# Extraire des données d'une image ISO XBOX

*Documentation pour programmeurs*
*Arkeur*

## Sommaire :



## Avant-Propos :

Tout d'abord, voici la page avec laquelle j'ai créé EMI ; mon programme d'extraction : http://www.xbox-linux.org/docs/gdfs.html. C'est la meilleur page qui détaille précisément comment est construite une xIso. Lisez là posément, car chaque mot est une réponse à vos futurs questions…

Par ou commencer ? Déjà, avant de décortiquer un peu une xIso, on va voir ce que l'on veut faire dans la plus part des cas. C'est à dire tester si l'xIso que votre utilisateur aura choisi est bien une image iso au format Xbox. Puis par la suite, vous aurez le choix d'extraire directement votre xIso ou lire seulement les noms de fichiers/dossiers quelle contient afin de les présenter à l'utilisateur.

C'est ce que l'on va voir ensemble au fil de ce document. Mais avant toute chose, il va falloir passer par la case départ:

## Xbox-Linux.org !

Allez à cette page : http://www.xbox-linux.org/docs/gdfs.html Vous y êtes ?OK, alors j'explique…

Déjà il faut savoir qu'un support de données ça a des secteurs. Imaginez un peu les supports de données comme des commodes et les secteurs comme les tiroirs de cette commode. Naturellement, plus vous avez de tiroirs dans votre commode plus votre commode est grande… Et bien c'est le même principe avec les supports de données. Plus il y a de secteurs, plus le support de données est grand.

Un tiroir ça a une taille (grand, petit, moyen…). Un secteur aussi, et on exprime cette taille en octets (bytes en anglais). **Les secteurs d'un média Xbox font toujours 2048 octets.** Comme le dit aussi la documentation du site, il y a sur un média xbox toujours 8Mo de réservé à réparer les fichiers corrompus sur un CD/DVD. On nomme ça les « Security Holders ». Mais ça, c'est juste à titre informatif parce que nous ça ne nous sert pas à grand chose pour ce que nous voulons faire…

Microsoft a également modifié la TOC de ses médias afin de nous faire croire que ceux ci comportent beaucoup moins de données qu'ils n'en contiennent réellement. C'est à dire que si

vous insérer votre DVD Xbox dans votre lecteur DVD PC, il n'affichera pas toutes les données de celui-ci ; le but étant bien entendu de se préserver des copies pirates. C'est aussi pourquoi le firmware des lecteurs DVD Xbox est un peu spécial. Le firmware qu'ils contiennent arrivent à lire les DVD Xbox car ils n'ont pas le même firmware que les lecteurs PC.

Enfin, avant de commencer à parler en hexadécimal, d'offset et autres, **sachez que les informations qui nous intéressent pour extraire des fichiers/dossier commencent à l'octet 65536 *(0x10000 en Hexa)* de l'xIso.**

**On en déduit alors que du 1er octet jusqu'au 65536ème, il n'y que des 0 *(0x00 en hexa)*!** Cela veut dire que lorsque nous lirons notre média Xbox, nous commencerons à lire à partir de 0x10000. **Ou tout simplement à partit du 32ème secteur (32*2048).**

Toujours en suivant la documentation du site, passons au « **VOLUME DESCRIPTOR » :**
**C'est informations sont toujours présentes au 32ème secteur du média.**
Pour se déplacer dans le média, on utilise une adresse : l'offset.
**On se déplace toujours en avant !** Donc vos variables de type « integer » seront forcément de type Uint (Unsigned Integer)… Pour ceux qui ne comprennent toujours rien, ça veut dire que l'offset ne pourra jamais être négatif.

