Documentation des énigmes

Sommaire :

Liste des énigmes

Les échanges de Bitcoin

Cette énigme consiste à trouver “n”, le paramètre qui permettra de vérifier une transaction en Bitcoin. Cette énigme est une version simplifiée des réelles transactions en Bitcoin.

On a une transaction telle que : H (H (xympn)). H correspond au code ASCII, on peut donc noter ASCII (ASCII (xympn)). L’énigme consiste à additionner les codes ASCII de “x”, “y”, “m”, “p” et “n”, et à ensuite prendre le code ASCII de ce premier résultat.

“n” est un entier pouvant aller de 0 à l’infini. Ce n’est pas la valeur de “n” que l’on ajoute alors à la somme des valeurs ASCII de “x”, “y”, “m” et “p” mais bien la valeur de son code ASCII aussi. Par exemple, si “n = 15”, on ajoutera ASCII (1) + ASCII (5) = 49 + 53 = 102 au total des codes ASCII déjà calculés.

On obtient alors un nombre et, pour vérifier la transaction, ce nombre doit être divisible par les valeurs “val1” et “val2” définies en fonction des niveaux de difficulté de l’énigme.

L’énigme doit donc résoudre les calculs suivants pour être résolue :

- ➔ ASCII (ASCII (xympn)) % val1 = 0
- ➔ ASCII (ASCII (xympn)) % val2 = 0
- ➔

Niveau 1 :

Ici, “x” représente la valeur d’Alice, “y” la valeur de Bob, “m” le montant de la transaction, “p” un nombre aléatoire et “n” le nombre à retrouver. Les valeurs d’Alice et Bob sont respectivement fixées à “A” et “B”.

“m” est l’entier correspondant au montant de la transaction. Ici, c’est un entier aléatoire entre 1 et 10.

“p” est un nombre généré aléatoirement entre 1 et 100. Ce nombre est là pour complexifier la résolution du problème, pour permettre de sécuriser l’échange dans un cas concret.

“n” est un entier pouvant aller de 0 à l’infini. C’est ce paramètre qui permet de vérifier la transaction, c’est donc celui à déterminer.

A ce stade, on introduit deux entiers, qu’on appellera “val1” et “val2”. Ce sont les entiers qui vont servir de contrainte à la résolution de notre problème. Le nombre trouvé à l’issue du calcul de H(H(xympn)) doit être divisible à la fois par “val1” et par “val2”. “val1” et “val2” sont fixés à 5 et 7.

Niveau 2 :

Ici, “x” représente la valeur de la personne qui émet la transaction, “y” la valeur de la personne qui reçoit la transaction, “m” le montant de la transaction, “p” un nombre aléatoire et “n” le nombre à retrouver. Les valeurs d’Alice et Bob sont respectivement fixées à “A” et “B”.

“x” et “y”, les valeurs respectives des deux personnes de l’échange. “x” est déterminée grâce à l’initiale de la personne à l'initiative de l’échange. C’est le code ASCII de cette lettre, par exemple, pour Oscar, on prend “x = 79” car c’est le code ASCII de “O”. “y” est déterminé de la même façon par l’initiale de la personne qui reçoit la transaction.

“m” est l’entier correspondant au montant de la transaction. Ici, c’est un entier aléatoire entre 1 et 10.

A ce stade, on introduit deux entiers qu’on appellera “val1” et “val2”. Ce sont les entiers qui vont servir de contrainte à la résolution de notre problème. Le nombre trouvé à l’issue du calcul de H(H(xympn)) doit être divisible à la fois par “val1” et par “val2”. “val1” et “val2” sont fixés à 5 et 7.

Par rapport au niveau 1, le niveau 2 propose des prénoms aléatoires, deux paramètres qui changent en plus pour complexifier l’énigme.

Niveau 3 :

Ici, “x” représente la valeur de la personne qui émet la transaction, “y” la valeur de la personne qui reçoit la transaction. “m” le montant de la transaction, “p” un nombre aléatoire et “n” le nombre à retrouver. Les valeurs d’Alice et Bob sont respectivement fixées à “A” et “B”.

“x” et “y” sont les valeurs respectives des deux personnes de l’échange. “x” est déterminée grâce à l’initiale de la personne à l'initiative de l’échange. C’est le code ASCII de cette lettre. Par exemple, pour Oscar, on prend “x = 79” car c’est le code ASCII de “O”. “y” est déterminé de la même façon par l’initiale de la personne qui reçoit la transaction.

“m” est l’entier correspondant au montant de la transaction. Ici, c’est un entier aléatoire entre 1 et 10.

