El sistema de fitxers

José Ramón Herrero Zaragoza
Teodor Jové Lagunas
Enric Morancho Llena

PID_00169371

UOC
Universitat Oberta de Catalunya

Índex

Introducció

Aquest mòdul didàctic se centra en l'estudi del **sistema de fitxers** (**SF**). Amb aquest objectiu fem els passos següents:

1) En primer lloc estudiarem el concepte de **fitxer** com a dispositiu lògic.

2) En segon lloc analitzarem el **sistema de fitxers** com a gestor d'objectes.

3) Finalment, veurem el cas concret del **sistema de fitxers en UNIX, en Mac OS i en Windows.**

Els tres aspectes s'estudien sempre des de la perspectiva de l'assignatura, que és el punt de vista de l'usuari del sistema operatiu.

Objectius

Els materials didàctics d'aquest mòdul inclouen les eines necessàries per a assolir els objectius següents:

1. Conèixer les funcions del sistema de fitxers.

2. Conèixer el concepte de *fitxer*, i saber quines característiques poden tenir els fitxers en funció de les seves propietats i de les seves tipologies.

3. Entendre la necessitat de facilitar l'ús per part dels usuaris. Per això es fan servir funcions de traducció per a aproximar els objectes del sistema als costums dels usuaris.

4. Aprendre els diferents tipus d'espais de noms i els seus avantatges i inconvenients.

5. Familiaritzar-vos amb les operacions que ofereixen els sistemes operatius per a gestionar els espais de noms.

6. Entendre els conceptes de *seguretat* i *protecció*.

7. Ser conscients de la necessitat d'oferir eines de protecció que permetin portar a terme diferents polítiques de protecció en funció de les necessitats dels usuaris.

8. Aprendre els diferents esquemes de protecció i els seus avantatges i inconvenients.

1. Definició del sistema de fitxers

El **sistema de fitxers** (**SF**[1]) és l'encarregat de gestionar el conjunt de fitxers continguts en un mateix dispositiu d'emmagatzematge. Aquesta gestió es concreta en els tres aspectes fonamentals que indiquem a continuació:

1) La definició dels dispositius lògics que configuren els fitxers.

2) L'espai de noms, que ens servirà per a poder localitzar i adreçar els fitxers.

3) El control de les accions que es poden efectuar sobre els fitxers.

Vegeu també

Vegeu els dispositius lògics en el subapartat 4.2 del mòdul didàctic 4.

(1)Utilitzem la sigla *SF* com a abreviació de *sistema de fitxers*.

Per ampliar aquesta definició, direm que l'SF és l'encarregat de proporcionar un espai de noms i un control d'accés a tots els dispositius. Des d'aquest punt de vista, tant es pot parlar de fitxers com de dispositius o, des d'una òptica encara més general, d'objectes gestionats pel sistema[2] i visibles per als usuaris.

(2)Objectes gestionats pel sistema, com ara fitxers, dispositius, processos, espais lògics, etc.

Cal notar que quan particionem un disc a escala lògica i el formatem per poder desar-hi diferents dades o sistemes operatius, el que fem és crear sistemes de fitxers independents per a cada partició. No és l'objectiu d'aquest document entrar a discutir els formats i les estructures internes de diferents tipus de sistemes de fitxers. No obstant això, com a usuaris hem de ser conscients que hi ha diferents formats i que en el moment en què particionem un disc i volem instal·lar un o diversos sistemes operatius, l'administrador ha d'indicar el tipus de sistema de fitxers que es vol crear. N'hi ha de molts tipus, que van evolucionant al llarg dels anys per a proporcionar noves funcionalitats o permetre mides de fitxers i sistemes de fitxers més grans. A manera d'exemple, n'esmentarem almenys un dels emprats en els sistemes operatius més habituals: ext4 en Linux, NTFS en Windows, o HFS+ en Mac Os.

2. El concepte de *fitxer*

2.1. Definició de *fitxer*

Un **fitxer** és un dispositiu lògic format per una agrupació lògica d'informació emmagatzemada en un dispositiu físic, com ara un disc, un llapis USB, o a la memòria, que pot ser manipulada com un tot. La informació que inclou té en comú un conjunt de propietats que la caracteritzen.

2.2. Les propietats dels fitxers

Un fitxer té un conjunt de característiques o propietats relacionades amb els aspectes següents:

1) La **informació** relativa al contingut del fitxer i a la seva modificació: grandària, data de creació, última data d'accés, última data de modificació, tipus d'informació, etc.

2) La **ubicació** d'aquesta informació dintre del dispositiu d'emmagatzematge. Aquest aspecte es refereix a la informació necessària perquè el sistema pugui localitzar el fitxer dintre del dispositiu d'emmagatzematge. En la dinàmica general d'ús dels fitxers, els usuaris del sistema no necessiten conèixer aquesta informació.

3) L'**accessibilitat** del fitxer. Aquest aspecte fa referència a la informació relacionada amb qui és l'usuari propietari del fitxer i amb les operacions que els diferents usuaris poden fer: escriure, llegir, executar, etc. Totes aquestes informacions estan relacionades amb el concepte de *protecció*.

