Ver también documento Nivel 7 de la práctica

En un sistema de ficheros tipo UNIX, en donde los inodos tienen dos punteros directos y uno indirecto simple, conocemos el contenido de los punteros de los siguientes inodos:

Número inodo	1	5	7	140	150	20	33
Punteros Directos	125	93	15	30	3	19	80
	120	20	18	37	32	22	75
Puntero Indirecto	150	10	NULL	28	NULL	33	45

También conocemos el contenido de los siguientes bloques de datos:

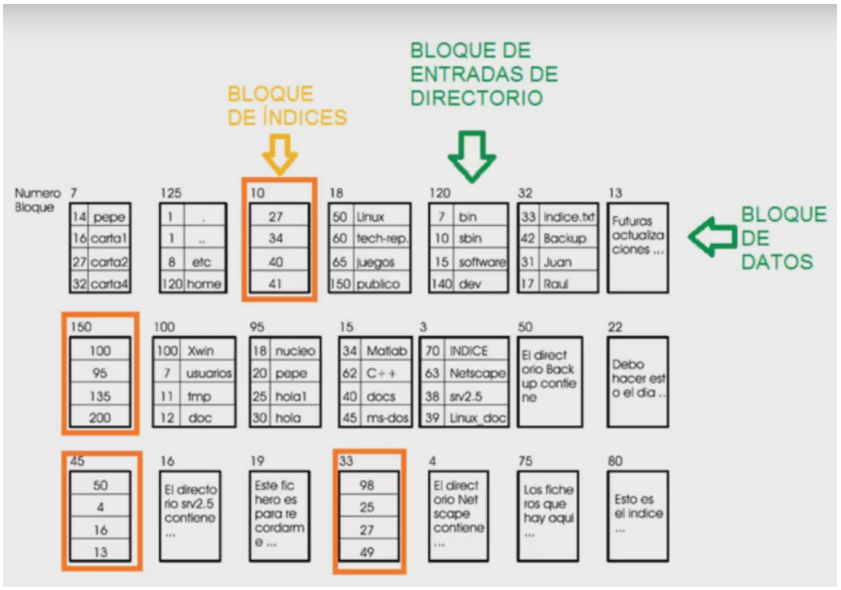
Numero Bloque	7		125		10	18		120		32		13
	14	pepe	1	.	27	50	Linux	7	bin	33	indice.txt	Futuras actualizaciones ...
	16	carta1	1	..	34	60	tech-rep.	10	sbin	42	Backup	
	27	carta2	8	etc	40	65	juegos	15	software	31	Juan	
	32	carta4	120	home	41	150	publico	140	dev	17	Raul	

	150		100		95		15		3		50	22
	100		100	Xwin	18	nucleo	34	Matlab	70	INDICE	El directorio Backup contiene	Debo hacer esto el dia ..
	95		7	usuarios	20	pepe	62	C++	63	Netscape		
	135		11	tmp	25	hola1	40	docs	38	srv2,5		
	200		12	doc	30	hola	45	ms-dos	39	Linux_doc		

	45	16	19	33	4	75	80
	50	El directorio srv2.5 contiene ...	Este fichero es para recordarme ...	98	El directorio Netscape contiene ...	Los ficheros que hay aqui ...	Esto es el indice ...
	4			25			
	16			27			
	13			49			

a) Explicar detenidamente como se encontraría el contenido del fichero */usuarios/publico/indice.txt* partiendo del directorio raíz.

Observemos primero los diferentes tipos de bloques de la zona de datos: bloques de índices (punteros), bloques de datos y bloques de entradas de directorio



Todos los tipos de bloques de la **zona de datos** deberían ocupar la misma cantidad de bytes: BLOCKSIZE[1]:

- Bloques de datos: unsigned char buffer [BLOCKSIZE]
- Bloques de punteros: unsigned int punteros[BLOCKSIZE/sizeof(unsigned int)]
- Bloques de entradas de directorio: struct entrada entradas[BLOCKSIZE/sizeof(struct entrada)] , siendo en nuestro caso:

```
struct entrada {
  char nombre[60];
  unsigned int ninodo;
};
```

La ruta+nombre que nos han dado es: **/usuarios/publico/indice.txt**.