A ce stade, on introduit deux entiers qu’on appellera “val1” et “val2”. Ce sont les entiers qui vont servir de contrainte à la résolution de notre problème. Le nombre trouvé à l’issue du calcul de H(H(xympn)) doit être divisible à la fois par “val1” et par “val2”. “val1” et “val2” sont deux valeurs aléatoires entre 1 et 9, qui ne peuvent pas être égales.

Par rapport au niveau 2, le niveau 3 propose des valeurs “val1” et “val2” aléatoires, ce qui permet de complexifier encore l’énigme.

Vous avez dit sûr … sûr

Pour résoudre cette énigme, il faut trouver un message secret “M” qui est envoyé par Alice à Bob. Alice n'envoie pas son message sans le crypter. Elle utilise donc une clé aléatoire et secrète et envoie le message crypté à Bob. Bob ne pouvant décrypter ce message, il le crypte à son tour et le renvoie à Alice. Alice va à son tour décrypter le message et renvoyer le résultat à Bob qui n'aura plus qu'à décrypter le message. Ainsi, Bob recevra le message “M” d’Alice sans que les deux personnes ne se soient rencontrées.

Pour être clair, on va nommer les variables :

- “M” étant le message d’Alice à décrypter.
- “KA” et “KB”, les clés de chiffrement de respectivement Alice et Bob.
- “MA”, le message crypté qu'envoie Alice.
- “MB”, le message crypté qu’envoie Bob.
- Et enfin “RA”, le message de Bob décrypté par Alice.

On a alors “M” qui est chiffré par Alice avec “KA” et on obtient “MA”. Alice envoie “MA” à Bob qui va le chiffrer avec “KB” et renvoie le résultat “MB”. Alice va déchiffrer ce message avec “KA” et cette fois renvoie “RA”. Bob en déchiffrant “RA” avec “KA” obtient le message “M” d’Alice.

Cependant, tout cela n’est pas sûr car si on connaît ce protocole, on peut obtenir les valeurs de “KA” et “KB” en interceptant simplement les échanges entre les deux interlocuteurs. Pour cela, il suffit de lister les équations :

- On sait que la première est le message M crypté avec KA donc :
- MA = **M + KA**
- La seconde équation est le message MA crypté ensuite avec KB :
- MB = MA + KB = **(M + KA) + KB**
- Enfin, la dernière est le message crypté par Bob mais décrypté par Alice donc :
- RA = MB - KA = (MA + KB) - KA = M + KA + KB - KA = **M + KB**

On peut alors facilement retrouver une clé secrète en soustrayant la 1ere ou la 3eme équation à la seconde.
Par exemple : MB - MA = (M + KA) + KB - (M + KA) = KB
Une fois une clé secrète retrouvée, il suffit de la soustraire à un message crypté par cette même clé. Pour notre exemple, il suffit de faire RA - KB pour retrouver le message M.

Comme toutes ces équations sont uniquement composées d’addition ou de soustraction, il est facile de retrouver une clé ou un message crypté.

Niveaux :

Les différents niveaux changent simplement la valeur maximale possible pour les nombres aléatoires (M, KA, KB). C'est-à-dire qu’au **niveau 1,** la valeur maximale est 10 ; pour le **niveau 2**, c’est 100 et pour le **niveau 3**, elle est de 1000.

Le partage de Shamir

Cette énigme consiste à trouver une valeur nommée “s”, paramètre d’une fonction polynôme de degré 2 écrite sous la forme ”y=ax+s”, “$y = ax^2+bx+s$” ou “$y=ax^3+bx^2+cx+s$” suivant le niveau.

Les valeurs “*a”, “b”, “c”* et “*s”* sont choisies aléatoirement lors de la création de la page html permettant d’obtenir une fonction toujours aléatoire. Pour éviter d’avoir des chiffres vraiment énormes, les 3 points “a”, “b” et “s” sont toujours pris dans une fourchette de -10 à 10 sauf pour le niveau 3 ou c’est entre -5 et 5. Ces 4 valeurs ne sont pas données lors de la création de la page. Elles doivent être retrouvées pour obtenir “s” le code secret.

On obtient 4 valeurs aléatoires nommées “x1”, ”x2”, “x3” et “x4” permettant de calculer la valeur de “y1”, “y2”, “y3” et “y4” obtenue en faisant la formule écrite ci-dessus, ce qui nous permet d’obtenir 3 couples de données (x, y) formant 3 points d’une courbe choisie aléatoirement, nommés respectivement “*A”, “B”* et “*C”*.

Niveau 1 :

Dans le premier niveau comme pour tous, seule la difficulté de l’équation va changer. Le niveau 1 étant le plus simple, on a une équation linéaire basique, à savoir “y=ax+s”, ce qui nous donne 2 équations pour trouver “a” et “s”, 2 étant le minimum pour trouver les valeurs.