Vegeu també

Vegeu el concepte de *protecció* en l'apartat 4 d'aquest mòdul didàctic.

2.3. Els tipus de fitxers

El SO pot reconèixer diferents tipus de fitxers segons l'estructura de la informació que contenen, la seva finalitat i les operacions d'accés que permeten, entre altres. En aquest mòdul didàctic ens centrarem només en aquestes tres primeres característiques, que estan fortament interrelacionades.

Així, doncs, en un SO podem trobar fitxers que contenen caràcters o informació en format binari, gràfic, etc. Hi podem trobar fitxers executables, biblioteques, fonts, directoris, documents, fulls de càlcul, bases de dades, etc. La informació continguda en els fitxers sempre té una estructura i una finalitat. Aquestes dues característiques condicionen la manera d'accedir-hi i el tipus de funcions que s'hi poden fer. En general, sobre els fitxers es pot fer el mateix conjunt d'operacions que es poden portar a terme sobre qualsevol dispositiu. No obstant això, en funció de l'estructura i la finalitat d'aquesta informació, el sistema operatiu pot ampliar o restringir el conjunt d'operacions possibles.

Vegeu també

Vegeu les operacions que poden tenir lloc sobre els dispositius en el subapartat 5.2 del mòdul didàctic 4.

Un cas d'ampliació d'operacions és, per exemple, el que es produeix amb els **fitxers executables**. Un fitxer executable, tal com hem vist, conté un tipus d'informació amb una estructura molt determinada que permet al SO carregar-la a la memòria com a codi d'un procés. Així, doncs, el sistema permet efectuar una acció especial sobre aquests fitxers: l'**execució**. Aquesta acció, tal com l'acabem de definir, no té sentit sobre fitxers que no siguin executables; tanmateix, alguns sistemes poden redefinir l'acció d'executar si associen a l'estructura d'un fitxer una aplicació que la pugui interpretar o manipular.

Exemples de redefinició de l'acció d'executar

A continuació presentem un parell d'exemples en què podeu veure la redefinició de l'acció d'executar:

a) Un exemple és el cas en què en fer clic sobre la icona d'un fitxer en una interfície gràfica s'inicia automàticament l'aplicació associada capaç d'entendre la informació que conté el fitxer.

b) Un altre exemple és el cas dels fitxers que inclouen *shell scripts*. En el sistema UNIX, quan es porta a terme l'acció d'executar un *shell script*, el sistema ho detecta i executa l'intèrpret d'ordres[3] (*shell*) que és capaç d'interpretar la informació del fitxer que configura les ordres o instruccions.

[(3)]L'intèrpret d'ordres que s'ha d'usar per a interpretar el fitxer s'indica en el sistema UNIX a la primera línia del fitxer, seguint el patró `#!nom_shell`. Per exemple, `#!/bin/bash`.

Vegeu també

Vegeu l'intèrpret d'ordres en el subapartat 7.1.3 del mòdul didàctic 4.

Un altre cas molt diferent són els **fitxers directori**, els quals tenen una estructura que consisteix en una llista de parelles de noms i nombres. La seva finalitat és donar nom als fitxers i permetre'n la localització. En aquest cas el sistema només ens deixa accedir als directoris mitjançant un conjunt molt concret d'operacions, a fi de protegir la gestió que fa sobre els fitxers en general.

Vegeu també

Vegeu els fitxers directori en l'apartat 3 d'aquest mòdul didàctic.

3. L'espai de noms

Tal com hem vist en la introducció, una de les **funcions pròpies de l'SF** és **proporcionar un espai de noms** que permeti localitzar i manipular els fitxers. Aquesta visió és extensible a tots els objectes que gestiona el sistema i que han de ser identificats en algun moment per part dels usuaris. Així, doncs, d'ara endavant en aquest apartat utilitzarem de manera indistinta els termes *fitxer*, *dispositiu* o *objecte*.

3.1. La funció de traducció

En un sistema es gestionen multitud d'objectes[4]. Per a poder gestionar-los tots és necessari complir els requisits següents:

(4)Alguns dels objectes que gestionen els sistemes són els fitxers, els dispositius, els processos i els espais lògics.

- Conèixer els objectes que hi ha i les seves característiques.
- Poder accedir-hi per a interactuar-hi.
- Poder crear-los i destruir-los.

Per a aconseguir efectuar aquestes operacions cal que els objectes tinguin nom, a fi que el sistema i els usuaris els puguin identificar. Dintre del sistema, s'identifica cadascun dels objectes mitjançant el que en podríem dir **noms interns**, que consisteixen en adreces de memòria, índexs de taules o nombres en general.

A més dels fitxers tenim, per exemple, dispositius amb els quals els usuaris hem d'interactuar directament. Aleshores el sistema ha de proporcionar noms propers als nostres hàbits. Ens és molt més fàcil recordar que el dispositiu amb el nom *impressora* és la impressora que no pas recordar un nombre generalment de diverses xifres.

Recordeu

En el cas del sistema operatiu UNIX, el nom intern és compost per la unió dels números anomenats *major* i *minor*. Afortunadament, els usuaris no cal que sapiguem quins són.

