# Examen Synthèse

Cet examen compte pour 40 % de la note finale

|  |  |
| --- | --- |
| **Prénom et nom** | **Matricule** |
| François Massé | 0332492 |

**Correction**

|  |  |
| --- | --- |
| **Questions** | **Résultats** |
| Question 1 – 5 points |  |
| Question 2 – 5 points |  |
| Question 3 – 5 points |  |
| Question 4 – 25 points |  |
| **Total – 40 points** |  |

**Les consignes pour cet examen sont :**

* À la fin de cet examen, vous devez remettre ce cahier d’examen;
* Toute documentation permise;
* L’envoi de notes écrites, par télépathie, via pigeons voyageurs, hiboux, ou à l’aide de tout autre mode de communication, est interdit;
* Veuillez soigner votre langue écrite (maximum 10% de pénalité);
* Tout plagiat entrainera la note de zéro.

# Attention

Les questions 1 à 3 doivent être répondues. Elles comptent pour 5 points chacune.

# QUESTION 1 5 Points

Expliquer la différence entre un algorithme de hachage et un algorithme de protection de mots de passe.

|  |
| --- |
| Un algorithme de hachage va simplement hacher l’information afin de permettre, par exemple, de comparer deux hachs d’un même document pris à des moments différents entre eux pour confirmer qu’il n’y a pas eu de modifications de son contenu.  Pour la protection d’un mot de passe, on veut protéger ceux-ci de l’utilisation d’une «rainbow table» alors on peut ajouter des salts secrets et/ou hacher le hash plusieurs fois pour augmenter le niveau d’obfuscation. |

# QUESTION 2 5 Points

Expliquer la différence entre un algorithme de chiffrement symétrique et asymétrique.

|  |
| --- |
| Symétrique : il y a une seule clé permettant de décrypter le message et elle doit être partagée identiquement par l’envoyeur et le receveur du message. Très sécuritaire  Asymétrique : Chaque membre de la communication possède une clé publique et une clé privée qui sont liées mathématiquement. L’envoyeur du message utilise la clé publique du receveur pour crypter le message et le receveur utilise sa clé privée pour briser le cryptage. Moins sécuritaire comme le type de cryptages est fait pour permettre d’être brisé par une «trap door». |

# QUESTION 3 5 Points

Expliquer la différence entre les concepts de *proof-of-work* et *proof-of-stake*.

|  |
| --- |
| Of work : on veut que la personne qui démontre son proof of work doive faire du travail avec sa machine afin de gagner le droit de créer un bloc. Pour Bitcoin c’est en générant un hash comportant plusieurs 0 en début de ligne.  Of stake : un individu doit mettre sur la ligne un avoir, 32 Eth dans le cas d’Etherium, afin d’entrer dans un bassin pour simplement attendre que se soit ton tour ou d’être choisi au hasard pour créer un nouveau bloc, pas besoin de faire du travail pour rien. |

# Attention

Vous devez seulement répondre à 4 des 8 sous-questions suivantes, chacune des questions compte pour 7,5 points. Si vous répondez à plus de 4 questions seulement les 4 premières seront corrigées.

Pour chacune des questions, vous devez expliquer en vos mots un concept que nous avons abordé en classe. Votre réponse doit comprendre des exemples d’utilisation ou la raison pourquoi ce sujet a été vu dans le cours de *Sécurité des données*.

Pour chacune des questions, vous retrouvez des pistes de réflexion pour vous guider dans la rédaction de votre réponse.

## Question 4.A

**Concept : *Hashing***

*Piste de réflexion : algorithmes, utilisations, protection supplémentaire, protection des mots de passe*

|  |
| --- |
| Hacher est une action irréversible qui consiste à prendre un document informatique de longueur théoriquement infinie et de le représenter par une chaine binaire de longueur prédéfinie par l’algorithme effectuant l’opération. En se faisant, toute l’information qui était contenue dans le document est perdue, mais comme utiliser le même document avec le même algorithme doit donner le même résultat, il est possible de comparer deux hash pour vérifier la validité. On peut donc mettre en mémoire le résultat du hash d’un mot de passe et quelqu’un qui pourrait par la suite voir la BD contenant cette information ne pourrait théoriquement pas accéder au compte puisqu’il ne saurait pas quel mot de passe a produit la chaine hachée. Idéalement, deux entrées différentes ne devraient jamais générer le même hash mais cela est impossible, de nouveaux algorithmes plus complexes de hachages sont donc produits afin de limiter les possibilités de collision. |

