The output lists the frequency of each character from space to Z, followed by the total number of characters (200) and the calculated entropy (4.151224).

```
[marod1@l4712-14 entropie] $ ./lettre 200 > text.bin
[marod1@l4712-14 entropie] $ ./h-lettre < text.bin
(space) = 15
A = 13
B = 0
C = 5
D = 11
E = 25
F = 3
G = 3
H = 12
I = 17
J = 0
K = 1
L = 5
M = 3
N = 7
O = 14
P = 2
Q = 0
R = 10
S = 13
T = 15
U = 8
V = 3
W = 6
X = 0
Y = 9
Z = 0
Nombre total de caracteres : 200
Entropie de l'entree : 4.151224
```

Non, la différence d'entropie n'est pas significative parce que même si la source n'est pas anglaise, la fréquence de chaque lettre (et le space) est similaire.

f) Les deux entropies sont similaires car les fréquences (probabilités de chaque lettre et espace) sont les mêmes. Il n'est pas surprenant de voir ce résultat. Après tout, l'entropie n'est qu'un indicateur et le fait que deux langues ont une entropie similaire ne signifie pas que leurs mots seront similaires car l'emplacement et la longueur des mots jouent également un rôle vital. Ainsi, la seule chose que cela nous indique est que les deux sources d'information ont une redondance similaire.

Question 2 - Histogrammes

a)

Les figures ci-dessous illustrent nos expérimentations dans la console de notre ordinateur pour chiffrer et déchiffrer les sources demandées à l'aide de *cesar*.

```
[marod1@l4712-15 Source - Entropie - Chiffrement] $ ./texte 200 > texte.bin
[marod1@l4712-15 Source - Entropie - Chiffrement] $ ./cesar < texte.bin > text.bin
[marod1@l4712-15 Source - Entropie - Chiffrement] $ ./cesar-d < text.bin
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 THE VISARD FROM HIS FACE THAT SETT A GLAS[ma
pie - Chiffrement] $ cat texte.bin
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 THE VISARD FROM HIS FACE THAT SETT A GLAS[ma
pie - Chiffrement] $ cat text.bin
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 WUXHMK KDWK SXOG WKH YLVDUG IURP KLV IDFH WKDW VHWV D JODV[ma
pie - Chiffrement] $
```

Figure 2: Chiffrement de texte à l'aide de cesar

```
pie - Chiffrement] $ ./lettre 200 > texte.bin
[marod1@l4712-15 Source - Entropie - Chiffrement] $ ./cesar < texte.bin > text.bin
[marod1@l4712-15 Source - Entropie - Chiffrement] $ ./cesar-d < text.bin
RS LRM EETNS ESENMEAGEFTDCTEPIYAYOGE EAA TARAAI SAATEEREPTLIHIDRYEFVENKPLIAUTIA EIOCNHAENA TEWOP TDOHO OEV H ETT OCEIO LYSBMTIENRLIINRRRS H IHBE AAOANSEGPHNNC DEREETMTIAI QUEGRNI IERHEETE R S EERNNT[ma
pie - Chiffrement] $ cat texte.bin
RS LRM EETNS ESENMEAGEFTDCTEPIYAYOGE EAA TARAAI SAATEEREPTLIHIDRYEFVENKPLIAUTIA EIOCNHAENA TEWOP TDOHO OEV H ETT OCEIO LYSBMTIENRLIINRRRS H IHBE AAOANSEGPHNNC DEREETMTIAI QUEGRNI IERHEETE R S EERNNT[ma
pie - Chiffrement] $ cat text.bin
JV OUP HNWQV HVHQPHDHJINGFWHSLWDBRJH HOD WOUDDL VODWHUHSWOLKLGUBHIYHQNSOLDXWLD HLFQKDHQD WHZGS WGRKRR RHY K HWW RFHLR OBVEPWLHOUOLLQUUV K LKEH DORDQVHJIKQF GHUHHWPWLDL RXHJUQL LHUKHMM U V HHUQW[ma
pie - Chiffrement] $
```

Figure 3: Chiffrement de lettre à l'aide de cesar

b)

La figure ci-dessous montre un exemple de commandes réalisées dans la console pour répondre à cette question.

```
[marod1@l4712-15 Source - Entropie - Chiffrement] $ ./h-lettre < texte.bin > test.csv
[marod1@l4712-15 Source - Entropie - Chiffrement] $ ./h-lettre < text.bin > test.csv
[marod1@l4712-15 Source - Entropie - Chiffrement] $ ./h-lettre < text.bin > test2.csv
```

Figure 4: Utilisation de h-lettre

Les quatre figures ci-dessous sont des histogrammes ordonnées montrant la proportion de chaque lettre dans nos fichiers.