L'SF és l'encarregat de proporcionar i gestionar un espai de noms que ens permeti assolir aquest objectiu.

Una de les funcions de l'SF és **facilitar una funció de traducció** entre uns noms als quals estem acostumats els humans i els noms interns del sistema o, el que és el mateix, els objectes tal com els coneix el sistema.

$$F\text{: nom} \rightarrow \text{objecte.}$$

El conjunt de tots els noms possibles configura el que anomenem **espai de noms**. La funció de traducció ha de ser unívoca: un nom només pot fer referència a un objecte, i només a aquest objecte.

El SO implementa la funció de traducció dels noms mitjançant estructures de dades que anomenem **directoris**, els quals, al seu torn, són implementats mitjançant fitxers en què el sistema emmagatzema l'estructura de dades que dóna lloc a la funció de traducció. Aquesta estructura de dades consisteix en una taula que relaciona els noms que conté el directori amb els noms interns del SO (vegeu la figura 1): cada entrada de la taula associa un nom[5] amb l'estructura de dades interna que representa l'objecte. Així, un directori es pot veure com una agrupació de noms d'objectes. Atès que els SF poden estar continguts en dispositius extraïbles, cada SF ha de contenir tota la informació que configura el seu espai de noms.

[5]El nom pot ser el nom d'un fitxer, d'un directori o, en general, d'un dispositiu lògic accessible per mitjà del sistema de fitxers.

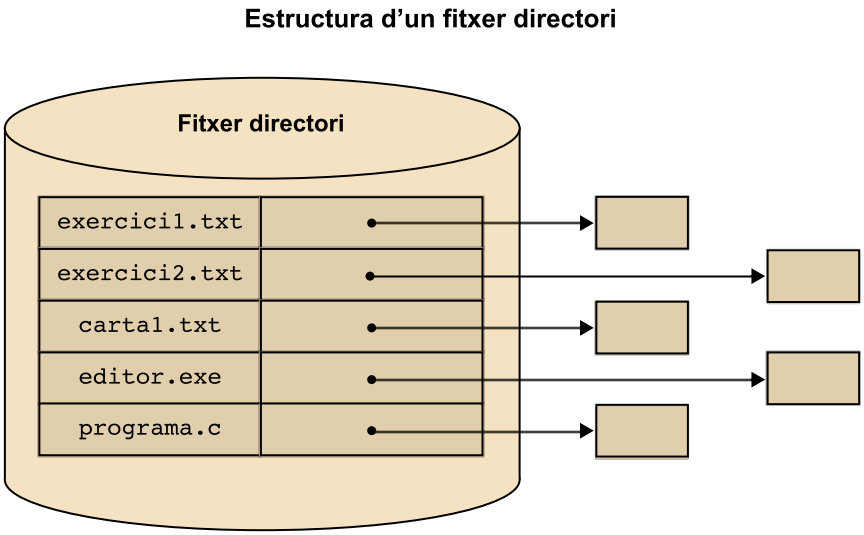

Figura 1

Un sistema operatiu pot oferir una visió d'un únic espai de noms per a tots els objectes i SF, o d'un espai de noms separat i independent dels altres per a cada tipus d'objecte o SF.

Espai de noms en UNIX, Mac Os i Windows

Els noms possibles per als fitxers i directoris són similars en tots tres sistemes. En UNIX i Mac Os els fitxers i els dispositius s'anomenen mitjançant un mateix espai, que és gestionat per l'SF. És a dir, diferents sistemes de fitxers es poden muntar[6] a partir d'un punt determinat d'un SF ja existent, de manera que l'usuari té la visió d'un únic espai de noms, organitzat a partir d'un directori que s'anomena *arrel* i es representa amb el caràcter /. De fet, el sistema operatiu Mac Os implementa un subsistema compatible amb UNIX, i no hi trobarem gaires diferències com a usuaris del sistema de fitxers. El cas de Windows és diferent, ja que dispositius i fitxers tenen espais separats:

- Els dispositius tenen uns noms prefixats, com `A:`, `B:`, etc., per als discos, i `CON:`, `LPT1:`, etc., per a la resta de dispositius.
- En canvi, els fitxers s'anomenen amb el nom del volum[7] que els conté seguit del nom del fitxer; per exemple`C:\AUTOEXEC.BAT`.

[6]Veurem operacions per a muntar i desmuntar sistemes de fitxers en el subapartat 3.3.

[7]Sovint s'anomenen *volums* (volum A:, volum C:, etc.).

3.2. L'estructura dels espais de noms

Aquest subapartat se centra en els diferents tipus d'espais de noms que pot tenir l'SF. Podem classificar els espais de noms en espais lineals i espais jeràrquics, que, alhora, poden estar distribuïts en forma d'arbre o en forma de graf dirigit.

1) L'espai lineal

L'espai lineal és l'espai de noms més senzill; té una sola dimensió en què tots els noms són al mateix nivell, continguts en un únic directori. Dit d'una altra manera, no s'hi poden fer classificacions dels diferents objectes.