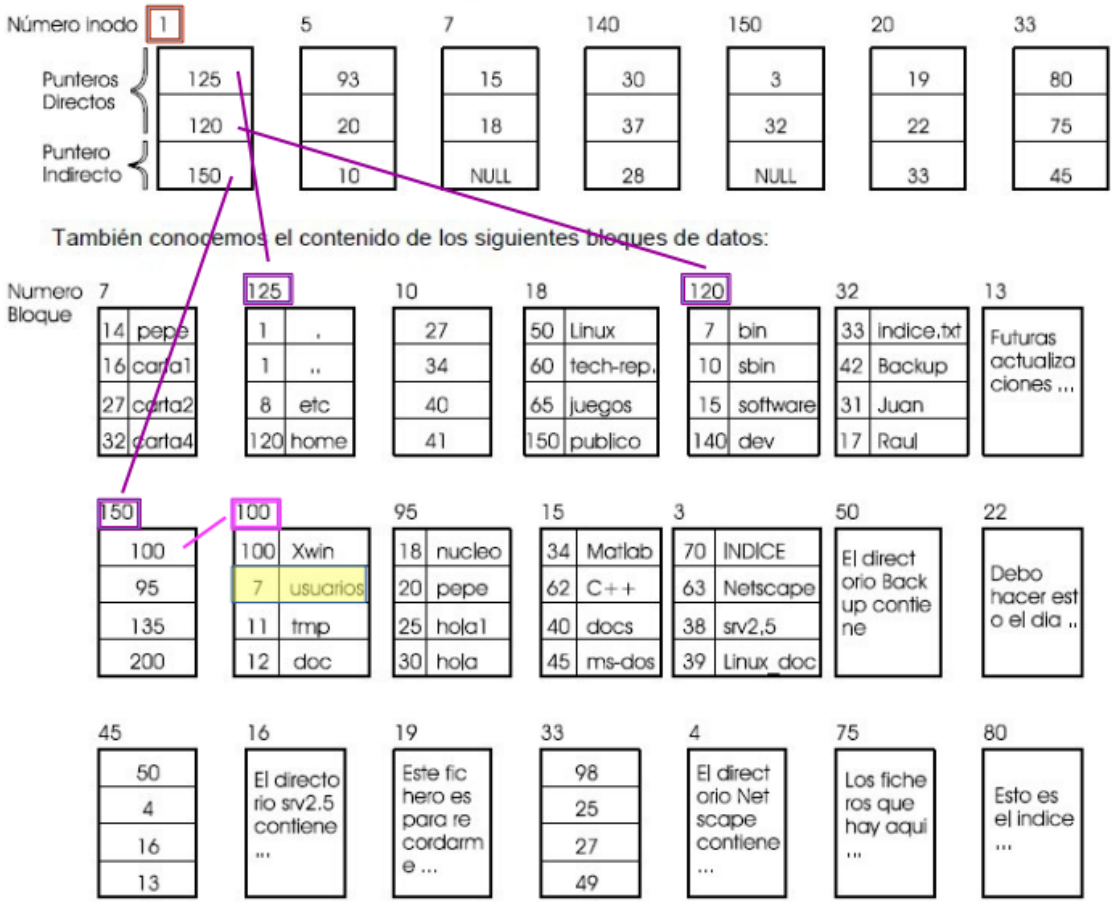
Empezamos recorriendo el inodo raiz (**en ese sistema de ficheros es el inodo 1**) que es un inodo de tipo directorio y por tanto su contenido son entradas. Buscamos el primer directorio, "usuarios", dentro de las entradas del directorio raíz. Es un recorrido **secuencial** de todas sus entradas (que pueden ocupar más de un bloque lógico) hasta que la hayamos encontrado o hayamos llegado a la última entrada.