Texte sans chiffrement

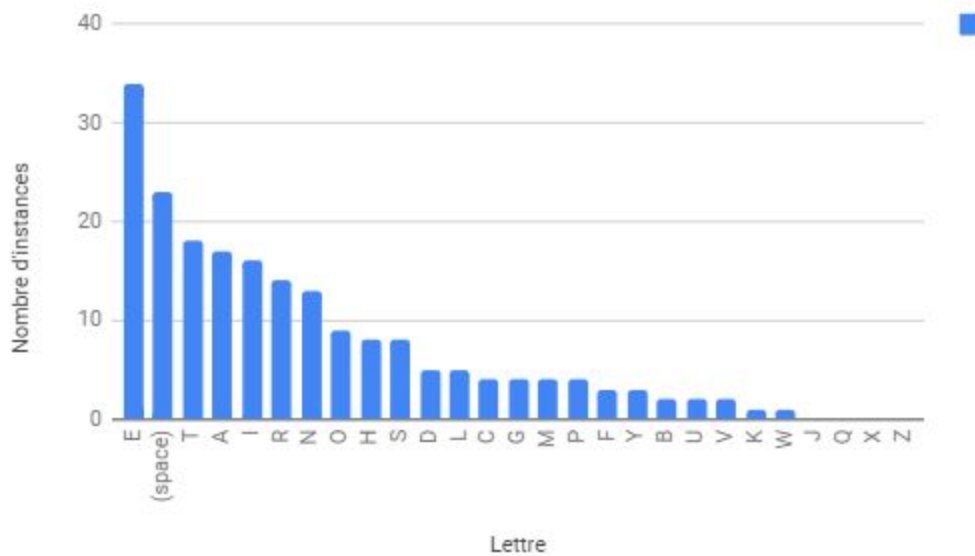


Figure 5: Histogramme - Texte sans chiffrement

Texte avec chiffrement

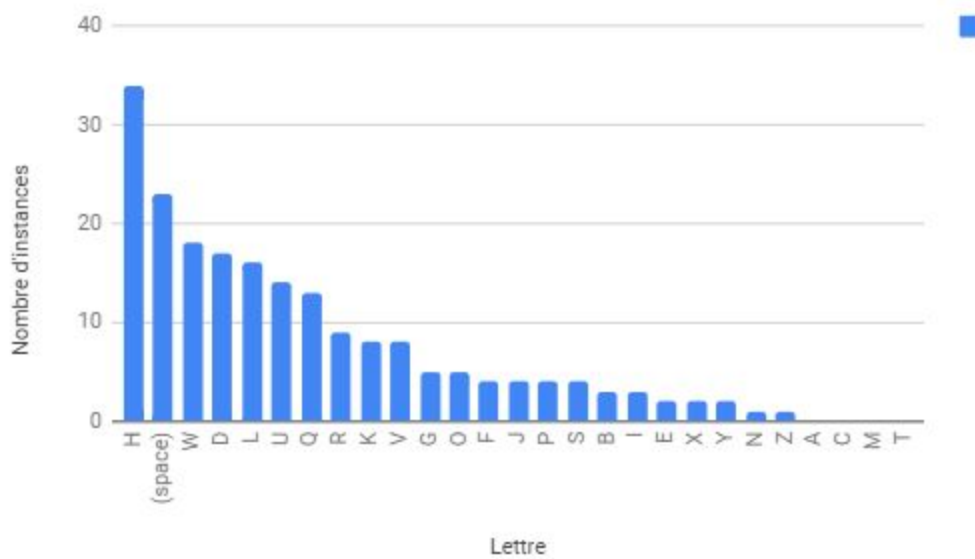


Figure 6: Histogramme - Texte avec chiffrement

Lettre sans chiffrement

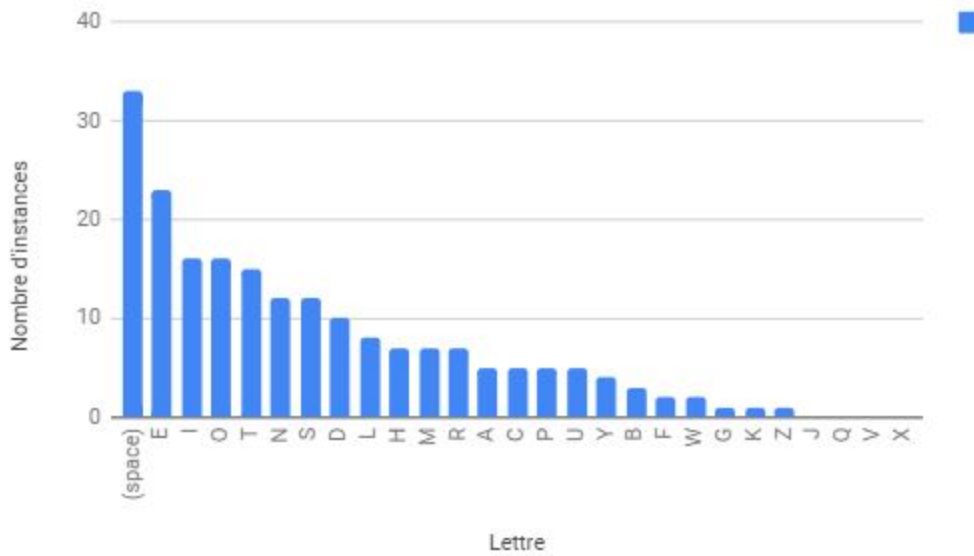


Figure 7: Histogramme - Lettre sans chiffrement

Lettre sans chiffrement

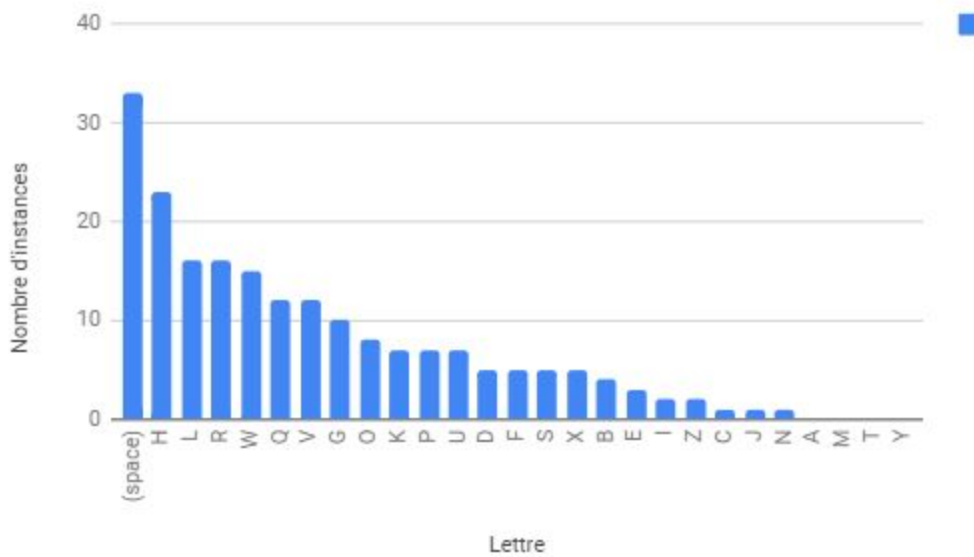


Figure 8: Histogramme - Lettre avec chiffrement

c)