Espais lineals en UNIX i en DOS

Trobem exemples d'espais lineals en els noms dels volums en Windows (`A:`, `B:`, ..., `Z:`), o en els identificadors dels processos o PID (*process identifier* o *process identification number*), que són nombres enters. Aquest nombre és gestionat pel sistema operatiu però es pot conèixer i el poden utilitzar fàcilment els usuaris en sistemes UNIX per a fer tasques de gestió dels processos que els pertanyen.

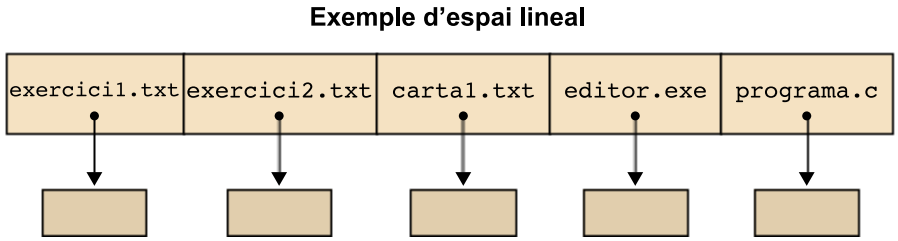

Figura 2

Els espais lineals són bons per a referenciar objectes interns del sistema o per a espais amb un nombre d'objectes petit.

Així, doncs, en un sistema monousuari en què hi hagi pocs fitxers i tots hagin de ser visibles per a un únic usuari, poden ser un bon mitjà d'organització. Ara bé, quan el nombre de fitxers creix, els noms que fem servir han de ser cada cop més complexos per a poder distingir els diferents fitxers. Cada vegada és més difícil poder recordar quin nom tenia un fitxer concret. També resulta inadequat quan tenim diversos usuaris. Tenint en compte aquestes situacions, l'objectiu de proporcionar uns noms pròxims als costums i hàbits dels humans és impossible si usem un espai lineal de noms. Per tant, fa molts anys que no s'utilitza aquesta organització lineal per a l'emmagatzemament dels fitxers.

En resum, pel que fa als **espais lineals**, podem dir que són:

- **Adequats** per a espais amb pocs objectes, ja que es tracta d'una estructura que permet tenir una bona visió de conjunt i aconseguir una localització ràpida de qualsevol objecte.

- **Inadequats** quan el nombre d'objectes comença a ser elevat, ja que el significat que poden tenir els noms per als humans pot arribar a quedar desvirtuat.

2) Espai jeràrquic

Com a usuaris ens agradaria poder agrupar els fitxers per tipus, treballs o qualsevol criteri que ens ajudi a organitzar l'SF. D'aquesta manera seria molt més fàcil localitzar els fitxers. La manera d'aconseguir aquest objectiu és dotar els noms del sistema de fitxers d'una estructura jeràrquica. Per **aconseguir un espai jeràrquic** tractem els directoris com a objectes amb un nom i, així, poden formar part dels objectes que agrupen altres directoris.

Amb aquesta idea podem organitzar els espais jeràrquics de les dues maneres següents:

a) Estructura d'arbre. És l'estructura jeràrquica més senzilla en què podem organitzar els fitxers. Un espai jeràrquic en forma d'arbre és format per una jerarquia de directoris amb un **directori arrel** del qual pengen altres directoris o fitxers, els quals configuren les **fulles de l'arbre**.

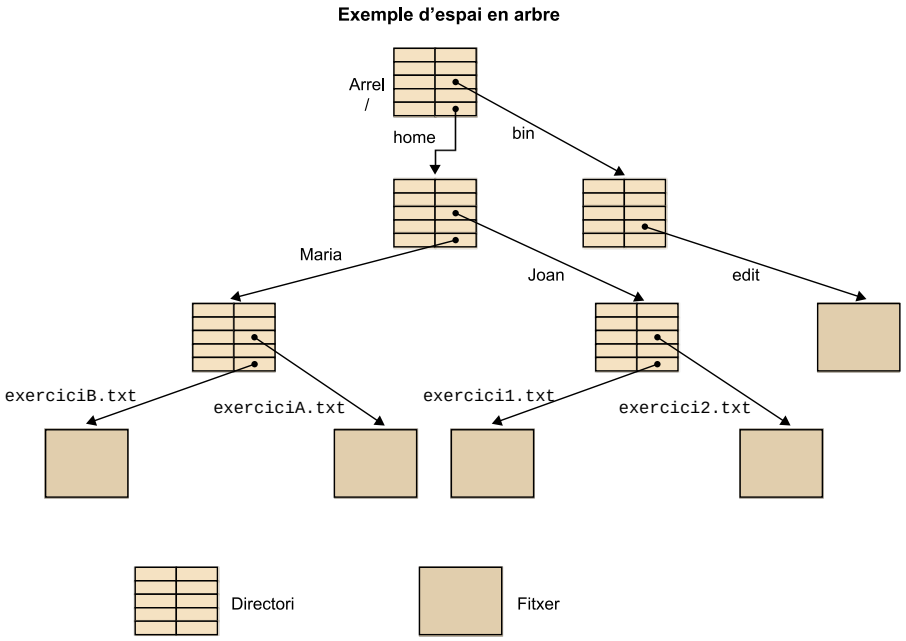

Figura 3

Anomenem **nom absolut** el nom que és format per la ruta que va des de l'arrel, passant pels diferents subdirectoris, fins a arribar al fitxer. Així, doncs, el nom absolut és compost per una seqüència de cadenes de caràcters, o components del nom, contingudes en els subdirectoris que configuren la ruta d'accés al fitxer. Els components se separen per un caràcter delimitador[8].