## QUESTION 4.B

**Concept : Chiffrement**

*Piste de réflexion : symétrique, asymétrique, signature, échange de clés, algorithmes*

|  |
| --- |
| Pour s’assurer qu’un message n’est lu que par le destinataire attendu, il est préférable de le coder. On ne peut utiliser seulement un hash puisque cela perd l’information initiale du message (irréversible). Il faut donc crypter le message de façon que seule l’utilisation d’une clé permette de le récupérer. La façon la plus sécuritaire est d’utiliser une clé symétrique qui est la seule à permettre de lire le message, mais cela comporte le risque que quelqu’un qui intercepte la clé puisse aussi lire les messages et qu’il faille échanger celle-ci avant de commencer à utiliser le cryptage. On peut aussi utiliser une clé asymétrique ou chaque membre de la communication possède une clé publique et une clé privée connectées mathématiquement. L’envoyeur utilise alors sa clé privée et la clé publique du receveur afin de crypter le message et le receveur peut sa propre clé privée et la clé publique de l’envoyeur pour décrypter le message. Les algorithmes de cryptages permettant ce type d’échange utilisent une «trap door» permettant de briser le code avec des informations mathématiquement liées à ce qui a été utilisé pour créer le code. Malheureusement, il est possible de trouver d’autres «trap door» que celle partagée par les membres de la communication et donc il est plus facile de craquer ce type de cryptages. Il est aussi possible d’inclure un hash du message d’origine pour signer la communication afin que le receveur puisse hacher ce qu’il reçoit et s’assurer que personne n’a modifié le message au cours de son transit. |

## Question 4.C

**Concept : Protocole HTTPS**

*Piste de réflexion : http, tls, échange de clé, certificats, chaine de vérification*

|  |
| --- |
| Un lien http transfère toutes ses communications sur internet en clair, elles peuvent toutes être lues par chaque nœud du réseau qui sert à les transmettre. Pour sécuriser les communications, on utilise plutôt une connexion HTTPS qui permet de chiffrer les communications par l’utilisation de protocoles de cryptages entre les serveurs et leurs clients comme le TLS. Pour ce faire, deux appareils voulant créer une connexion https vont commencer leur communication en utilisant un cryptage asymétrique afin d’échanger leurs informations et les protocoles de cryptages symétriques auxquels ils ont accès. Puis ils vont s’échanger une clé de cryptages symétrique et continuer tous leurs échanges en cryptant leurs messages grâce à cette clé unique. Le reste de la communication est donc plus sécuritaire. |

## Question 4.D

**Concept : Procédure stockée**

*Piste de réflexion : rôle, fonctionnement, avantages, inconvénients*

|  |
| --- |
|  |

## Question 4.E

**Concept : Triggers**

*Piste de réflexion : rôle, fonctionnement, avantages, inconvénients*

|  |
| --- |
|  |

## Question 4.F

**Concept : Vulnérabilité Web – Injection**

*Piste de réflexion : définition, SQL, XSS, risques, mesures en mettre en place pour les contrer*

|  |
| --- |
|  |

## Question 4.G

**Concept : Vulnérabilités Web - Broken access control**

*Piste de réflexion : définition, exemples, mesures en mettre en place pour les contrer*

|  |
| --- |
| Dans une application, certains utilisateurs ne devraient pouvoir faire que certaines actions. On ne veut pas qu’un simple client ait accès à des fonctionnalités d’administrateur par exemple. On pourrait penser qu’en mettant ces opérations ségrégées dans des pages seulement accessibles par un administrateur on pourrait protéger l’application contre de telles problématiques, mais ce n’est pas vraiment vrai sur le web. Lorsqu’une page envoi une commande vers le serveur, elle n’envoie que la commande et le serveur n’a intrinsèquement pas de moyen de savoir si la commande est bien venue d’un browser contenant sa page ou d’une application comme Postman qui a isolé celle-ci. Si la commande contient toutes les informations requises, elle sera traitée, point final. Quelqu’un qui est capable de recréer la commande pourra alors la relancer de n’importe où. Pour y remédier, il y a plusieurs approches complémentaires. D’abord il est possible de faire barrer une adresse IP si elle envoie trop de commandes erronées afin d’empêcher de tester des commandes au hasard jusqu’à en trouver une qui fonctionne. Ensuite, il y a de réclamer de l’envoyeur une confirmation d’identité, un token valide obtenu au moment de la connexion, pour chaque interaction avec le serveur. Une autre approche est de limiter le nombre d’actions qui peuvent être prises à partir du web, aucune commande web ne devrait permettre d’effacer la base de données par exemple. |

## QUESTION 4.H

**Concept : Blockchain**

*Piste de réflexion : fonctionnement, termes, PoW, PoS, bitcoin, ethereum…*

|  |
| --- |
|  |

## Bonus

**Combien avait-il de chats, de souris et d’ours?**

|  |
| --- |
| Il y avait deux chats l’orange et celui qui a un «eye patch» et une souris aux oreilles roudes. |

**Fin de l’examen**