Voici la traduction du tableau du site :
*(je vous mets la représentation chiffrée en bleue à côté des valeurs hexadécimale)*

| OFFSET | Taille (octet) | Description | On en a besoin ?!! |
|---|---|---|---|
| 0x000 (0) | 0x014 (20) | Contient « MICROSOFT*XBOX*MEDIA ». C'est une constante. Cette chaîne de caractère est toujours présente dans une xIso. | Oui, pour tester si le média est un média Xbox. |
| 0x014 (20) | 4 (4) | Contient la table des secteurs du dossier principal … Personnellement, j'ai tjrs compris qu'à moitié à quoi ça sert. De toutes façons, il suffit de la lire et de récupérer sa valeur. | Oui, pour calculer l'emplacement des données. Pour le moment, j'ai toujours trouvé que la valeur était toujours égale à **264**. |
| 0x018 (24) | 4 (4) | Taille en octets de la table des secteurs du dossier principal | Non, pas pour ce qu'on veut faire |
| 0x1C (28) | 8 (8) | Structure FILETIME qui correspond à la date de création de l'xIso. Représentée au format FAT *(aussi bien pour la date que pour l'heure).* | Non, pas pour ce qu'on veut faire |
| 0x024 (36) | 0x7C8 (1992) | Non utilisé ? Ne contient que des 0. | Non, pas pour ce qu'on veut faire |
| 0x7EC (2028) | 0x14 (20) | Contient « MICROSOFT*XBOX*MEDIA ». C'est une constante. Cette chaîne de caractère est toujours présente dans une xIso. | Oui, car si à cet endroit il n'y a pas « MICROSOFT*XBOX*MEDIA », c'est que l'image Xbox est corrompue ! Donc, pas la peine de chercher plus loin… |

Note: Si le dossier principal de la table des secteurs est vide, le secteur et la taille contiendront 0.

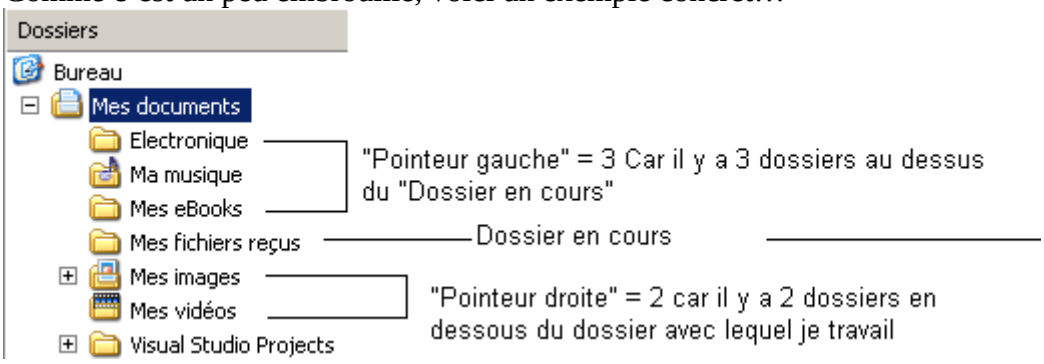
Allez, passons au « **DIRECTORY TABLE» :**

La table des répertoires contient la liste de toutes les entrées de dossiers/fichiers. Un espace est rempli avec 0xFF (255). La table peut contenir des secteurs contigus.

Bref, tout ça on n'en a pas besoin pour ce que l'on veut faire…

Maintenant, passons au « **DIRECTORY ENTRY» :**
Tout est stocké dans une arborescence binaire et structurée. Pour accélérer la vitesse de déplacement entre les secteurs, et les fichiers, Microsoft à mis **2 pointeurs pour chaque entrée** : Un « pointeur droite » et un « pointeur gauche » qui pointe sur les sous répertoires du dossier en cours. Le nombre des répertoires ayant un ordre alphabétique moins important que le dossier sur lequel on travail est stocké dans le « pointeur gauche ». Pour les répertoires classés alphabétiquement après le dossier sur lequel on travail, le nombre de ses dossiers est stockés dans « pointeur droite ».