- Empezamos por el primer bloque lógico del directorio raíz. El BL 0 del inodo 1 está en el BF 125 (eso lo determinó la función traducir_bloque_inodo() cuando se escribió y ahora la utilizamos para consultarlo). Comparamos los nombres de las entradas de ese bloque. Como no la hemos encontrado, pasamos al siguiente bloque lógico.
- El BL 1 del inodo 1 está en el BF 120. Comparamos los nombres de las entradas de ese bloque. Como no la hemos encontrado, pasamos al siguiente bloque lógico, pero en este

[1] En este ejemplo BLOCKSIZE=256, sizeof(struct entrada)= 64, así que tenemos 4 entradas de directorio por bloque. Teniendo en cuenta que todos los bloques han de tener el mismo tamaño, los bloques de punteros deberían contener 256/4 = 64 elementos, pero por simplicidad del ejemplo, para hacer el seguimiento gráfico de cómo se buscan las entradas, se han considerado bloques de 4 punteros.

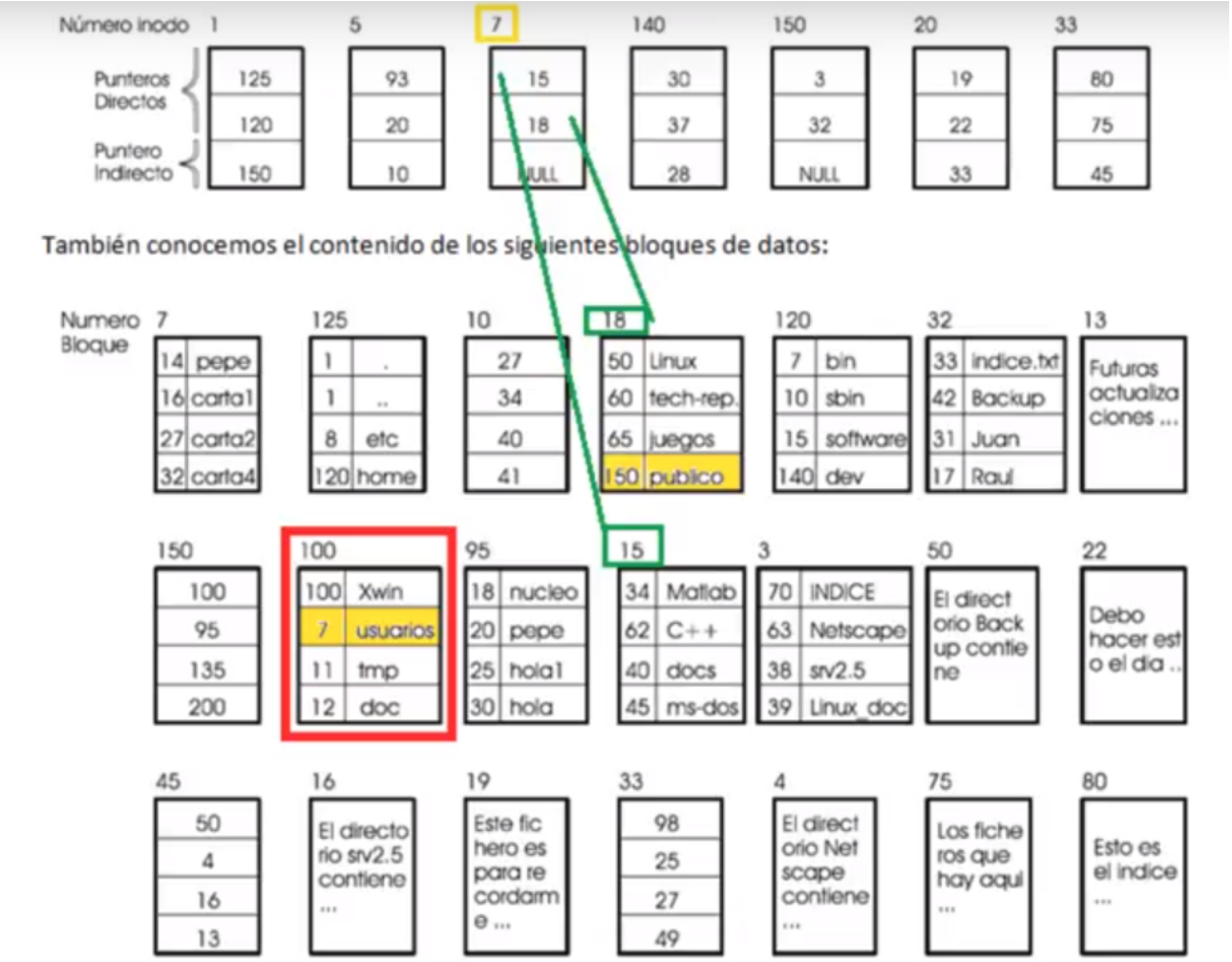
caso, el siguiente campo del inodo no es un puntero directo, sino un puntero indirecto que nos apunta a un bloque de punteros que está en el BF 150.

