1) Expliquez en détail son rôle (qu'est qu'il représente typiquement), sa structure, son format, son contenu…

Ce fichier contient tous les mots de passe des utilisateurs locaux de manière cryptés, c'est à dire, le résultat de hachage qu'il faut obtenir lorsque l'on hash le bon mot de passe.
Il est utilisé par le système d'exploitation lorsque qu'un utilisateur souhaite s'authentifier sur la machine. Ce fichier permet de comparer si le hachage d'un mot de passe donnée correspond bien à celui du qu'à produit le bon mot de passe.
Une ligne est constituée comme suit :
<nom utilisateur>:$<algo>$<salt>$<hashage>:<nb de jour depuis changement du MDP>:<nb de jour avant de pouvoir changer le MDP>:<Nb de jour avant MDP périmé>:<Nb de jour avertissement MDP périmé>:<limite de péremption MDP>:<Date de desactivation du compte>:<N/A>

algo : ici la valeur 5 correspond à l'algorithme sha-256
salt : correspond à une valeur aléatoire permettant d'injecter une inconnue supplémentaire lors du hashage. Cela rend impossible de pouvoir craquer le MDP en ayant que le hashage du MDP.
hashage : Le résultat du hashage du mot de passe en ayant utilisé l'agorithme et le sel.

2) Lancer le logiciel John The Ripper (supposé installé conformément au support de TP) pour cracker le password qu'il contient, en faisant l'hypothèse suivante : un maximum de 4 caractères alphanumériques. Produire une copie d'écran de la commande de lancement et de son résultat (avec au moins 4 lignes de status). Vous devez bien entendu donner le mot de passe que vous avez "deviné".

$ ./john --incremental=alnum --max-length=4 <fichier_shadow>

```
[root@c8-milazzo ~]# rm -f ./sec105/john-1.9.0-jumbo-1/run/john.pot; ./sec105/john-1.9.0-jumbo-1/run/joh
n --incremental=alnum --max-length=4 myshadowMILAZZO
Using default input encoding: UTF-8
Loaded 1 password hash (sha256crypt, crypt(3) $5$ [SHA256 256/256 AVX2 8x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:00:03 0,16% (ETA: 18:34:59) 0g/s 7927p/s 7927c/s 7927C/s dceo..tjsh
0g 0:00:00:11 0,60% (ETA: 18:35:00) 0g/s 8017p/s 8017c/s 8017C/s pnzs..tcty
0g 0:00:00:40 2,15% (ETA: 18:35:22) 0g/s 8075p/s 8075c/s 8075C/s 83vv..8moh
0g 0:00:00:57 3,07% (ETA: 18:35:23) 0g/s 8079p/s 8079c/s 8079C/s lkv7..jhzy
```

```
7sWR             (test)
1g 0:00:25:29 DONE (2021-05-13 18:29) 0.000653g/s 8114p/s 8114c/s 8114C/s 77Wr..7EWu
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Le mot de passe pour l'utilisateur test est **7sWR**

3) Indiquer le temps estimé par John pour ce calcul, et produire la formule mathématique et le calcul qui vous permet de confirmer ce temps maximal (en minutes)

D'après les 4 statuts ci-dessus on voit que le programme traite en moyenne 8050 mots-de-passe par secondes.
Le champs des possible pour un mot de passe alphanumérique faisant entre 1 et 4 caractères s'exprime comme suit :

$\sum_{i=1}^{n} c^{i}$ avec **i** = le nombre de caractère possible et **n** = la longueur max du mot de passe.

Dans notre cas on pose :
n = 62 (26 lettres min + 26 lettres maj + 10 chiffres)
i = 4 (4 caractères max)

Cela donne un univers des possible de $62^1 + 62^2 + 62^3 + 62^4 = 15\,018\,570$ éléments.

Pour 15 018 570 mots de passe possibles et environ 8050 MDP traités par seconde

on calcule temps max estimé = $15\,018\,570 \Big/ 8050 \approx 1865$ secondes

Soit 1865 / 60 qui donne environ **31 minutes.**

4) Lancer successivement 4 fois ce calcul en vous intéressant au temps (faire précéder la commande john de la commande linux time) (faites autre chose en même temps, c'est long...). Donner ces 4 temps et les comparer à la valeur de la question 3. Commentaire.

Premier temps :

```
0g 0:00:25:45 82,17% (ETA: 19:05:30) 0g/s 7986p/s 7986c/s 7986C/s 2yW1..jmW7
7sWR             (test)
1g 0:00:25:53 DONE (2021-05-13 19:00) 0.000643g/s 7986p/s 7986c/s 7986C/s 77Wr..7EWu
Use the "--show" option to display all of the cracked passwords reliably
Session completed

real    25m54.486s
user    99m29.968s
sys     0m1.082s
```

Deuxième temps :

```
0g 0:00:25:19 82,12% (ETA: 19:31:16) 0g/s 8117p/s 8117c/s 8117C/s boWj..bEWD
7sWR             (test)
1g 0:00:25:28 DONE (2021-05-13 19:25) 0.000654g/s 8116p/s 8116c/s 8116C/s 77Wr..7EWu
Use the "--show" option to display all of the cracked passwords reliably
Session completed

real    25m29.638s
user    99m23.431s
sys     0m0.728s
```

Troisième temps :

```
7sWR             (test)
1g 0:00:25:35 DONE (2021-05-13 19:52) 0.000651g/s 8080p/s 8080c/s 8080C/s 77Wr..7EWu
Use the "--show" option to display all of the cracked passwords reliably
Session completed

real    25m36.393s
user    99m36.044s
sys     0m0.735s
```

Quatrième temps :

```
7sWR             (test)
1g 0:00:25:33 DONE (2021-05-13 20:17) 0.000651g/s 8089p/s 8089c/s 8089C/s 77Wr..7EWu
Use the "--show" option to display all of the cracked passwords reliably
Session completed

real    25m34.754s
user    99m27.200s
sys     0m0.783s
```

Les 4 temps sont identiques à quelques dizaines de millisecondes près.
Cela vient du fait que le mode incrémentale de John the Ripper fonctionne avec des dictionnaires et qu'il va utiliser un ordre pré calculé afin d'améliorer les chance de tomber sur le bon mot de passe. Pour le même dictionnaire, il est normale de tomber sur le même ordre donc à vitesse de traitement égale, le temps de crackage du mot de passe est sensiblement le même.

5) Produire la commande openssl qui vous permet de vérifier que vous avez bien trouvé le mot de passe correspondant au contenu du fichier myshadowVOTRENOM

```
[root@c8-milazzo ~]# cat myshadowMILAZZO |cut -d$ -f4 |cut -d: -f1 && openssl passwd -5 --salt aGp8h.6fR
T65m7kl 7sWR |cut -d$ -f4
HcLPbKdNCduVOb/.Tni/O611keyBozQiPOMjiJm61r0
HcLPbKdNCduVOb/.Tni/O611keyBozQiPOMjiJm61r0
[root@c8-milazzo ~]#
```

6) Expliquez en quoi et pourquoi l'affirmation de la question 5 est erronée

Je ne suis pas sûr de comprendre la question mais je dirait que l'on est jamais sûr d'avoir trouvé le mot de passe mais plutôt UN mot de passe capable de générer le même hash que le vrai mot de passe.