[8] En UNIX i Mac Os el caràcter delimitador és /. En Windows, en canvi, el caràcter delimitador és \.

Per a facilitar la manipulació dels fitxers en les estructures jeràrquiques es defineix:

- El **directori inicial**, que és el directori des d'on un usuari pot crear la seva estructura de directoris i on col·locarà els fitxers que creï.

- El **directori de treball**, que és el directori on es troben els fitxers amb els quals està treballant en un instant concret. Quan un usuari comença la sessió de treball, el directori de treball és el directori inicial.

Associat al concepte de *directori de treball* hi ha el concepte de **noms relatius**, que són els noms formats pel recorregut que va des del directori de treball fins al fitxer. Diversos fitxers poden tenir el mateix nom relatiu, sempre que aquest s'obtingui a partir de directoris de treball diferents[9].

[9]Amb la qual cosa els noms absoluts no són iguals.

Com ja hem dit, una estructura en arbre soluciona els problemes organitzatius que planteja l'estructura lineal. Això no obstant, té els dos inconvenients següents:

- No permet compartir d'una manera senzilla fitxers entre diferents usuaris.
- No permet moure's per l'arbre de directoris en sentit ascendent, canviant el directori de treball.

b) Estructura de graf dirigit. Per solucionar els problemes que planteja l'estructura en arbre, definim l'espai de noms amb una estructura de graf dirigit com la que es representa en la figura 4.

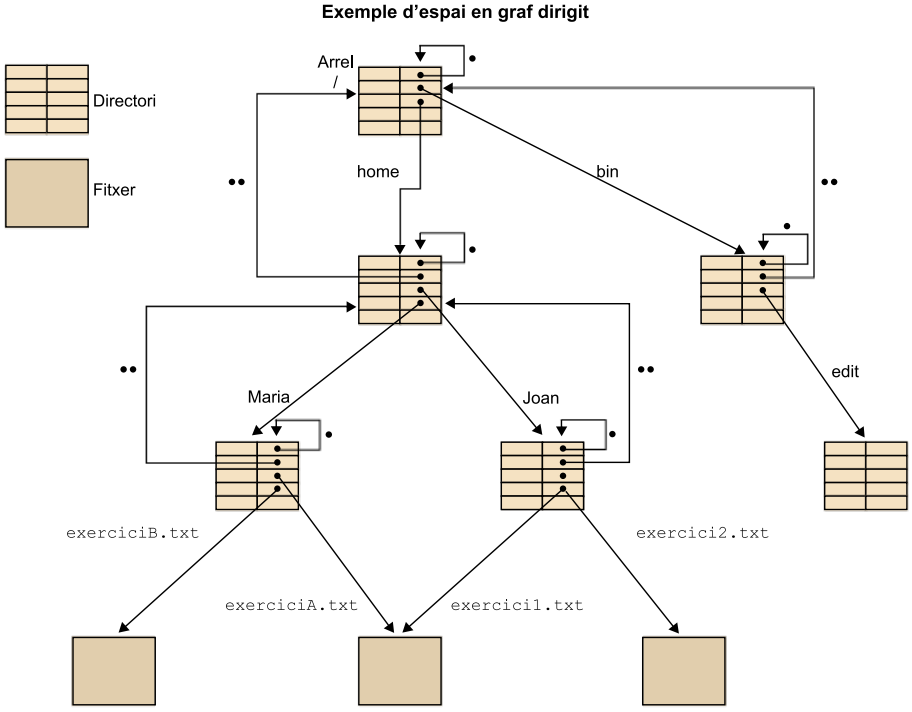

Figura 4

Amb l'estructura de la figura 4 un fitxer pot ser adreçat alhora amb els noms absoluts `/home/joan/exercici1.txt` i `/home/maria/exerciciA.txt`. D'aquesta manera pot ser compartit fàcilment per més d'un usuari.

Vegeu també

En el subapartat 3.3.2.c introduirem l'operació que permet crear noms, que ens servirà per a aquest propòsit.

En una estructura en graf dirigit, es pot accedir a un mateix objecte mitjançant més d'un nom i, per tant, també s'hi pot accedir des de directoris de treball diferents.

Amb aquesta estructura el sistema genera de manera automàtica noms addicionals per a cada directori de l'SF. Per exemple, en UNIX i en DOS, a cada directori, hi figuren dos noms especials: el ".", és el directori actual, i el "..", el superior i/o pare.

- El nom "." fa referència al directori que el conté. S'utilitza per a referenciar el directori de treball sense necessitat de conèixer-ne el nom absolut.

- El nom ".." és el nom del directori pare que el conté. S'utilitza per a referenciar, mitjançant noms relatius, objectes que tenen com a part del seu nom subdirectoris que es troben per sobre del directori de treball.

En general, els directoris "." i ".." permeten referenciar objectes mitjançant noms relatius sense haver de conèixer el directori de treball actual.