- Leemos ese bloque y nos indica que hemos de leer el BF 100 para obtener el correspondiente bloque lógico (nuestra función de traducir_bloque_inodo() ya se encarga de ir atravesando los niveles de punteros que sean necesarios para proporcionarnos finalmente el bloque físico de datos). Leemos sus entradas y encontramos una cuyo nombre es “usuarios” y nos indica que su inodo es el 7.



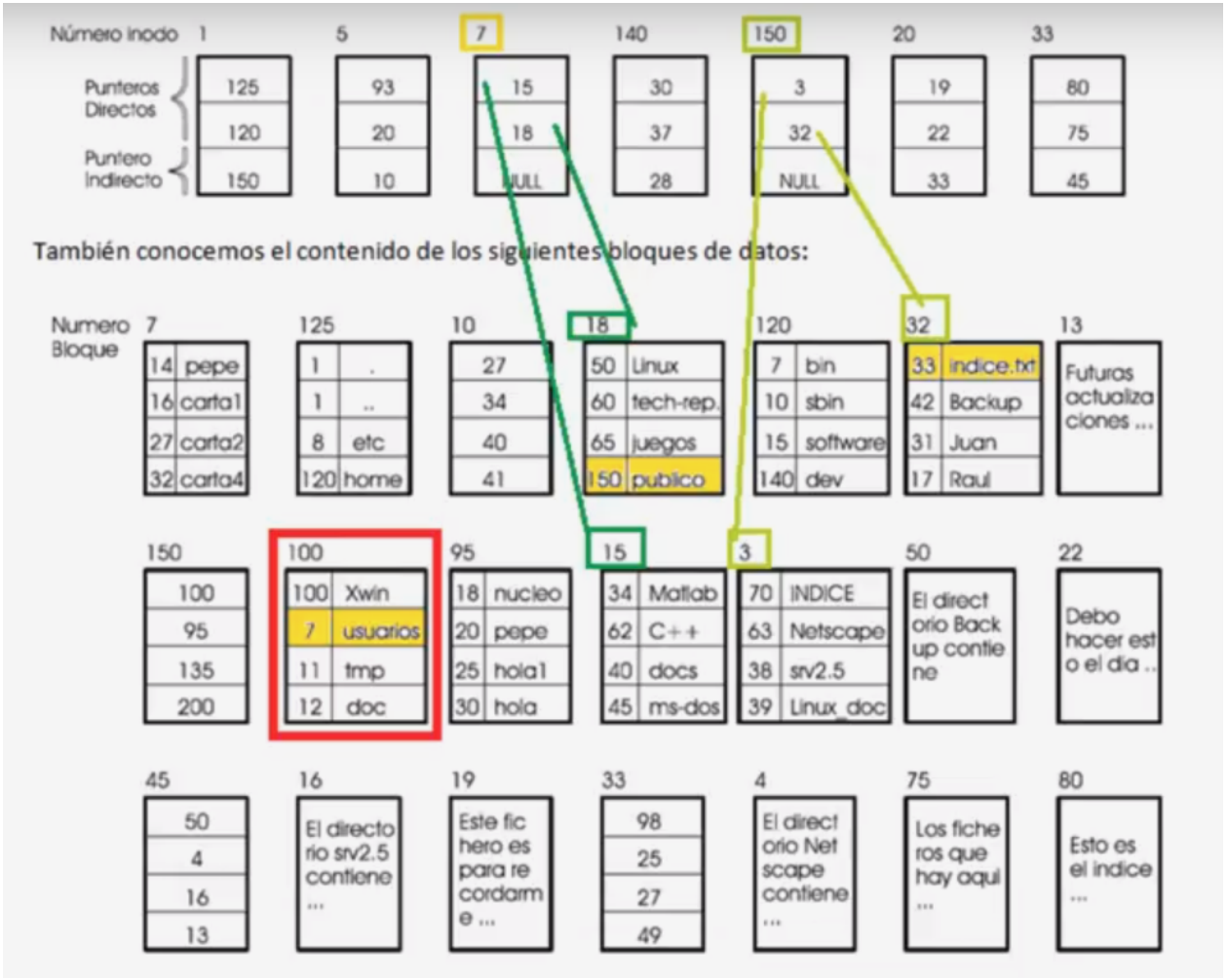
Recordemos que nuestra ruta+nombre es: /usuarios/publico/indice.txt.

