

**Partiel (Sujet 1) 2022-2023 - CYBER1 (2h00)**  
**Architecture des Ordinateurs**

NOM :

PRÉNOM :

Vous devez respecter les consignes suivantes, sous peine de 0 :

- Lisez le sujet en entier avec attention
- Répondez sur le sujet
- Ne détachez pas les agrafes du sujet
- Écrivez lisiblement vos réponses (si nécessaire en majuscules)
- Les appareils électroniques sont tous interdits (calculatrices également)
- Ne trichez pas

**1 Questions (8 points)****1.1 (1 point) Rappelez les 14 premières puissances de 2 :**

$2^0$	$2^1$	$2^2$	$2^3$	$2^4$	$2^5$	$2^6$	$2^7$	$2^8$	$2^9$	$2^{10}$	$2^{11}$	$2^{12}$	$2^{13}$

**1.2 (4 points) Convertissez ces nombres en décimaux. Vous donnerez leur interprétation non-signée puis signée.**

	non-signé	signé
% 1101 0101 0110		
% 1010 0001 1100		
\$ 9BD		
\$ A66		

1.3 (3 points) Convertissez ces nombres décimaux en binaire sur 12 bits, puis en hexadécimal.

	binaire	hexadécimal
42		
4321		
−116		

1.4 (2 points) Convertissez ces nombres en flottants au format IEEE 754 simple précision :

	exposant										hexadécimal							
68,78125	%										\$							
−218,3203125	%										\$							



## 2 Problème (10 points)

Afin de vous plonger réellement dans le *forensic* (analyse forensique ou investigation numérique en français), une toute petite FAT12 a été réalisée et utilisée pour produire quelques fichiers et dossiers contenant quelques mots. L'objectif est de retrouver ce contenu.

Cet exercice a été réalisé avec une série de commandes que vous pourrez réaliser de votre côté pour également observer l'évolution d'une petite FAT12 (voir **truncate(1)**, **mkfs(1)** et **mount(1)**).

### Description du système de fichiers FAT

Le format FAT (*File Allocation Table*) est relativement simple, mais de nombreux champs dans les structures ainsi que des régions ne seront pas utiles dans notre cas. Une partition formatée en FAT12 ou 16 se divise en quatre régions :

- le secteur de boot contenant le BPB (BIOS Parameter Block)
- une table d'allocation de fichiers (FAT en anglais), et sa copie de sauvegarde
- la région du dossier racine (cette région n'existe pas en FAT32)
- la région du contenu des fichiers et dossiers

Région réservée (boot & paramètres)	FAT	Dossier racine	Données (contenu fichiers & dossiers)
--	-----	-------------------	--

La FAT (la deuxième région) est un tableau dont chaque numéro de case correspond à un cluster.

La valeur dans la case indique que le cluster est :

- libre (0x000)
- abîmé/*bad cluster* (0xFF7)
- alloué (valeurs entre 0x002 et 0xFF5, ainsi que 0xFFF)

Pour chaque entrée de 12 bits dans la FAT :

- les clusters 3 et 7 sont libres,
- les clusters 2 et 4 et 5 sont utilisés,
- le cluster 6 est abîmé.

...	
FAT[2]	0x005
FAT[3]	0x000
FAT[4]	0xFFFF
FAT[5]	0x004
FAT[6]	0xFF7
FAT[7]	0x000
...	

FAT exemple :

...
00 50 00
FF F0 04
FF 70 00
...

Un fichier volumineux est stocké dans plusieurs clusters liés entre eux dans la FAT, comme une liste chaînée. Par exemple, en lisant les clusters 2, 5, puis 4 dans cet ordre précis, on obtiendrait le contenu d'un fichier. Le cluster 2 renvoie vers le 5, qui renvoie vers le 4, qui indique la fin (0xFFFF).

Jusque là, nous avons vu comment retrouver le contenu des données stockées dans une partition formatée en FAT12. Pour identifier des fichiers et des dossiers parmi les données, il est nécessaire d'étudier les *directory entries* (ou *direntry*) : des structures de données contenant les caractéristiques des objets stockés dans la partition.