Les résultats obtenus correspondent à ce qu'on s'attend à observer. En effet, les histogrammes pour les sources lettre et texte sans chiffrement sont très similaires. Ceci est normal puisque lettre génère ses lettres en fonction de leur fréquence dans la langue anglaise tandis que texte choisit un texte en anglais. On s'attend donc à ce que les deux histogrammes soient similaires avec les lettres les plus fréquentes qui s'y retrouvent plus de fois que les lettres les moins fréquentes. C'est exactement le phénomène observé avec l'espace, le e et le t étant représentés le plus de fois et qui sont les lettres les plus fréquentes en anglais. Pour ce qui est du chiffrement, on remarque le même graphique qu'auparavant mais avec des lettres différentes. Ceci est normal puisque le chiffrement remplace les lettres du texte original par une autre lettre (par exemple tous les e deviennent des h). Par contre, il est intéressant de remarquer que notre chiffrement s'est effectué de la même manière dans le cas de lettre et de texte. Il semble donc que cesar ne chiffre pas aléatoirement et est donc peu sécuritaire.

Dans le cas où nous voudrions analyser deux lettres, il est évident que les fréquences seraient plus élevées dans le cas de texte puisque des séquences telles que "th" sont fréquentes en anglais. Lettre ne tiendrait pas compte des fréquences de telles séquences et il aurait forcément moins d'instances dans le texte généré.

d)

Si on analysait les fréquences pour deux lettres consécutives, il serait plus facile de déchiffrer le texte généré par texte mais aussi difficile de déchiffrer le texte généré par lettre pour les raisons mentionnées en c). En résumé, les séquences telles que "ee" ou "th" seraient beaucoup plus fréquentes dans un texte en anglais et il serait donc facile de trouver la correspondance de ces lettres chiffrées si on tentait de déchiffrer le texte. Le texte de Lettre étant seulement une séquence de lettres aléatoires, les fréquences de combinaisons de lettres ne seraient pas significatives. Il est difficile de savoir si nous avons correctement déchiffré son texte. S'il est possible d'obtenir un fichier généré avec Texte, on pourrait alors facilement déchiffrer ce fichier et le chiffrement pour chaque lettre (par exemple T = H) devrait correspondre d'après nos observations en b). Il serait alors facile de déchiffrer le code généré par lettre.

Question 3 - Masque jetable

a)

Nous obtenons une entropie avec h-bit de **0.999897** et avec h-ascii de **7.811901** sur le fichier généré avec monnaie. Pour le fichier généré avec binaire, nous obtenons une entropie avec h-bit de **0.947331** et de **0.780522** avec h-ascii.

```

[marod1@l4712-15 Source - Entropie - Chiffrement] $ ./monnaie 1024 > test
[marod1@l4712-15 Source - Entropie - Chiffrement] $ ./binaire 1024 > test1
[marod1@l4712-15 Source - Entropie - Chiffrement] $ ./h-bit < test
0 = 4047
1 = 4145
Nombre total de bits : 8192
Entropie du texte entre : 0.999897
[marod1@l4712-15 Source - Entropie - Chiffrement] $
[marod1@l4712-15 Source - Entropie - Chiffrement] $ ./h-ascii < test1
Nombre total d'octets : 1024
Entropie de l'entree : 0.780522
[marod1@l4712-15 Source - Entropie - Chiffrement] $ ./h-ascii < test
Nombre total d'octets : 1024
Entropie de l'entree : 7.811901
[marod1@l4712-15 Source - Entropie - Chiffrement] $ ./h-bit < test1
0 = 5196
1 = 2996
Nombre total de bits : 8192
Entropie du texte entre : 0.947331
[marod1@l4712-15 Source - Entropie - Chiffrement] $ █

```

Figure 9: Lignes de commandes pour la question 3 a)