Exemple d'adreça

En el cas de la figura 4, si el directori de treball és `/home/maria`, es pot accedir al fitxer amb nom absolut `/home/joan/exercici2.txt` amb el nom relatiu `../joan/exercici2.txt`.

El principal inconvenient dels espais amb estructura de graf dirigit és, com veurem tot seguit, la seva gestió.

3.3. Les operacions sobre l'espai de noms

Un cop vist el que és un espai de noms, passem a les operacions que hi podem fer. El sistema ofereix tres tipus d'operacions: operacions per a manipular l'SF com un tot, operacions per a gestionar els continguts dels directoris i, finalment, operacions per a manipular el directori de treball. Tot seguit, tractarem cadascun d'aquests tipus d'operacions.

3.3.1. Operacions de manipulació de l'SF

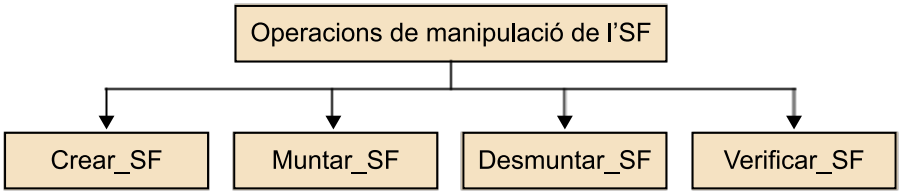

Figura 5

a) Operació *crear_SF*. Com ja hem vist, un SF està inclòs dintre d'un dispositiu d'emmagatzematge i organitza la informació d'aquest dispositiu en fitxers. Perquè això sigui possible s'ha de dotar el dispositiu d'un conjunt d'estructures de dades que permetin definir els fitxers, localitzar-los mitjançant un espai de noms, crear-ne de nous, gestionar l'espai lliure, etc. L'operació *crear_SF* és l'encarregada de generar totes aquestes estructures que es desaran dintre del

dispositiu mateix d'emmagatzematge. Aquest fet permetrà, en cas que s'utilitzi un dispositiu extraïble, transportar l'SF d'un sistema a un altre sense que hi hagi pèrdua d'informació.

b) Operació *muntar_SF*. Alguns SO han de ser informats abans d'accedir a un nou SF, tant si és perquè s'acaba d'introduir un volum nou que conté un SF en un dispositiu extraïble, com si és perquè s'ha creat un SF nou en un dispositiu ja existent, etc. El sistema necessita inicialitzar una sèrie d'estructures de dades internes per a optimitzar els accessos a l'SF, per a reconèixer de quin tipus d'SF es tracta, per a restringir el tipus d'accés[10] sobre l'SF o per a donar, com en UNIX, una visió d'un únic espai de noms, etc. Totes aquestes accions les porta a terme l'operació *muntar_SF*. Cal dir que aquesta operació no modifica l'SF. Només canvia informació de l'SO a memòria per tal d'accedir-hi de manera automàtica en accedir al punt (directori) de muntatge.

[10] Els tipus d'accés permesos són de lectura i d'escriptura.

c) Operació *desmuntar_SF*. De manera anàloga al cas anterior, abans de retirar el volum d'un dispositiu extraïble al qual s'està accedint com a SF, s'ha d'efectuar l'operació *desmuntar_SF* a fi que s'alliberin les estructures de dades que s'han reservat en l'operació de muntatge i s'actualitzi el contingut de l'SF. És important que es faci correctament perquè, si per exemple extraiem un llapis USB sense desmuntar-lo correctament, podem perdre els canvis fets. Això passa perquè el sistema operatiu pot anotar a la memòria els canvis que s'han de fer, i retarda l'escriptura al disc per a moments posteriors, per tal d'augmentar el rendiment, ja que l'accés a la memòria és normalment molt més ràpid que l'accés al disc. Si s'extreu el dispositiu sense haver materialitzat els canvis al dispositiu, es perden. El mateix pot passar si un ordinador es queda de sobte sense alimentació elèctrica.

d) Operació *verificar_SF*. Finalment, el sistema ens proporciona eines per a verificar les estructures de dades que configuren un SF i que estan contingudes en el dispositiu mitjançant l'operació *verificar_SF*. Normalment aquestes eines no formen part del nucli de l'SO, sinó que són aplicacions i utilitats que s'executen sobre el SO.

Així, doncs, les operacions de manipulació de l'SF poden tenir la forma següent:

- Estat = crear_SF(*nom*,*tipus*, etc.).
- Estat = muntar_SF(*dispositiu*,*tipus*,*modeL/E*, directori, etc.).
- Estat = desmuntar_SF(*dispositiu*).
- *Verificar_SF* no és una crida al sistema, sinó una utilitat.

3.3.2. Operacions de manipulació de directoris

Les operacions de manipulació de directoris ens permeten accedir a la taula que implementa la funció de traducció, que es troba emmagatzemada dins del fitxer que configura el directori.