Niveau 2 :

Dans le second niveau, l’équation devient un polynôme de degré 2 exprimé sous la forme : “$y1=ax^2+bx+s$”. Le site nous donne 3 équations faites avec des valeurs différentes, ce qui est le minimum pour permettre à l’utilisateur de trouver les différentes valeurs de “a”, “b” et “s”.

Niveau 3 :

Dans le dernier niveau, on a 4 équations polynomiales de degré 3 écrites sous la forme : "$y=ax^3+bx^2+cx+s$".

Substitution

Dans cette énigme, 2 messages nous sont donnés : 1 chiffré et 1 non chiffré. Le non chiffré permet de déchiffrer le premier et donne accès à la phrase secrète qui devra être rentrée et correcte. Ceci va déclencher l'apparition d'un post-scriptum qui prend la forme d'un pangramme permettant d'avoir toutes les lettres et savoir par laquelle la lettre chiffrée a été remplacée.

La phrase secrète est fabriquée de manière aléatoire à partir de petits morceaux de phrase, à savoir un Prénom, un verbe, le mot 'des' et un objet permettant d'avoir une énigme qui change à chaque fois que l'on vient sur le site.

Niveau 1 :

Dans ce niveau, le chiffrement est toujours le même et seule la phrase secrète change, à chaque fois que l'on recharge la page. La substitution se fait toujours avec un décalage de 3 donc :

a devient d

b devient e

…….

y devient b

z devient c

Niveau 2 :

Dans ce niveau, le chiffrement et la phrase changent à chaque fois que l'on recharge la page. Le chiffrement est aléatoire. Comme pour l'énigme précédente, il s'agit d'un décalage. Les lettres peuvent subir un décalage de 2 à 25 crans.

a devient b	*……*	*a devient z*
b devient c	*……*	*b devient a*
….		
y devient z	*……*	*y devient x*
z devient a	*……*	*z devient y*

Niveau 3 :

Comme pour le niveau précédent, le chiffrement et la phrase sont aléatoires. Cependant, le plus gros changement est que nous n'avons plus un chiffrement par décalage mais par modification aléatoire, autrement dit une lettre devient une lettre choisie aléatoirement parmi les autres restantes. Si a devient t, une autre lettre ne pourra pas devenir t à son tour, sinon l'énigme serait infaisable. On a donc dans ce niveau-là une vraie substitution aléatoire.

Solidité d’un mot de passe

Niveau 1 -> consiste à trouver le temps en années pour trouver un mot de passe de “N” caractères.

Niveau 2 -> Avec la vitesse du processeur et le nombre d’années, on doit trouver le nombre d’éléments que l’ordinateur peut calculer.

Niveau 3 -> Avec le nombre d'années et les caractéristiques de l’ordinateur, trouver le nombre de caractères minimal qu’il faut pour atteindre ce nombre d’années.

Niveau 1 :

On donne le nombre de caractères du mot de passe, le nombre de possibilités pour chaque caractère. On donne aussi la vitesse et le nombre de cœurs du processeur.

Nombre de caractères du mot de passe compris entre 3 et 13.
Nombre de caractères total possible 105.
Vitesse du processeur entre 2.4 et 3.7 GHz.
Nombre de cœurs du processeur entre 1 et 8.
Donc :
Nombre d’opération par seconde = vitesseProcesseur * 1000000 * nombreCoeur.
Nombre de possibilités du mot de passe = nombre de possibilité puissance nombre de caractères du mot de passe.

La solution est égale au nombre de possibilités sur le nombre d'opérations.

Si la solution est de l’ordre des heures, alors on pose la question en demandant le temps en heures.
Si la solution est de l’ordre des jours, alors on pose la question en demandant le temps en jours.
Si la solution est de l’ordre des années, alors on pose la question en demandant le temps en années.

Niveau 2 :

Avec la vitesse du processeur et le nombre d’années on doit trouver le nombre d’éléments que l’ordinateur peut calculer.

Niveau 3 :

On donne le nombre total de caractères, la vitesse et le nombre de cœurs du processeur. On donne également un nombre d'années.

Nombre de caractères total possible 105.
Vitesse du processeur entre 2.4 et 3.7 GHz.
Nombre de cœurs du processeur entre 1 et 8.
Nombre d'années entre 80 et 10000.

Donc :

Nombre d'opération par seconde = vitesseProcesseur * 1000000 * nombreCoeur.

La solution est donc le logarithme népérien du nombre d'opérations par seconde plus le logarithme népérien des années, le tout sur le logarithme népérien du nombre de possibilités par caractères.