Comprendre le codage d'une carte de crédit . . quelques calculs faciles et illustratifs

Sujet «unplugged» (sans ordinateur)

Voilà une carte bleue, elle a un code à 4 chiffres, ce code confidentiel est envoyé par la banque par courrier confidentiel, et il ne faut surtout pas le noter ailleurs que . . dans le tréfond du cerveau de l'utilisateur. Lorsque la carte bleue est utilisée, seuls 3 essais sont possibles, ensuite la carte est . . désactivée ou confisquée.
Voyons quel est son niveau de sécurité.

-1- ***Taper le code au hasard !***

- 1.1 Avec ce code à 4 chiffres, combien y a t'il de combinaisons possibles ? Appelons N ce nombre.
  - Il y a donc Ok1 = 1/N chances (1 chance sur N) de taper au hasard le bon code et
  - Il y a Ko1 = (N-1)/N (N-1 chances sur N) de taper au hasard le mauvais code, c'est à dire Ko1 = 1 – Ok1 = 1 – 1 / N chances

    * Vérifier que le calcul entre Ok1 et Ko1 correspond et calculer numériquement ces nombres.
- 1.2 Combien de chances de taper le bon code au bout de 2 essais ?
  - Calculons d'abord le nombre de chances de taper le *mauvais* code: c'est

    Ko2 = le nombre de chances de taper le mauvais code au 1er essai x le nombre de chances de taper le mauvais code au 2eme essai

    * Calculer numériquement ce nombre
  - Le nombre de chances de taper le bon code est alors Ok2 = 1 – Ko2

    * Calculer numériquement ce nombre
  - * Calculer en utilisant la même méthode, combien il y a de chances de taper le bon code au bout de 3 essais.

    * Permettre d'essayer le code 3 fois augmente t'il de beaucoup les chances de taper le bon code au hasard ?
- 1.3 Mais . . si il était possible d'essayer son code plus de 3 fois, sans limite, que se passerait-il alors ?

  * Décrire précisément une méthode pour ``craquer´´ le code si il n'y avait pas de limite au nombre d'essais

  * Quel est le nombre maximum d'essais à effectuer pour ``craquer´´ le code ?

  * A votre avis: quel est le nombre d'essais à effectuer en moyenne pour craquer le code ?

  * A votre avis: quel est le plus efficace pour craquer le code: essayer au hasard ou . . essayer un par un ?

<u>Remarque</u>:
On suppose ici que les notions intuitives de probablité sont connues:
En particulierque Proba(oui) = 1 - Proba(non);
On ne mentionne volontairement pas la subtilité entre tirage avec et sans remise, mais celà pourrait être une piste qui montrerait d'ailleurs que la différence est numériquement . . négligeable !
Proba[craquer après K essais] = P(OkK) = 1 – ((N-1)/N))^K avec remise et
= 1 – (N-1)/N .. (N-K)/(N-K+1) sans remise

-2- Démonter la carte de crédit !

Pour vérifier que le code tapé est le bon . . il faut forcément stocker ce code sur la carte de crédit, sinon comment vérifier que le code essayé est le bon ? Mais alors . . il suffirait de ``démonter´´ la carte de crédit (c'est à dire aller lire sa mémoire) pour découvrir ce code et . . pirater la carte ! Et bien . . imaginons que nous puissions aller lire la mémoire de la carte de crédit . . il y a bien une information qui permet de véfifier si le code est le bon . . mais . . surprise ce n'est ***pas*** le code !! C'est une bonne surprise pour les banquiers[1] et une mauvaise

1 Si une carte de crédit est piratée, ce n'est pas le client qui doit payer la somme volée, il est assuré contre ce type de vol (et paye d'ailleurs cette assurance avec les autres frais liés à cette carte de crédit). C'est donc une companie d'assurance

surprise pour qui penser pouvoir pirater.

Essayons de comprendre comment ce code est . . crypté (c'est à dire ``rendu invisible´´) sur la carte de crédit.