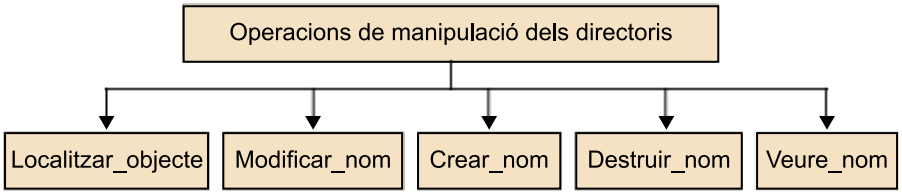

Figura 6

Vegeu també

Vegeu la funció de traducció en el subapartat 3.1 d'aquest mòdul didàctic.

a) Operació *localitzar_objecte*. Consisteix, donat un nom (relatiu o absolut), a localitzar l'objecte al qual fa referència. Segons el sistema hi pot haver una crida que efectuï directament aquesta operació, o pot ser necessari fer una cerca a partir d'una crida que retorni una per una les entrades de directori. La localització d'un objecte mitjançant el seu nom és un procediment que forma part de multitud de crides. En general, qualsevol crida al sistema que faci referència a un dels objectes continguts en l'SF ha d'utilitzar aquest procediment.

b) Operació *modificar_nom*. Consisteix a localitzar l'entrada del directori que conté el nom que s'ha de modificar i, un cop verificat que el nom nou és únic (no existeix prèviament), canviar-lo pel nou.

c) Operació *crear_nom*. És l'encarregada de crear un nom nou per a un objecte concret. Aquesta acció es pot separar en diferents operacions en funció de si l'objecte ja existeix i, per tant, es pretén senzillament crear un nom nou, o si l'objecte no existeix i, per tant, la creació del nom es fa en el mateix instant de la creació de l'objecte:

- La **creació de noms sobre objectes ja existents**, que es pot veure com l'establiment d'un enllaç entre el nom i l'objecte al qual fa referència. Des d'aquesta òptica, la creació d'un nom nou consisteix simplement a afegir una línia d'informació a la taula del directori. Abans, però, s'ha de localitzar l'objecte al qual es vol donar un nom nou, i s'ha de comprovar que el nom sigui únic. Aquest tipus de nom s'anomena **enllaç físic** (*hard link*), en contraposició als anomenats *enllaços simbòlics*.
  Un **enllaç simbòlic** (*symbolic link* o *soft link*) és un enllaç que no relaciona directament un nom amb un objecte, sinó que ho fa indirectament mitjançant un enllaç a un altre fitxer que conté el nom de l'objecte, com es veu en la figura 5 (que podeu veure a la pàgina següent). Aquesta mena de noms s'utilitza fonamentalment per a donar un nom a un objecte que es troba en un SF diferent d'on està el directori que conté l'enllaç simbòlic. L'existència dels enllaços simbòlics fa que l'operació *localitzar_objecte* hagi de saber de quin tipus és cada enllaç, i actuar en conseqüència.

Nota

Un enllaç físic només pot apuntar a un objecte del mateix SF. En canvi, un enllaç simbòlic pot apuntar a un objecte d'un altre SF.

- La **creació de noms juntament amb els objectes als quals fan referència**, que hi afegeix la problemàtica pròpia de la creació dels objectes en concret. Es poden crear fitxers, directoris o dispositius de diferents tipus. Nosaltres ens centrarem en l'operació de crear un subdirectori. En aquest cas, el sistema ha de proporcionar una operació que, donat el nom del directori que es vol crear i el del directori des d'on ha de penjar, creï el fitxer[11] que ha de contenir el directori nou, la seva taula de traducció i els enllaços "." i "..".

[11]Un directori és un fitxer amb un format especial que permet emmagatzemar la taula de traducció del directori.

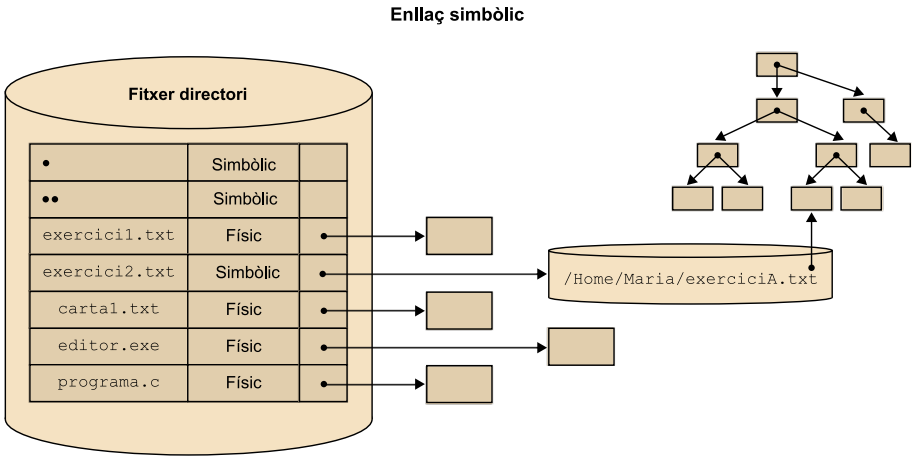

Figura 7

d) Operació *destruir_nom*. Consisteix a localitzar el directori que conté l'últim component del nom que es vol destruir i esborrar l'entrada de la taula de traducció que el conté. A més, si el fitxer al qual feia referència aquest nom ja no té cap més nom associat, el SO el destruirà i alliberarà els recursos que ocupa. En comprovar si el fitxer té un altre nom, trobem les dues situacions següents:

- Detectar si un fitxer no té cap més nom és senzill si ens referim als enllaços físics, ja que els enllaços físics existents estan comptabilitzats en les característiques del fitxer i estan inclosos en el dispositiu d'emmagatzematge que conté l'SF.