Comme c'est un peu embrouillé, voici un exemple concret…




Le dossier en cours est « Mes fichiers reçus ». Mais rappelez-vous que l'on parle de « directories entries » donc la valeur des pointeurs peut correspondre au nombre de dossiers ou de fichiers. Pour savoir si c'est un fichier ou un dossier, on utilise **l'attribut du fichier : Si l'attribut de la prochaine entrée dans la table est égale à 16 alors le prochain type de donnée de la table sera un DOSSIER.**

**Voici le tableau tiré xbox-linux.org :**

| OFFSET | Taille (octet) | Description | On en a besoin ? |
|---|---|---|---|
| 0x0000(0) | 2 | C'est mon fameux « pointeur gauche ». C'est un DWORD | Oui, pour calculer l'emplacement du prochain fichier/dossier… Par contre, bizarre… Moi je lis ça avec un (UInt16) !! |
| 0x0002(2) | 2 | Idem mais pour mon « pointeur droite » | Oui, pour calculer l'emplacement du prochain fichier/dossier… Idem ! |
| 0x0004(4) | 4 | Début du secteur du fichier. (UInt64) | Oui, pour calculer l'emplacement du prochain fichier/dossier… |
| 0x0008(8) | 4 | Taille total du fichier (UInt32) | Oui, c'est les données du fichier ! |
| 0x000C(12) | 1 | Attributs du fichier (Uint32) ***(voir le tableau juste en dessous)*** | Oui afin de faire une image avec des fichiers identiques à ceux contenus dans l'xIso. |
| 0x000D(13) | 1 | Longueur du nom de fichier (Uint32) | Oui, pour récupérer le nom de fichier |
| 0x000E(14) | <Taille du nom de fichier> | C'est le nom de fichier stocké sur « X » caractères. X étant le nombre de caractères qui le compose. | Oui, c'est le nom du fichier/dossier !! |
| 0x000E (14)+ nom du fichier | <0-3> | Contient 255 (0xFF) jusqu'au prochain DWORD | -Je n'ai toujours pas compris à quoi sa sert. -Ca tombe bien, je n'en ai pas eu besoin ☺ |

- Si un sous répertoire est vide, son entrée décrite dans son répertoire parent est égale à 0.

Valeurs des attributs du fichier:

| Valeur | Désignation |
|---|---|
| 0x01 (1) | Fichier en lecture seule |
| 0x02 (2) | Fichier caché |
| 0x04 (2) | Fichier système |
| 0x10 (16) | Dossier (avec une directory table à l'intérieur) |
| 0x20 (32) | Fichier archivé |
| 0x80 (128) | Fichier normal… |

## Tester si le média est un « Média Xbox » :

Bon bah voilà, on a fait le plus compliqué ! Vous allez voir, le reste ça passe tout seul…
Maintenant qu'on a toutes les infos pour extraire les données d'une images xIso, on peut commencer à « programmer »…

Tout d'abord l'**Algorithme**. Il est simple et on va le faire ensemble. Pour y arriver du premier coup, je vais vous donner ma méthode…
J'ai ma fille à côté de moi, elle pleure un peu. Je sais que pour la calmer, il suffit qu'elle entende ma voix. Donc je vais lui raconter ce que doit faire le programme. Bien entendu, comme elle est petite, il faut **tout** lui détailler… Allons-y :

«- J'ai un gros fichier de plusieurs Giga que je voudrais mettre sur la console. Il contient une sauvegarde d'un répertoire et je veux le remettre sur le Xbox.

-Ce fichier s'appelle une Iso et plus couramment une xIso car elle a le format Xbox.