b) L'entropie (ascii et bit) est très similaire dans les deux cas.

```

Terminal
Fichier Edition Affichage Rechercher Terminal Aide
[marod1@l4712-14 Source - Entropie - Chiffrement] $ ./monnaie 1024 > fichierb
[marod1@l4712-14 Source - Entropie - Chiffrement] $ ./binaire 1024 > fichierb
[marod1@l4712-14 Source - Entropie - Chiffrement] $ ./monnaie 1024 > clee
[marod1@l4712-14 Source - Entropie - Chiffrement] $ ./masque clee 1024 fichierb
sortiem
[marod1@l4712-14 Source - Entropie - Chiffrement] $ ./masque clee 1024 fichierb
sortieb
[marod1@l4712-14 Source - Entropie - Chiffrement] $ ./h-bit fichierb
h-bit ne prend aucun argument et utilise l'entree standard
[marod1@l4712-14 Source - Entropie - Chiffrement] $ ./h-bit < fichierb
0 = 5124
1 = 3068
Nombre total de bits : 8192
Entropie du texte entre : 0.954073
[marod1@l4712-14 Source - Entropie - Chiffrement] $ ./h-bit < fichierb
0 = 4073
1 = 4119
Nombre total de bits : 8192
Entropie du texte entre : 0.999977
[marod1@l4712-14 Source - Entropie - Chiffrement] $ ./h-ascii < fichierb
Nombre total d'octets : 1024
Entropie de l'entree : 7.805173
[marod1@l4712-14 Source - Entropie - Chiffrement] $ ./h-ascii < fichierb
Nombre total d'octets : 1024
Entropie de l'entree : 0.809727
[marod1@l4712-14 Source - Entropie - Chiffrement] $ ./h-bit < sortieb
0 = 4128
1 = 4064
Nombre total de bits : 8192
Entropie du texte entre : 0.999956
[marod1@l4712-14 Source - Entropie - Chiffrement] $ ./h-bit < sortiem
0 = 4037
1 = 4155
Nombre total de bits : 8192
Entropie du texte entre : 0.999850
[marod1@l4712-14 Source - Entropie - Chiffrement] $ ./h-ascii < sortieb
Nombre total d'octets : 1024
Entropie de l'entree : 7.810806
[marod1@l4712-14 Source - Entropie - Chiffrement] $ ./h-ascii < sortiem
Nombre total d'octets : 1024
Entropie de l'entree : 7.805534
[marod1@l4712-14 Source - Entropie - Chiffrement] $ █

```

Figure 10: Lignes de commandes pour la question 3 b)

c)

La méthode de chiffrement par masque jetable est une manière simple et sécuritaire de chiffrer ces messages. Par contre, elle comporte certaines difficultés d'application. Il faut s'assurer que les deux parties aient la bonne clé sinon l'essence du message pourrait s'en trouver complètement faussé. Il faut aussi s'assurer que la clé générée est générée de façon parfaitement aléatoire.

Question 4 - Analyse de risque

a) Au site B, la probabilité d'incident est de 25% et l'impact d'un ouragan est la destruction de nos installations, qui ont coûté 100.000\$. Donc, l'espérance de perte est $E = 0.25 \times 100.000\$ = 25.000\$$. Au site A, la probabilité d'incident serait proche de 0% et l'impact serait 500.000\$. Alors $E = 0 \times 500.000\$ = 0\$$.

$E(\text{Site A}) = 0\$$

$E(\text{Site B}) = 25.000\$$

La différence de prix entre les deux sites est de 400.000\$. Il est clair que ça vaut la peine payer 100.000\$ et avoir une espérance de perte de 25.000\$ que payer 500.000\$ pour avoir une espérance de perte de 0\$. Il faudrait $400000/25000 = 16$ ans pour qu'il soit plus avantageux d'avoir acheté les installations dans le site A.

b)

1. Intégrité, parce que le jeu ne sera pas juste.
2. Disponibilité parce que les clients ne pourront pas se connecter.
3. Confidentialité parce que les données des clients seront exposées.

c)

Acteur	Capacité	Opportunité	Motivation	Probabilité	Impact	Risque
Tricheur	4	4	4	4	2	8
C.O	1	4	1	2	2	3
Concurrents	2	4	2	2.66	2	5.32

Acteur	Capacité	Opportunité	Motivation	Probabilité	Impact	Risque
--------	----------	-------------	------------	-------------	--------	--------

Tricheur	1	4	1	2	4	8
C.O	4	4	1	3	4	12
Concurrents	2	4	4	3.33	4	13.32

Acteur	Capacité	Opportunité	Motivation	Probabilité	Impact	Risque
Tricheur	1	3	1	1.66	3	4.98
C.O	4	3	4	3.66	3	10.98
Concurrents	1	3	2	2	3	6

On peut clairement voir qu'au Scenario I, l'acteur plus dangereux serait le tricheur, au Scenario II, il serait les sites de poker concurrents et au Scenario III il serait le crime organisé.

d)

1. La motivation des concurrents serait plus grande parce qu'ils veulent savoir comment vous avez si bien réussi. Alors ça changerait la probabilité aussi, de 2.66 à 3.33 et le risque total de 5.32 à 6.66, mais l'acteur plus dangereux continuerai à être les tricheurs.
2. La motivation de la mafia serait plus grande parce qu'ils veulent obtenir leur argent donc une motivation de 4 ferait une probabilité de 4 et en conséquence un risque de 16, le plus haut possible. Le C.O deviendrait l'acteur plus dangereux.
3. Ça changerait l'opportunité des tricheurs parce que le système est mieux donc il y aura une probabilité de 1 par un impact de 3, ça ferait un risque de 3 donc très faible.

e)

Acteur	Capacité	Opportunité	Motivation	Probabilité	Impact	Risque
Tricheur	1	3	1	1.66	3	4.98
C.O	4	3	4	3.66	3	10.98
Concurrents	1	1	2	1.33	3	4

Si la main-d'œuvre est si bon marché et le prix si compétitif, il est clair que cela en vaut la peine, mais l'objectif dans ce scénario devrait être de minimiser l'acteur le plus dangereux qui est la mafia.

Partie B

Question 1 - Codage

a)

σ : [0-9] puisque le NIP est composé de chiffres uniquement.

τ : [0-9] puisqu'on envoie tout simplement le NIP 2 fois au codeur et l'alphabet est donc identique

τ' : [L'ensemble de la table ASCII] puisqu'on substitue et permute les bits envoyés par le codeur 16 fois et donc le résultat final pourrait être constitué de n'importe quel symbole de la table ASCII qui correspond à la nouvelle séquence de bits.

b)

σ : Le langage est les nombres entre 0000 et 9999 puisque le NIP est composé de 4 chiffres.