Un dossier est littéralement un tableau contenant des structures de données décrivant chacune un fichier ou un dossier. Il y a donc autant de cases dans le tableau qu'il y a de fichiers et dossiers contenus à ce niveau hiérarchique.

```
dossier/
dossier/fichier1
dossier/fichier2
dossier/fichier3
```

```
struct direntry  entries[3]; // dossier
entries[0];      // dossier/fichier1
entries[1];      // dossier/fichier2
entries[2];      // dossier/fichier3
```

La structure représentant une *dirent* est de la forme suivante. On notera que FAT12 est très limité, ainsi, les fichiers ont des noms de 11 caractères maximum (8 avant l'extension, et 3 après). À noter : la taille enregistrée dans le champs *size* est en octets.

```
struct dirent {
    char[11] name;
    char      attributes;
    char[14] reserved_and_dates;
    int       first_cluster;
    long      size;
} __attribute__((packed))
```

Les types de données font :

char : 1 octet (8 bits)  
int : 2 octets (16 bits)  
long : 4 octets (32 bits)

Attributs :

ATTR\_READ\_ONLY 0x01  
ATTR\_HIDDEN 0x02  
ATTR\_SYSTEM 0x04  
ATTR\_VOLUME\_ID 0x08  
ATTR\_DIRECTORY 0x10  
ATTR\_ARCHIVE 0x20

Taille en octets d'une *dirent* : \_\_\_\_\_ (0,5 pts)

Pour vous aider à retrouver les chaînes de caractères, une table ASCII décimale/caractères est fournie :

Dec	10	13	32	45	46		48	49	50	51	52	53	54	55	56	57
Char	\n	\r	(espace)	-	.		0	1	2	3	4	5	6	7	8	9

Dec	65	66	67	68	69	70	71	72	73	74	75	76	77
Char	A	B	C	D	E	F	G	H	I	J	K	L	M

Dec	78	79	80	81	82	83	84	85	86	87	88	89	90
Char	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Dec	97	98	99	100	101	102	103	104	105	106	107	108	109
Char	a	b	c	d	e	f	g	h	i	j	k	l	m

Dec	110	111	112	113	114	115	116	117	118	119	120	121	122
Char	n	o	p	q	r	s	t	u	v	w	x	y	z

## 2.1 (2 points) Première étape : séparation des champs du dossier racine

Le dossier racine (ou *root directory* en anglais) est le premier dossier dans lequel des fichiers et des dossiers peuvent se trouver. La région du dossier racine contient en réalité une *dirent* avec quelques valeurs spécifiques.

En lisant la région du dossier racine, on extrait les données suivantes. Recopiez les différents champs dans le tableau associé, sans les interpréter pour le moment.

```
53 45 43 52 45 54 20 20
54 58 54 20 00 AB 6E 60
2C 56 2C 56 00 00 6E 60
2C 56 03 01 1F 00 00 00
56 4F 49 54 55 52 45 53
20 20 20 10 00 00 76 60
2C 56 2C 56 00 00 76 60
2C 56 04 02 00 00 00 00
```

	dirent[0] (f1)						dirent[1] (f2)					
name												
attributes												
first_cluster												
size												

## 2.2 (2 points) Deuxième étape : conversion des champs

Maintenant que vous avez extrait les champs, il est nécessaire de les convertir pour obtenir des valeurs interprétables. Convertissez de l'hexadécimal vers le décimal pour retrouver les caractères.

Concernant les noms de fichiers, les normes FAT12 et FAT16 précisent que les 8 premiers caractères servent à coder le nom du fichier, et les 3 derniers servent à coder l'extension. Si un caractère correspond à un espace, laissez sa case vide.

direntry[0] (f1)	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	.	<div></div>	<div></div>	<div></div>
direntry[1] (f2)	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	<div></div>	.	<div></div>	<div></div>	<div></div>

Retrouvez maintenant les attributs de chaque direntry, puis convertissez les entiers en valeurs décimales.

Avant de convertir les entiers, il faut savoir qu'en FAT, les entiers sont codés en *little endian* (*petit boutiste* en français), c'est-à-dire que les octets sont écrits au fur et à mesure du plus petit poids au plus grand. Ainsi, si on écrivait « 1337 » en little endian par paquets de un chiffre, on écrirait « 7331 », car 7 a le poids le plus petit (celui des unités), et 1 a le poids le plus fort (celui des milliers). Pour effectuer les conversions d'entiers, vous devrez donc inverser l'ordre de lecture des octets avant de les convertir (ainsi, « BE 3F » doit être interprété comme « 3F BE » avant d'être converti en décimal, « AB CD EF » doit être interprété comme « EF CD AB », et ainsi de suite).

	taille (en octets)	numéro du premier cluster	attributs			
direntry[0] (f1)			<input type="checkbox"/> Read Only	<input type="checkbox"/> Hidden	<input type="checkbox"/> Volume ID	<input type="checkbox"/> System
			<input type="checkbox"/> Directory	<input type="checkbox"/> Archive		
direntry[1] (f2)			<input type="checkbox"/> Read Only	<input type="checkbox"/> Hidden	<input type="checkbox"/> Volume ID	<input type="checkbox"/> System
			<input type="checkbox"/> Directory	<input type="checkbox"/> Archive		