Quand mon utilisateur aura choisi l'xIso a extraire, le programme passera par 3 étapes :

1. **Je m'assure que l'image rentrée est bien une xIso**
2. Pour faire patienter l'utilisateur, le programme lui montrera ou il en est. **Pour cela, je vais compter combien il y a de fichiers/dossiers à extraire.**
3. **Le programme finira par extraire les données dans un dossier tout en affichant combien de dossier/fichiers il reste à extraire.**

**Comment tester si l'xIso est vraiment une xIso ?:**
Simple, il faut commencer à **lire le fichier en mode binaire.** Puis comme on veut juste le tester, on va faire 2 tests rapides :
1er test :

1- **On sait que tout média Xbox contient la chaîne de caractères « MICROSOFT*XBOX*MEDIA » sur le 32ème secteur de l'xIso.** Je vais donc :
2- Ouvrir mon xIso et me placer au 65536ème octets de l'xIso *(0x10000)* ; soit le 32ème secteur..
3- Je vais ensuite lire 20 octets *(0x014)* et placer le résultat dans un **buffer** de caractères.
4- Si mon **buffer** contient **« MICROSOFT*XBOX*MEDIA »** cela veut dire que le média est un média xIso. Sinon, il faut informer l'utilisateur qu'il doit choisir une autre image.

2ème test :
OK, arrivez ici le fichier semble être un média Xbox. Mais peut-être que l'image est corrompue ? Cela arrive souvent au cours d'un mauvais transfert…

Pour cela, on va faire un second test ; on va tester la fin de l'entête de ce média. Pour cela, on sait que quelques octets plus loin *(2008 exactement ou 0x7D8 pour les puristes…)* on va retrouver une seconde fois la constante **« MICROSOFT*XBOX*MEDIA ».** Ce test consistera simplement à lire les 2030 octets prochains et de savoir si les 20 derniers contienne notre constante…

1- Après le premier test notre curseur de fichier est positionné au 65556ème octet de l'xIso *(0x10000 + 0x014).*
2- On avance donc de 2008 octets *(0x7D8)* pour arriver directement sur la fin du header et se préparer à lire les 20 prochains octets.
**3- On vide notre buffer.** Et on le remplit en lisant 20octets *(0x014).*
4- Que contient **buffer** ? S'il contient **« MICROSOFT*XBOX*MEDIA »** alors l'xIso est valide sinon, il faut informer l'utilisateur que l'xIso est bien :
   - un média Xbox *(premier test OK)*
   - Mais que celle ci est corrompue *(second test faux).*

Une fois que nous avons fait tout ça, on peut passer au comptage facultatif des données à extraire… »

Et maintenant, ma fille est endormie… *(Normal, j'ai du la saouler avec mon histoire !)*

## Lister les fichiers et les dossiers de l'xIso :

Maintenant, que nous savons que notre xIso est un média Xbox valide, deux choix s'offrent à nous :

- On cherche la rapidité extrême et on attaque directement l'extraction des fichiers. Si tu es un de ceux là, saute directement au prochain chapitre !

- On cherche à faire un programme le plus « propre » possible, prenant en compte la curiosité de l'utilisateur. Dans ce cas, on fait un premier comptage des fichiers afin de montrer à l'utilisateur un pourcentage représentant le nombre de fichiers restants à extraire. On perd par contre quelques secondes car il y a une lecture de l'xIso en plus.

Pour compter les fichiers, il faut un algorithme. Pour extraire, il faut reprendre le même algorithme + Ecrire sur le disque à un moment donné. Ma fille se réveille et se remet à pleurer, voilà comment je la ré-endore…

« OK, maintenant que tu sais extraire une image xIso, je vais t'expliquer comment on fait un comptage de toutes les données à extraire.

Pour cela, il me faut 2 variables : **TOTAL_DOSSIERS** et **TOTAL_FICHIERS**.