τ : Le langage est les nombres entre 00000000 et 99999999 puisqu'on envoie le NIP deux fois pour éviter les erreurs de transmissions et que celui est composé de 4 chiffres.

τ' : Le langage est n'importe quelle séquences de 8 caractères ASCII puisque la substitution et la permutation change l'alphabet dans ce cas et l'agrandit à l'ensemble de la table ASCII.

c)

1- Un attaquant peut intercepter le message et en modifier le contenu (le NIP) pour une valeur de son choix et ainsi avoir accès au compte de la personne.

2- Le système est vulnérable aux injections SQL comme entrée de données et des modifications non permises sur la base de données pourraient être effectuées.

3- Il existe des machines à craquer DES et ce standard n'est plus aussi sûr qu'avant.

4- On ne valide jamais l'ancien NIP donc n'importe qui peut modifier le nouveau NIP

5- On ne valide pas l'heure de l'envoi d'une requête. Un attaquant pourrait donc intercepter une requête, la craquer et puis envoyer la requête modifiée des jours plus tard.

e)

1- La première modification élimine l'attaque à l'aide de la machine à craquer DES puisque même une fois craquer, on ne peut pas obtenir le NIP.

2- Même si on craque le NIP, le timestamp ne sera pas le bon puisqu'on aura pris le temps de craquer le message.

3- Même si l'attaquant tente de modifier le message pour un nouveau NIP qui lui convient, il ne connaît pas l'ancien NIP et il n'aura pas le temps de le craquer à cause du timestamp.

f)

La troisième option me semble la meilleure puisqu'on a une deuxième vérification (2FA). En effet, pour changer un NIP, on doit non seulement avoir la carte bancaire mais aussi connaître l'ancien NIP. Le timestamp permet de valider que le message est récent pour éviter que quelqu'un craque les NIPs et les manipulent. C'est donc l'option qui corrige le plus d'attaques mentionnées en b).

Question 2 - Certificats à clé publique, HTTPS et SSL

A) Firefox donne une erreur « This Connection is Untrusted » à cause du certificat, qui est self-signed et expiré.

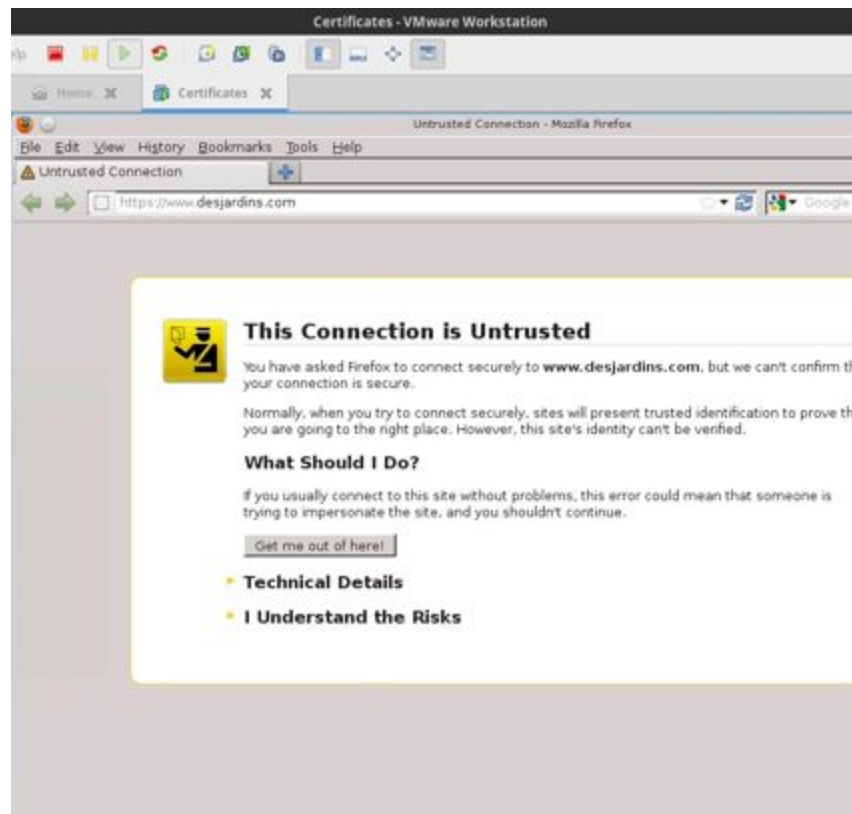


Figure 12: Q2 a)

B) Il y a plusieurs manières de découvrir qu'un site est frauduleux comme l'URL, le certificat SSL ou les liens qu'il y a dans la page, qui normalement seront vrais.

C) Maintenant on peut accéder au site web parfaitement.

D) Comme le certificat est self-signed on a la même erreur.