## 2.3 (1 point) Troisième étape : lecture d'un fichier

L'une des direntry précédente a un nom particulièrement intéressant, et il est clair qu'une information importante se trouve dans le cluster pointé. Voici les données extraites du cluster dont il est question. Convertissez le message contenu dans le fichier, mais n'oubliez pas de vous arrêter à la taille en octets indiquée par la direntry associée (c'est-à-dire f1). Faites attention à la casse, c'est-à-dire aux majuscules et minuscules lorsque vous écrirez votre réponse.

4C	65	20	73	65	63	72	65
74	20	61	20	63	68	65	72
63	68	65	72	20	65	73	74
20	45	50	49	54	41	0A	54
4F	50	20	65	63	6F	6C	65
00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00


## 2.4 (5 points) C'est reparti...

Le message récupéré semble parfaitement clair, mais c'est trop simple pour être la réponse attendue. La deuxième direntry dispose peut être de la réponse. Le cluster pointé par celle-ci renvoie ces données, remplissez la structure qui devrait logiquement lui être adjointe.

```

2E 20 20 20 20 20 20 20
20 20 20 10 00 60 B0 4C
31 56 31 56 00 00 B0 4C
31 56 04 02 00 00 00 00
2E 2E 20 20 20 20 20 20
20 20 20 10 00 60 B0 4C
31 56 31 56 00 00 B0 4C
31 56 00 00 00 00 00 00
4D 31 20 20 20 20 20 20
54 58 54 20 00 06 BD 4C
31 56 31 56 00 00 BD 4C
31 56 05 02 0C 00 00 00
4D 32 20 20 20 20 20 20
54 58 54 20 00 BB C2 4C
31 56 31 56 00 00 C2 4C
31 56 06 02 0A 00 00 00

```

	direntry[0] (f11)						direntry[1] (f12)					
name												
attributes												
first_cluster												
size												

	direntry[2] (f13)						direntry[3] (f14)					
name												
attributes												
first_cluster												
size												

direntry[0] (f11)



direntry[1] (f12)

direntry[2] (f13)

direntry[3] (f14)

	taille (en octets)	numéro du premier cluster	attributs			
direntry[0] (f11)			<input type="checkbox"/> Read Only	<input type="checkbox"/> Hidden	<input type="checkbox"/> Volume ID	<input type="checkbox"/> System
direntry[1] (f12)			<input type="checkbox"/> Directory	<input type="checkbox"/> Archive	<input type="checkbox"/> Read Only	<input type="checkbox"/> Hidden
direntry[2] (f13)			<input type="checkbox"/> Volume ID	<input type="checkbox"/> System	<input type="checkbox"/> Directory	<input type="checkbox"/> Archive
direntry[3] (f14)			<input type="checkbox"/> Read Only	<input type="checkbox"/> Hidden	<input type="checkbox"/> Volume ID	<input type="checkbox"/> System
			<input type="checkbox"/> Directory	<input type="checkbox"/> Archive	<input type="checkbox"/> Read Only	<input type="checkbox"/> Hidden

Avec toutes les méta-données réunies jusqu'à maintenant concernant f1, f2, f11, f12, f13, f14, que déduisez-vous à propos de f11 et f12? (observez particulièrement les numéros de clusters)

Finalement, il reste encore deux derniers clusters à convertir. Attention à la taille des données, ainsi qu'aux majuscules et minuscules.

```
56 65 67 61 2D 4D 69 73
73 79 6C 0A 6E 31 0A 00
```


```
43 68 6F 75 70 65 74 74
65 0A 00 00 00 00 00 00
```


Les commandes ayant permis de produire et utiliser la FAT sont les suivantes :

```
touch Disk.img
truncate Disk.img -s 50K
# FAT12, secteurs de 512o, 1 secteur par cluster
mkfs.vfat -F12 -S512 -sl Disk.img
sudo mount Disk.img /mnt
### ... ###
cd /
sudo umount /mnt
```

Cependant, la plus petite FAT possible avec les outils linux reste une FAT12 de 50 kilo-octets, c'est-à-dire, toujours trop pour tenir sur une ou deux feuilles A4 en examen. N'oubliez pas d'utiliser *ghex* (ou d'autres éditeurs hexadécimaux) régulièrement pour observer les changements d'état de la FAT lorsque vous expérimenterez chez vous.

## SUJET 1