Et comme tout bon programmeur, je pense à les initialiser à zéro avant de les utilisées. Pourquoi ? Parce que toutes variables dont le nom est à gauche **et** à droite du signe égal *(=)* doit être initialisées. Ce qui me fait :

**TOTAL_DOSSIERS = 0**
**TOTAL_FICHIERS = 0**

Ensuite, j'ai besoin d'autres variables :
**pDroit** qui sera mon « pointeur droite » *(voir chapitre sur Xbox-Linux.org)*
**pGauche** qui sera mon « pointeur gauche »
**secteur** qui contiendra l'adresse de mon prochain secteur
**taille** qui contiendra la taille de mon fichier
**attribut** qui contiendra les attributs de mon fichier
**longueur_nom_fichier** qui contient le nombre de caractères dans le nom de mon fichier
**nom_fichier** qui contiendra le nom de mon fichier
**dirTable** qui est expliqué au premier chapitre. Actuellement, je l'ai toujours vu égale à 264 !
Et **xIso** qui sera notre fichier ouvert en lecture sur notre xIso

Bien entendu, tout ça je ne l'ai pas inventé, je l'ai récupéré dans le tableau un peu plus haut… Mais il faut cependant respecter quelque chose : le type de ces données ! Et oui, car lire un « integer » et lire un « long » ce n'est pas la même chose…

Voici leur type :

| Nom de la variable | Type |
|---|---|
| TOTAL_DOSSIERS | Entier sur 32 bits (integer) |
| TOTAL_FICHIERS | Entier sur 32 bits (integer) |
| PDroit | Entier sur 16bits (short) |
| pGauche | Entier sur 16bits (short) |
| Secteur | Chaîne de caractères |
| Taille | Chaîne de caractères |
| Attribut | Chaîne de caractères |
| longueur_nom_fichier | Chaîne de caractères |
| nom_fichier | Chaîne de caractères |
| dirTable | Entier sur 64 bits (long) |
| xIso | Pointeur sur un fichier ouvert en lecture |

**A savoir !** secteur,taille,attribut et longueur_nom_fichier ne devrait pas être des Chaîne de caractères mais long, int, octet, octet. Manque de bol, chez moi ça marche pas alors je ne me suis pas cassé la tête, j'ai mis le tout en « string » et ça passe nickel…

Le meilleur moyen d'expliquer à un programmeur comment fonctionne un programme, c'est l'algorithme. Le voici !

**COMPTAGE :**

```
FONCTION COMPTEUR (xIso, offset , dirTable, xIsoFile)
            \*Récupération des informations:
            Lire xIso de OFFSET caractère en partant du début
            pGauche = Lire un entier de 16bits dans xIso
            pDroit = Lire un entier de 16bits dans xIso
            secteur = Lire un entier de 32bits dans xIso
            taille = Lire un entier de 32bits dans xIso
            attribut = Lire un octet dans xIso
            longueur_nom_fichier_fichier = Lire un octet dans xIso


            \*Test s'il y a un fichier:
            SI longueur_nom_fichier est différent de 0 ALORS
                \*Test le type de la donnée trouvée (fichier ou dossier) :
                SI attribut = 16 ALORS
                    \*C'est un dossier:
                    TOTALS_DOSSIERS = TOTALS_DOSSIERS + 1
                    \* On passe à la donnée suivante :
                    EXTRACTION(xIso, secteur * 2048, secteur)
                SINON
                    \*C'est un fichier :
                    TOTAL_FICHIERS = TOTAL_FICHIERS + 1
                FIN DE SI

                \* Récursivité dans le dossier ou l'on se trouve
                SI pDroit <> 0 ALORS
                    \* On passe à la donnée suivante :
                    EXTRACTION(xIso, (dirTable * 2048) + (pDroit * 4), dirTable)
                FIN DE SI

                SI pGauche <> 0 ALORS
                    \* On passe à la donnée suivante :
                    EXTRACTION(xIso, (dirTable * 2048) + (pGauche * 4), dirTable)
                FIN DE SI

           FIN DE SI \*Fin du test du fichier
FIN DE LA FONCTION COMPTEUR
```

Voilà ! Ce n'est pas très dur n'est-ce pas ?! Bien entendu, pas la peine de faire un copier/coller de cela, c'est juste un algorithme… Vous savez désormais que vous aurez **TOTAL_DOSSIERS** et **TOTAL_FICHIERS** à extraire.