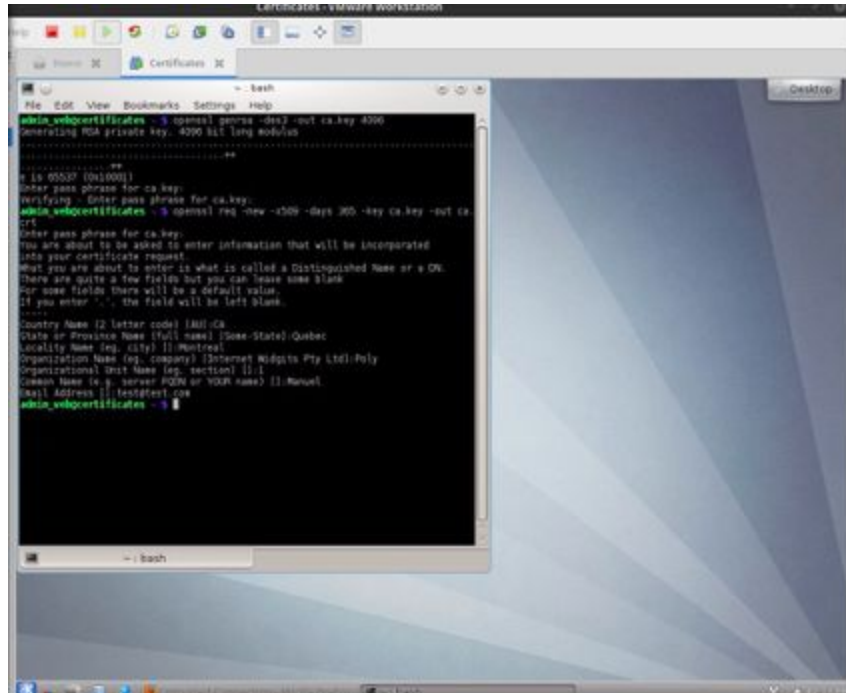


Figure 13: Q2 d)

- E)** Sur rbc.com on a une erreur « Invalid security certificate » et aussi c'est expiré. Sur bmo.com la même erreur à cause de « No issuer chain was provided. »
- F)** Parce que nous avons dit à Firefox de ne pas stocker de façon permanente l'exception et ensuite nous avons effacé le cache, les cookies et les exceptions de sécurité donc l'erreur réapparaît.
- G)** Comme avant, l'erreur réapparaît.
- H)** Comme avant, l'erreur réapparaît.
- I)** Parce que nous ne connaissons pas l'origine (entreprise ou personne délivrant le certificat) donc accepter un certificat auto-signé ou pire, une CA, est une énorme menace de sécurité pour le système que les gens peuvent exploiter, comme vous dites à Firefox de, dès maintenant, commencer à faire confiance à ces faux certificats.

Question 3 - Chiffrement par bloc et modes d'opération

a)

Voici le résultat du chiffrement EBC obtenu:

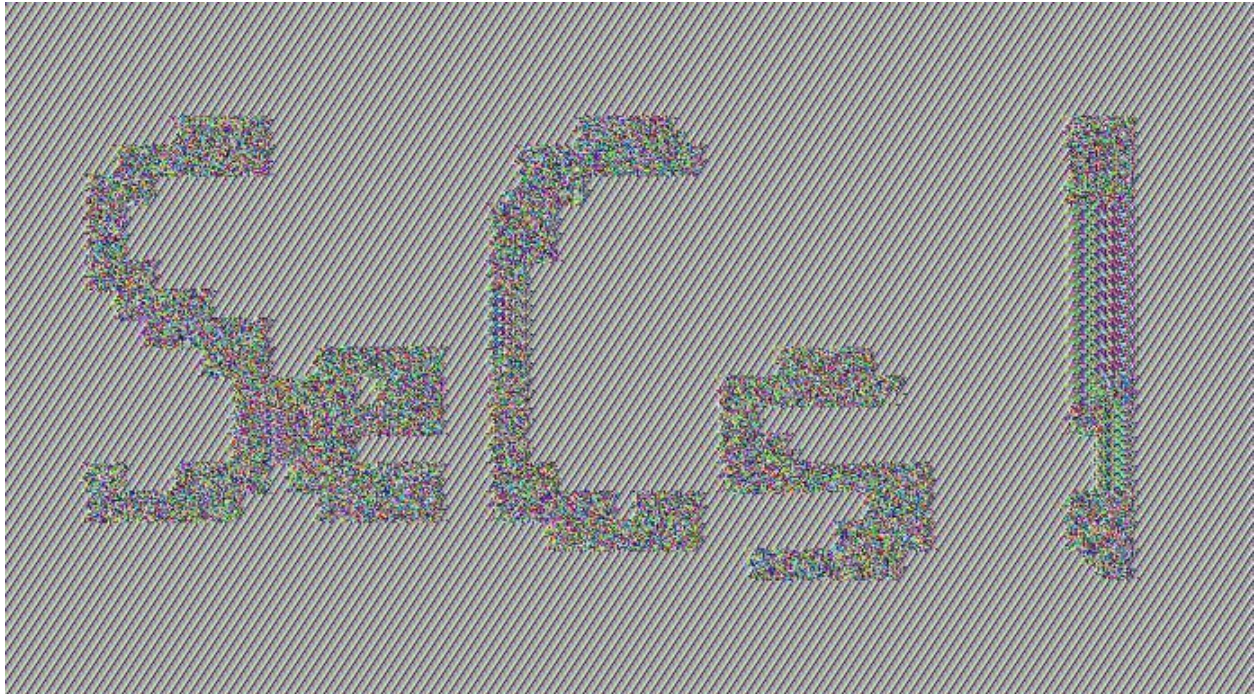


Figure 14: Chiffrement EBC

On constate que cette méthode de chiffrement est assez faible puisqu'il est très facile de voir le mot de passe après le chiffrement. En effet, un des plus grands inconvénients du chiffrement ECB est que deux blocs non chiffrés identiques seront chiffrés de la même manière par la suite. Notre fichier initial n'ayant que deux motifs de couleur, le chiffrement final n'a aussi que deux motifs et le chiffrement est donc très faible.

b)

Voici le résultat du chiffrement CBC obtenu:

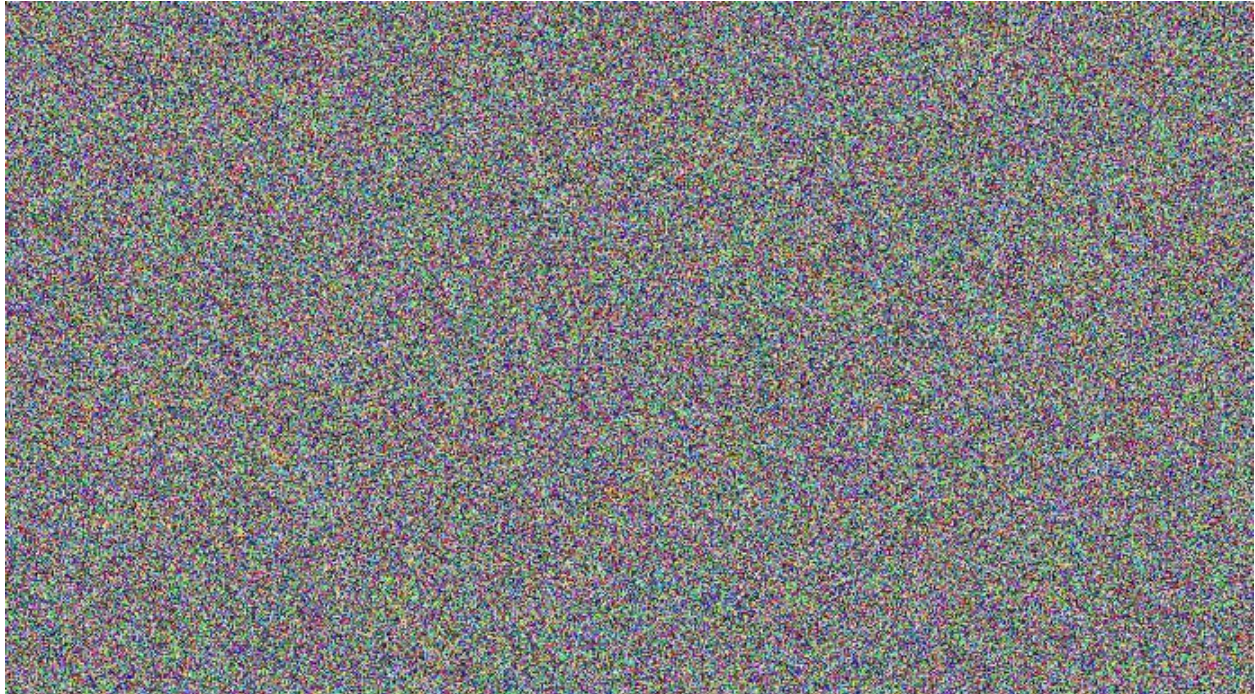


Figure 15: Chiffrement CBC

On constate tout de suite que ce chiffrement est nettement supérieur au chiffrement précédent. On ne peut pas distinguer le mot de passe contrairement à la question 3a).

c)

Après avoir réalisé les tests de chiffrement par bloc, il est évident qu'on peut voir qu'il est primordial de bien choisir son chiffrement puisque les résultats sont bien différents. La méthode de chiffrement EBC a pour avantage d'être simple mais les blocs non chiffrés identiques seront chiffrés de la même manière. Il n'est donc pas judicieux d'utiliser ce chiffrement sur des fichiers avec beaucoup de blocs identiques comme nous avons fait dans ce travail pratique. En ce qui a trait à la méthode de chiffrement CBC, cette méthode est beaucoup plus robuste puisqu'elle prend en compte non seulement le bloc non chiffré courant mais aussi le bloc chiffré précédent lors de son chiffrement. On évite donc le problème de blocs chiffrés identiques du chiffrement EBC. Par contre, cette méthode est plus complexe à implémenter et le déchiffrement est plus long puisque les blocs doivent être décodés en ordre.

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

A) Le fichier ne contient pas des mots de passe parce que ça serait un problème de sécurité. Les permissions sont -rw-r--r--, donc tout le monde peut accéder au fichier parce qu'il n'y a pas des mots de passe.

B) Les deux fichiers ont été modifiés parce que le fichier passwd a ajouté le nouvel utilisateur et shadow a ajouté le mot de passe du nouvel utilisateur.

C) Cette fois c'est seulement shadow qui change parce que c'est le fichier des mots de passe. Seulement root et le groupe de root peuvent accéder au fichier justement parce que ces informations (mots de passe) sont très importantes.

D) Oui, shadow a été modifié parce que même si le mot de passe est le même, le fichier n'a pas les mots de passe en texte clair, mais ils sont cryptés et donc changent à chaque fois.

E) Ça n'est pas possible pour la même raison pour laquelle le fichier est modifié même si le mot de passe est le même, parce que ce qu'on peut voir dans shadow que c'est crypté et pas interchangeable avec des autres utilisateurs. C'est clairement un type de cryptage dynamique.

F) Les deux fichiers ont été modifiés pour effacer toute l'information sur l'utilisateur. (mot de passe, group id...)

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

a)

Le fichier john.pot contient des hashes et des mots de passe obtenus par John. En comparant ces hash avec les hashes des users dans password1, on remarque que nous avons obtenus les informations de connexions qui suivent:

- **user: inf4420 password: 0244fni**
- **user: john password: john1**
- **user: david password: claudia**
- **user: admin password: security**


```

Mdp john # cat john.pot
$1$Wila6SGN$LPLfCWuikEZkOb7CPT01p.:0244fni
$1$n/P09Tgu$CAs0ZntIFmZk3tAfrZY2B0:john1
$1$Aw/cHolc$laW8KVkQeJAernWE1TL3B/:claudia
$1$arMaK13M$PMY2T2poiPR4pdGW26rlw0:security

Mdp john # cat /root/password1
root:$1$f38ucvOp$jHAsVh50HdqAGIUVPesrU/:0:0:root:/root:/bin/bash
bin:*:1:1:bin:/bin:/sbin/nologin
daemon:*:2:2:daemon:/sbin:/sbin/nologin
adm:*:3:4:adm:/var/adm:/sbin/nologin
lp:*:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:*:5:0:sync:/sbin:/bin/sync
shutdown:*:6:0:shutdown:/sbin:/sbin/shutdown
halt:*:7:0:halt:/sbin:/sbin/halt
mail:*:8:12:mail:/var/spool/mail:/sbin/nologin
news:*:9:13:news:/etc/news:
uucp:*:10:14:uucp:/var/spool/uucp:/sbin/nologin
operator:*:11:0:operator:/root:/sbin/nologin
games:*:12:100:games:/usr/games:/sbin/nologin
gopher:*:13:30:gopher:/var/gopher:/sbin/nologin
ftp:*:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:*:99:99:Nobody:/:/sbin/nologin
dbus:!!:81:81:System message bus:/:/sbin/nologin
ucsa:!!:69:69:virtual console memory owner:/dev:/sbin/nologin
rpm:!!:37:37:/:var/lib/rpm:/sbin/nologin
haldaemon:!!:68:68:HAL daemon:/:/sbin/nologin
pcap:!!:77:77:/:var/arpwatch:/sbin/nologin
nscd:!!:28:28:NSCD Daemon:/:/sbin/nologin
named:!!:25:25:Named:/var/named:/sbin/nologin
netdump:!!:34:34:Network Crash Dump user:/var/crash:/bin/bash
sshd:!!:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
rpc:!!:32:32:Portmapper RPC user:/:/sbin/nologin
mailnull:!!:47:47:/:var/spool/mqueue:/sbin/nologin
smmsp:!!:51:51:/:var/spool/mqueue:/sbin/nologin
rpcuser:!!:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin
nfsnobody:!!:65534:65534:Anonymous NFS User:/var/lib/nfs:/sbin/nologin
inf4420:$1$Wila6SGN$LPLfCWuikEZkOb7CPT01p.:500:100:/:home/inf4420:/bin/bash
john:$1$n/P09Tgu$CAs0ZntIFmZk3tAfrZY2B0:501:100:/:home/john:/bin/bash
david:$1$Aw/cHolc$laW8KVkQeJAernWE1TL3B/:502:100:/:home/david:/bin/bash
bruce:$1$IfOTPg76$Mxa0QIZ2kA6rauJTn8Fnt.:503:100:/:home/bruce:/bin/bash
clark:$1$9QY4DdGZ$RVaeFvNu4r2P.UW9sx4Xo/:504:100:/:home/clark:/bin/bash
admin:$1$arMaK13M$PMY2T2poiPR4pdGW26rlw0:505:100:/:home/admin:/bin/bash
nettech:$1$aEZaMa1M$U1K2tS0xq7hFr1xWEdFr//:506:100:/:home/nettech:/bin/bash

```

Figure 16 et 17: MDP obtenus à l'aide de password1

Pour le fichier password2, nous avons obtenu les informations de connexions suivantes:

- user: lola password: niemtel
- user: andre password: Tigers5
- use: morning password: 3sunshine

```
Home X Mdp X
Mdp john # john /root/password2
Loaded 5 password hashes with 5 different salts (FreeBSD MD5 [32/64 X2])
niemtel (lola)
Tigers5 (andre)
3sunshine (morning)
guesses: 3 time: 0:00:08:49 (3) c/s: 17822 trying: 34750440 - 34750441
guesses: 3 time: 0:00:08:56 (3) c/s: 17827 trying: shatlind - shatlief
Session aborted
```

Figure 18: MDP obtenus à l'aide de password2

b)

La formule de l'entropie est :

$$H(x) = \sum(p_i * \log_2(\frac{1}{p_i})).$$

[a-zA-Z]:

$$H(x) = \log_2 52 \cong 5.7 \text{ bits}$$

[a-zA-Z0-9]:

$$H(x) = \log_2 62 \cong 5.95 \text{ bits}$$

L'ensemble de la table ascii:

Il y a 128 caractères dans la table ASCII et donc l'entropie est:

$$H(x) = \log_2 128 \cong 7 \text{ bits}$$

c)

Il faut créer un mot de passe qui emprunte des caractères à un grand alphabet afin d'augmenter la sécurité de nos informations.

d)

- Il faut éviter les mots courants
- Il faut avoir des majuscules et des chiffres
- Il faut qu'il soit le plus long possible

Partie C

Question 1 - Échec du protocole RSA

a)

La formule de déchiffrement est $\text{nbrOriginal}^e \bmod n = \text{nbrEncrypté}$. Nous possédons e et n de la clé publique de Bob et nous savons que les nombres originaux vont de 0 à 25. Nous n'avons donc qu'à tester les 25 possibilités de nbrOriginal pour trouver celle qui fonctionne dans l'équation.

b)

Nous avons utilisé la formule en a) en faisant varier nbrOriginal de 0 à 25 jusqu'à ce que la formule soit égale et avons obtenu le code déchiffré suivant:

0 13 6 11 4 = AMFKD

c)

Il faut éviter les 0 et les 1 dans le message original puisqu'ils retournent la même valeur lorsqu'ils sont chiffrés !

Question 2 - Déchiffrement "simple"

Le texte est:

"firmness his invasions on the rights of the people.he has refused for a long time after such dissolutions to cause others to be elected whereby the legislative powers incapable of annihilation have r"

Tout d'abord, nous avons exécuté le script de fréquence sur le texte encodé, puis nous avons effectué une recherche simple et remplacé le résultat de la fréquence par la fréquence de la langue anglaise. Cela a rendu le texte plus lisible et avec une simple intuition et une recherche sur google, nous avons pu déchiffrer le texte.