- 2.1. Quelques idées pour crypter et décrypter le code.
  * Proposer au moins ***trois*** idées pour mémoriser sur la carte de crédit un nombre qui ne soit pas le code, ***mais*** que permette de retrouver le code par un petit calcul ou un algorithme:
    - Par exemple: on pourrait convenir de remplacer tous les 0 par des 1 et tous les 1 par des 0 .
      - mais si le code est ``2345´´ . . ça ne sert à rien ! Alors à vous de trouver mieux.
  - Dès que chaque groupe à trois idées, partager toutes les idées au niveau de la classe entière:
    * Est-ce que finalement les gens n'ont pas des idées similaires ?
  * N'est il pas alors facile de ``tester´´ ces idées jusqu'à trouver la bonne ?
    - Bien sûr avec une seule carte de crédit le ``pirate´´ n'a que 3 essais,
      - mais si il a pu voler beaucoup de cartes de crédit il va pouvoir tester toutes ces idées en ne faisant que 1 ou 2 essais pour chaque carte de crédit et . . dès qu'il trouvé la bonne soluttion il pourra . . toutes les utiliser.
      - Et surtout celui qui a fabriqué le mécanisme de cryptage ou le connait . . peut pirater toutes les cartes bleues du monde !
- 2.2 Inventer un cryptage sans décryptage.
  - Prenons le nombre à 4 chiffres qui correspond au code et (i) multiplions ce nombre par 10007 et (ii) ajoutons les 8 chriffres 2 à 2 pour redonner un code à 4 chriffres, puis (iii) recommençons l'étape (i) et (ii) 13 fois. On se retrouve avec un autre code à 4 chiffres . . que nous appelons le *résultat.*
    - Le résultat n'est sûrement pas le code initial !
    - Il sera bien difficle de retrouver le code initial à partir de ce résultat car notre calcul des étapes (i) et (ii) ne . . peut pas se faire ``à l'envers´´ !
  - On a donc fabriqué un ``encryptage´´ qui ne peut pas être décrypté, donc personne ne peut deviner le code à partir du résultat qui est mémorisé sur la carte:

    Code initial de la carte de crédit --encryptage --> Résultat indécryptable.

    que nous pouvons écrire avec des notations mathématiques

    encryptage(code) = résultat

    c'est à dire que la fonction d'encryptage appliqué au code donne le résultat.

    * Est-ce que pour un code donné son encryptage donne toujours le même résultat ?
    * Est-ce que le fait de connaître la méthode décrite en (i) (ii) et (iii) me permet de deviner le code à partir du résultat ?
    * Comment utiliser alors cette méthode pour vérifier le code d'une carte de crédit ?
    * Est-ce que plusieurs codes peuvent donner le même résultat ? Si c'est le cas . . que risque t'il de se passer ?
      - Pour vous aider: regarder ce qui se passe si vous additionnez les chiffres 2 à 2 de 12456789 et 21547698.
  - Supposons que nous vivions dans un monde où il soit possible de faire des mutliplications mais PAS de calculer des racines carrées:
    - Par exemple on peut calculer 1234^2 = 1234 * 1234 = 1522756 mais ``impossible´´ de deviner que la racine carrée de 1522756 est 1234.
    * Dans ce monde imaginaire expliquer comment encrypter un code pour donner un résultat

---

qui garantit le paiement. Ceux qui tentent de pirater une carte de crédit n'ont alors pas à faire face à des clients isolés . . mais aux puissances companies d'assurance des banquiers qui ont les tous les moyens de pister, découvrir et poursuivre en justice ceux qui croient pouvoir devenir des pirates numériques. De fait il y a bien moins de ``vols électroniques´´ que . . de bracage de banques et c'est à la fois l'informatique et . . une bonne organisation économique qui assure cette sécurité.

qui permette de *vérifier* si le code est exact sans jamais connaître explicitement le code.
* Avec cette méthode, dans ce monde imaginaire, est-ce que pour un code donné son encryptage donne toujours le même résultat ?
* Avec cette méthode, dans ce monde imaginaire, est-ce que plusieurs codes peuvent donner le même résultat ?