## L’extraction :

Vous en voulez encore ?!! Bon, et bien passons au plus facile… Vous savez compter combien il y a de fichiers/dossiers dans l’xIso ? Soit ! Alors on va recompter… Mais au passage, du TOTAL_FICHIERS = TOTAL_FICHIERS + 1, on va écrire ce fichier et créer le répertoire lorsque l’on passera sur TOTAL_DOSSIERS = TOTAL_DOSSIERS + 1.

Les plus logiques d’entre vous utiliseront sûrement la même fonction en rajoutant à celle-ci un paramètre booléen « comptage » afin de savoir si l’on veut juste compter ou extraire la donnée…

Maintenant, transformons notre fonction de comptage en véritable extracteur de données xIso…

*Avant tout, il faut initialiser nos variables soit :
**pDroit , pGauche** sont des int16
**secteur, dirTable** sont des int64
**taille** est un int32
**attribut** est un octet
**longueur_nom_fichier** est un octet
**nom_fichier** est une chaîne de caractères
**xIso** est un pointeur sur l'xIso qu'a choisi notre utilisateur
**entête** est une chaîne de caractère *(pour tester « MICROSOFT*XBOX*MEDIA »*
**TOTAL_FICHIERS**, **TOTALS_DOSSIERS** sont des int32
**Comptage** est un booléen

*Ensuite, il faut ouvrir notre xIso pour la lire :
**Ouvrir xIso en Lecture**

*Maintenant, on test si c'est un média Xbox !
***TEST_MEDIA:**

```
Avancer au  32ème secteur de xIso en partant du début du fichier
entête = lire 20 octets dans xIso

If entête est différent de "MICROSOFT*XBOX*MEDIA" ALORS
   AFFICHER « Votre xIso n'est pas un média Xbox. »
   FIN DU PROGRAMME
FIN DE SI
```

*Puisqu'on arrive ici, c'est que l'xIso est valide. On test donc son entête et on récupère au *passage la valeur de la table des répertoires. On met cette valeur dans « dirTable ».
*Personnellement, j'ai toujours trouvé que cette valeur était égale à 264…

```
Avancer de 20 octets dans xIso
\*On récupère dirTable :
dirTable = lire 2 octets dans xIso

\*On passe les infos qui ne nous servent pas…
Avancer de 2004 octets dans xIso

\*On vérifie si l'image du média est corrompue :
entête = lire 20 octets dans xIso
If entête est différent de "MICROSOFT*XBOX*MEDIA" ALORS
   AFFICHER « Votre xIso est un média Xbox corrompue ! »
   FIN DU PROGRAMME
FIN DE SI

\*On initialise nos variables de compteurs :
TOTAL_DOSSIERS = 0
TOTAL_FICHIERS = 0
\*Maintenant, on a le choix :
   -  soit on compte le nombre de fichiers et de dossiers et donc on
      appelle la fonction d'extraction de cette façon :
   EXTRACTION (xIso,dirTable *2048, dirTable, vrai)
   -  soit on extrait directement les données :
   EXTRACTION (xIso,dirTable *2048, dirTable, faux)
\* On pense à fermer notre fichier :
FERMER xIso
FIN DU PROGRAMME.
```