Ahora hemos de recorrer el inodo 7, correspondiente a “usuarios”, buscando “publico” dentro de sus entradas. Empezamos por el primer bloque lógico del directorio usuarios. El BL 0 del inodo 7 está en el BF 15 (a nosotros nos lo devolverá la función traducir_bloque_inodo()). Comparamos los nombres de las entradas de ese bloque. Como no la hemos encontrado, pasamos al siguiente bloque lógico. El BL 1 del inodo 7 está en el BF 18. Comparamos los nombres de las entradas de ese bloque y encontramos una cuyo nombre es “publico” y nos indica que su inodo es el 150.

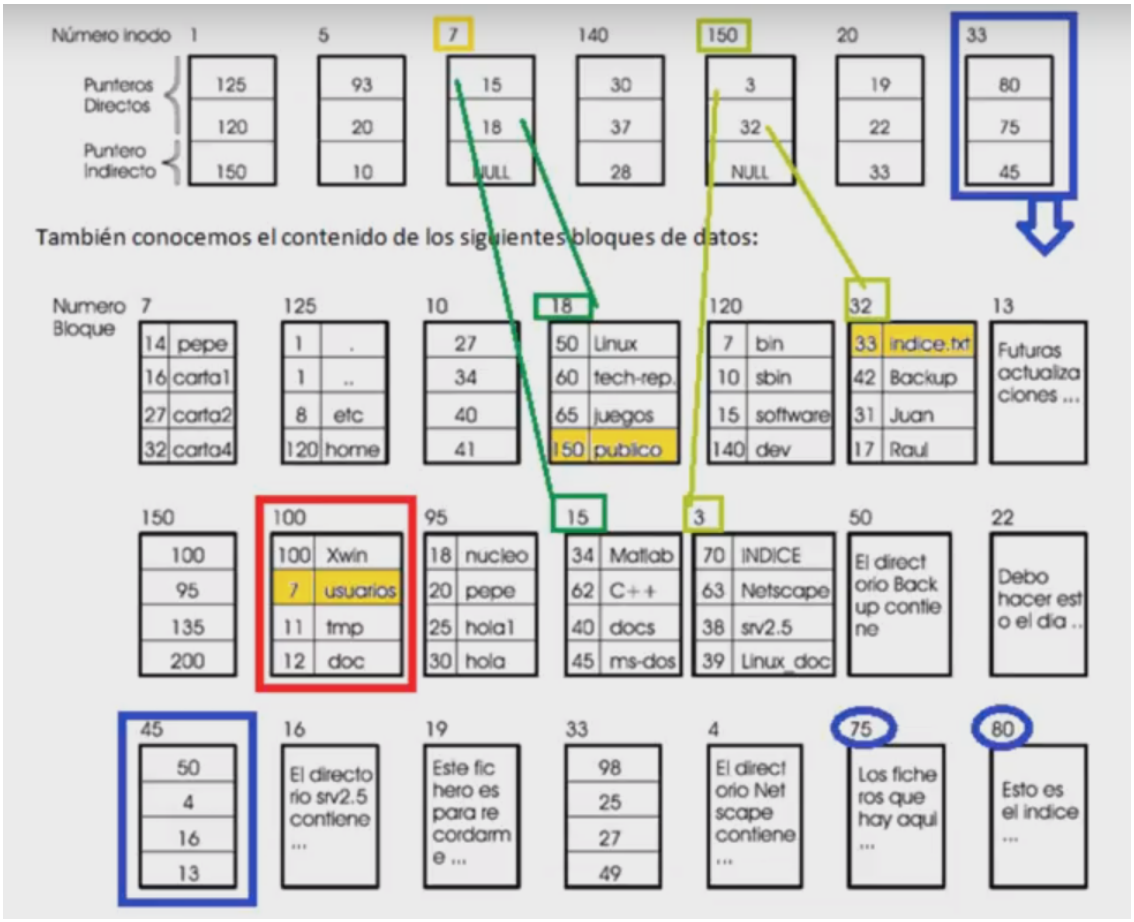


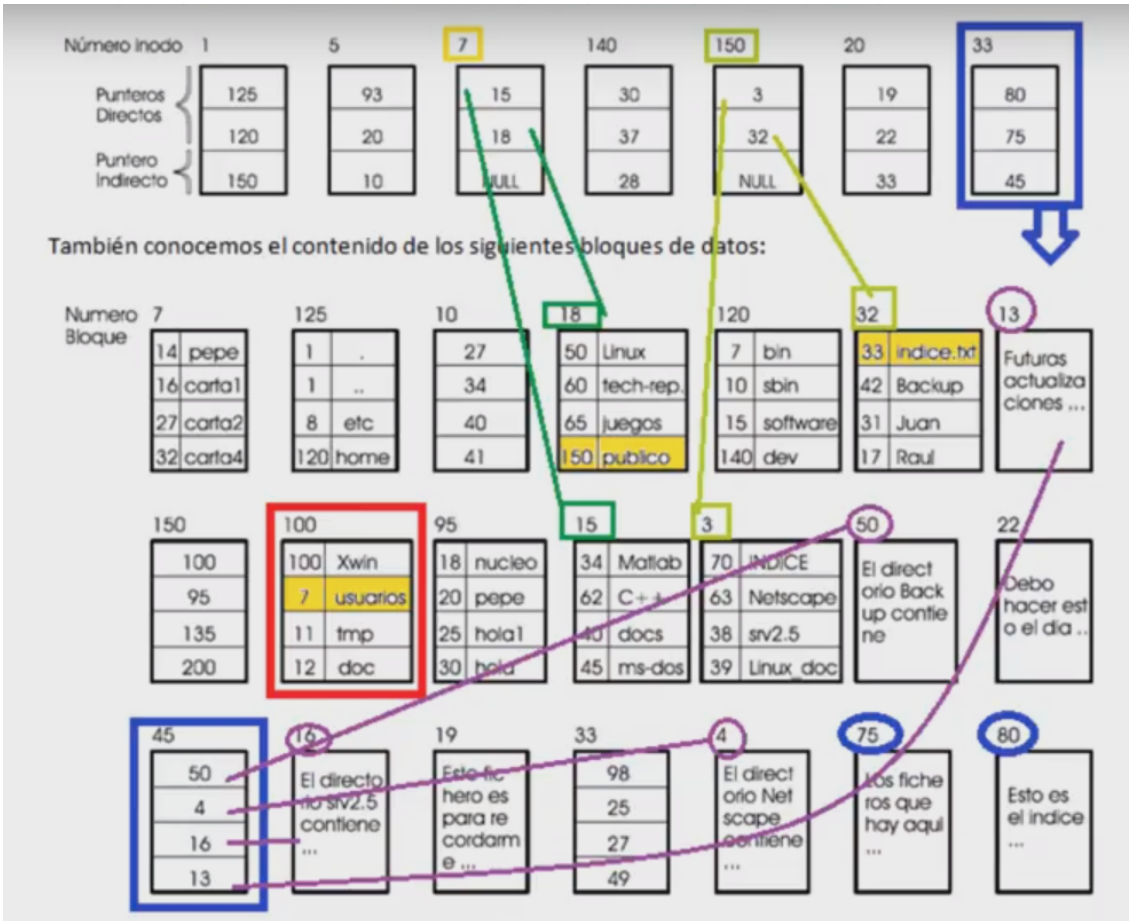
Seguimos analizando nuestra ruta+nombre: /usuarios/publico/indice.txt.