3 Application aux mots de passe dans les ordinateurs

- Lorsqu'il s'agit de taper son mot de passe pour entrer dans un ordinateur, impossible de limiter le nombre d'essai: car si quelqu'un se trompe . . tout est bloqué et ce serait ingérable de retourner au magasin faire ``débloquer´´ l'ordinateur, il faut donc alors se protéger contre un pirate qui essayerait tous les codes possibles . . surtout si ce pirate est un ``robot´´ c'est à dire un programme d'ordinateur qui essaye tous les codes possibles, qui énumère ces codes jusqu'à trouver le bon.

- 3.1 Compter les possibilités
  * Supposons que le code soit une suite de 6 chiffres : combien cela fait il de possibilités ?
  * Combien de temps faut il environ à un robot (qui peut essayer de environ 1000 codes en une seconde, à la vitesse usuelle de l'internet pour ce type d'échanges) pour énumérer tous ces codes ?
  * Combien de temps faut il environ à un ``cheval de troie´´ robot (un virus logiciel qui est dans votre ordinateur et qui peut donc faire des essais à la vitesse du processeur, c'est à dire 1 million de codes par seconde) pour énumérer tous ces codes ?
  * Supposons que le code soit un mot de 6 lettres sans aucun sens, donc une suite de 6 lettres de l'aphabet:
    Combien cela fait il de possibilités ? Combien de temps pour être énumérer par le même robot ?
  * Si au lieu de prendre un mot sans aucun sens, l'utilisateur choisi un mot qui a un sens, c'est à dire un des 10000 noms communs ou nom propre du dictionnaire:
    Combien de temps pour être énumérer par le même robot ? Par un cheval de troie ?
  * Supposons que le code soit une suite de 6 caractères (une des 26 lettres, des 10 chiffres ou des 32 signes de ponctuation du clavier !@#$%^&*()_+..)
    Combien cela fait il de possibilités ? Combien de temps pour être énumérer par le même robot ? Par un cheval de troie ?
  * Supposons finalement que le code soit une suite de 12 caractères au lieu de 6
    Combien cela fait il de possibilités ? Combien de temps pour être énumérer par le même robot ? Par un cheval de troie ?
    - Résumer les éléments précédents dans un petit tableau.
- 3.2 Donner des conseils aux utilisateurs des ordinateurs pour que leur mot de passe soit protégés
  * Utiliser des résultats précédents pour donner 3 conseils (à rédiger en une ligne chacun) aux utilisateurs des ordinateurs pour que leur mot de passe soit protégés: prendre un mot au hasard ? Utiliser uniquement des chiffres ? Choisir un mot qui ait du sens pour bien s'en souvenir ?
- 3.3 Donner des conseils pour faire la différence entre un utilisateur et un robot qui cherche à pirater un ordinateur
  - Bien évidemment aucun mécanisme de mot de passe au monde ne laisse un robot ou cheval de troie ``craquer´´ ses mots de passe ! Il faut à la fois ne pas gêner un utilisateur même étourdi qui cherche à se connecter et en pas donner de chance à un robot de pirater l'ordinateur.

  * Proposer au moins 3 conseils (à rédiger en une ligne chacun) aux programmeurs de mécanisme de mot de passe pour que le logiciel fasse automatiquement la différence entre un utilisateur et un robot sans jamais ``bloquer´´ l'utilisateur mais en faisant en sorte que le temps pour qu'un robot pirate l'ordinateur soit de plusieurs siècles[2]: En introduisant des délais ? En obligeant l'utilisateur à entrer des mots de passe suffisament compliqués ?

2 parmi les solutions ``canoniques´´ au problème précédent, on peut proposer que le mécanisme de mot de passe empêche de proposer plus de 1 mot de passe par seconde, et qu'à chaque essai infructueux le temps d'attente croisse; proposer aussi que le mot de passe soit choisi au hasard par l'ordinateur et non par l'humain, etc...