***EXTRACTION:**

```
FONCTION EXTRACTION (xIso, offset , dirTable, xIsoFile, comptage)
            \*Récupération des informations:
            Lire xIso de OFFSET caractère en partant du début
            pGauche = Lire un entier de 16bits dans xIso
            pDroit = Lire un entier de 16bits dans xIso
            secteur = Lire un entier de 32bits dans xIso
            taille = Lire un entier de 32bits dans xIso
            attribut = Lire un octet dans xIso
            longueur_nom_fichier_fichier = Lire un octet dans xIso
            nom_fichier = lire longueur_nom_fichier_fichier de caractères dans xIso

            \*Test s'il y a un fichier:
            SI longueur_nom_fichier est différent de 0 ALORS
                \*Test le type de la donnée trouvée (fichier ou dossier) :
                SI attribut = 16 ALORS
                    \*C'est un dossier:
                    TOTALS_DOSSIERS = TOTALS_DOSSIERS + 1

                    Si comptage = vrai ALORS
                       \*Création du dossier:
                       On créé le répertoire "nom"
                       TOTAL_DONNEES_EXTRAITES = TOTAL_DONNEES_EXTRAITES + 1
                       On change le répertoire parent par "nom"

                       \* On informe l'utilisateur de ce que l'on fait:
                       AFFICHER "[TOTAL_DONNEES_EXTRAITES/TOTAL_DOSSIERS+TOTAL_FICHIERS]:"
                       AFFICHER "Le répertoire "nom" a été créé."
                    FIN DE SI

                    \* On passe à la donnée suivante :
                    EXTRACTION(xIso, secteur * 2048, secteur)

                    Si comptage = vrai ALORS
                       \*Il n'y a plus de sous-dossiers,
                       \*on revient au répertoire parent:
                       On change le répertoire parent par ".."
                    FIN DE SI
                SINON
                    \*C'est un fichier :
                    TOTAL_FICHIERS = TOTAL_FICHIERS + 1


                    Si comptage = vrai ALORS
                      \*On créé le fichier et on l'extrait:
                       ECRIRE dans "nom" "taille" octets.
                       TOTAL_DONNEES_EXTRAITES = TOTAL_DONNEES_EXTRAITES + 1

                       \* On informe l'utilisateur de ce que l'on fait:
                       AFFICHER "[TOTAL_DONNEES_EXTRAITES/TOTAL_DOSSIERS+TOTAL_FICHIERS]:"
                       AFFICHER "Le fichier "nom" a été extrait.
                    FIN DE SI
                FIN DE SI

                \* Récursivité dans le dossier ou l'on se trouve
                SI pDroit <> 0 ALORS
                    \* On passe à la donnée suivante :
                    EXTRACTION(xIso, (dirTable * 2048) + (pDroit * 4), dirTable)
                FIN DE SI

                SI pGauche <> 0 ALORS
                    \* On passe à la donnée suivante :
                    EXTRACTION(xIso, (dirTable * 2048) + (pGauche * 4), dirTable)
                FIN DE SI

            FIN DE SI \*Fin du test du fichier
FIN DE LA FONCTION EXTRACTION
```

Voilà ! Vous voyez, extraire des données d'un média Xbox ne relève pas de la sorcellerie… Après, vous pouvez améliorez votre fonction en testant la valeur attribut et appliquer cette attribut lorsque votre fichier est créé sur le disque.

Sinon, juste un petit conseil, passez par des buffers multiples de la taille des secteurs de votre système de fichiers pour écrire votre données sur le disque. Je m'explique…
Imaginons que vous êtes sous un système de fichier NTFS. Vos secteurs font donc 4096octets. Donc vous déclarez la taille de voter buffer comme un multiple de 4096.

Pourquoi ? Parce que la tête de lecture de votre disque dur se déplacera moins et sera donc plus rapide… Il faut savoir qu'il est plus rapide de lire secteur par secteur un support de données que par « bouts de secteur » ou autres… ;-)

Voilà ! C'est fini pour l'extraction ! Mais je vous laisse par la même occasion sur le chemin de la création des xIso… Ce n'est pas dur, vous avez toutes les clés en mains dans ce document !

Bien entendu, je ne suis pas du genre à pondre un How-To et me cacher… s'il y a des erreurs, des fautes ou tout simplement des remarques *(en bien ou en mal même si je préfère les biens !)* je suis à votre disposition…

Vous trouverez comment me contacter sur le site : http://Arkeur.ath.cx/xbox

Sur ce, à vous de jouer !
**Arkeur**