Ahora hemos de recorrer el inodo 150, correspondiente a “publico”, buscando “indice.txt” dentro de sus entradas. Empezamos por el primer bloque lógico del directorio publico. El BL 0 del inodo 150 está en el BF 3 (a nosotros nos lo devolverá la función traducir_bloque_inodo()). Comparamos los nombres de las entradas de ese bloque. Como no la hemos encontrado, pasamos al siguiente bloque lógico. El BL 1 del inodo 150 está en el BF 32. Comparamos los nombres de las entradas de ese bloque y encontramos una cuyo nombre es “indice.txt” y nos indica que su inodo es el 33.



Ya tenemos localizado el inodo correspondiente a indice.txt. Ahora recorreríamos todos los bloques lógicos del inodo 33 para mostrar su contenido (nosotros tendremos una función mi_read() que llamará a mi_read_f() en un bucle (con un tamaño de buffer concreto) para leerlo completo, similar a lo que hace el programa ficticio leer.c).


Número inodo 1 5 7 140 150 20 33
Punteros Directos
Puntero Indirecto
125 120 150
93 20 10
15 18 NULL
30 37 28
3 32 NULL
19 22 33
80 75 45
También conocemos el contenido de los siguientes bloques de datos:
Numero Bloque 7
14 pepe
16 carta1
27 carta2
32 carta4
125
1 .
1 ..
8 etc
120 home
10
27 34 40 41
18
50 Linux
60 tech-rep.
65 juegos
150 publico
120
7 bin
10 sbin
15 software
140 dev
32
33 indice.txt
42 Backup
31 Juan
17 Raul
13
Futuras actualizaciones ...
150
100 95 135 200
100
100 Xwin
7 usuarios
11 tmp
12 doc
95
18 nucleo
20 pepe
25 hola1
30 hola
15
34 Matlab
62 C++
40 docs
45 ms-dos
3
70 INDICE
63 Netscape
38 srv2.5
39 Linux_doc
50
El direct orio Back up contie ne
22
Debo hacer est o el dia ..
45
50 4 16 13
16
El directo rio srv2.5 contiene ...
19
Este fic hero es para re cordarm e ...
33
98 25 27 49
4
El direct orio Net scape contiene ...
75
Los fiche ros que hay aqui ...
80
Esto es el indice ...


Número inodo 1 5 7 140 150 20 33
Punteros Directos
Puntero Indirecto
125 120 150
93 20 10
15 18 NULL
30 37 28
3 32 NULL
19 22 33
80 75 45
También conocemos el contenido de los siguientes bloques de datos:
Numero Bloque 7
14 pepe
16 carta1
27 carta2
32 carta4
125
1 .
1 ..
8 etc
120 home
10
27 34 40 41
18
50 Linux
60 tech-rep.
65 juegos
150 publico
120
7 bin
10 sbin
15 software
140 dev
32
33 indice.txt
42 Backup
31 Juan
17 Raul
13
Futuras actualizaciones ...
150
100 95 135 200
100
100 Xwin
7 usuarios
11 tmp
12 doc
95
18 nucleo
20 pepe
25 hola1
30 hola
15
34 Matlab
62 C++
40 docs
45 ms-dos
3
70 INDICE
63 Netscape
38 srv2.5
39 Linux_doc
50
El direct orio Back up contie ne
22
Debo hacer est o el dia ..
45
50 4 16 13
16
El directo rio srv2.5 contiene ...
19
Este fic hero es para re cordarm e ...
33
98 25 27 49
4
El direct orio Net scape contiene ...
75
Los fiche ros que hay aqui ...
80
Esto es el indice ...