- Per contra, la detecció d'enllaços simbòlics no és trivial, atès que els enllaços simbòlics del fitxer que es vol destruir no han d'estar necessàriament en el mateix dispositiu, i aquest no ha d'estar necessàriament muntat en el sistema. Per tant, pot resultar impossible detectar-los, i el sistema ignora la seva existència durant l'operació de destruir.

Com a resultat d'això, pot ser que apareguin enllaços simbòlics que no assenyalin cap fitxer existent o, encara pitjor, que apuntin a un fitxer destinació diferent de l'altre al qual volien apuntar, però que ha reutilitzat el nom del fit-

xer destruït. Això últim, però, pot tenir la seva utilitat pràctica, perquè permet la instal·lació de noves versions dels fitxers destinació de manera transparent a l'usuari.

En cas que es vulgui destruir un directori, cal que aquest estigui buit. Es considera buit si només té les entrades corresponents al directori mateix "." i al superior ".." dintre del qual es troba.

e) Operació *veure_noms*. Permet consultar el contingut d'un directori, i també veure el tipus i les propietats dels objectes als quals fa referència.

Així, doncs, les operacions de manipulació de directoris poden quedar de la manera següent:

- Estat = localitzar_objecte(*nom, informació_interna*).
- Estat = modificar_nom(*nom_nou*,*nom_vell*).
- Estat = crear_nom(*directori*,*nom_nou*,*simbòlic_físic*,*objecte*).
- Estat = destruir_nom(*nom*).
- Valors = veure_noms(*directori*).

3.3.3. Operacions de manipulació del directori de treball

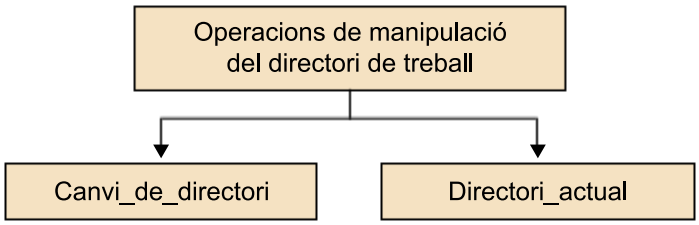

Figura 8

Hi ha dues operacions de manipulació del directori de treball: *canvi_de_directori* i *directori_actual*. La primera serveix per a canviar el directori de treball. La segona ens indica en quin directori estem treballant en un instant determinat.

Els paràmetres que podrien tenir les operacions de manipulació del directori de treball són els següents:

- Estat = canvi_de_directori(*nou_directori*).
- Directori_actual = directori_actual().

4. La protecció

Igual que en l'apartat anterior, enfocarem el tema de la protecció des del punt de vista de l'accés als fitxers, encara que els conceptes i les tècniques que estudiarem es poden estendre fàcilment al cas general de tots els objectes gestionats pel SO.

Com ja sabem, un SO gestiona un conjunt de recursos i objectes sobre els quals es poden efectuar accions. És responsabilitat del sistema operatiu garantir-ne el bon funcionament, i impedir que la dinàmica d'uns usuaris estigui afectada per les accions d'altres usuaris o per les accions d'agents externs al sistema computador.

El concepte de **protecció** fa referència al primer d'aquests aspectes. És el mecanisme que proporciona el sistema per autoritzar o denegar els accessos que en un instant concret sol·liciten els usuaris.

El concepte de **seguretat del sistema**, en canvi, se centra bàsicament en el segon aspecte, i inclou temes com la identificació i autenticació dels usuaris, i també qüestions administratives i organitzatives relacionades amb la gestió d'un sistema informàtic.

Com és evident, *seguretat* i *protecció* són dos conceptes fortament relacionats. Malgrat això, en aquest mòdul ens centrarem en el mecanisme de protecció, i suposarem que els usuaris han estat degudament identificats i que, per tant, són realment qui diuen que són.

4.1. La protecció: concepte i objectius

El concepte de protecció es refereix al control que porta a terme l'SO sobre les diverses maneres que tenen els usuaris d'accedir als objectes del sistema.

Hi ha casos en què no és necessària l'existència d'un mecanisme de protecció:

a) L'estudi de la protecció té sentit des del moment en què hi ha més d'un usuari en el sistema, i aquests poden decidir com i amb qui comparteixen els fitxers que creen. En conseqüència, la protecció no té sentit en un sistema amb un sol usuari. En aquest cas podem admetre que no cal cap tipus de control sobre els accessos que es fan als fitxers. Tots els fitxers pertanyen al mateix usuari, i ell controla l'ús que en fa.

b) Un cas similar és el d'un sistema multiusuari en el qual es deixa que tots els usuaris puguin accedir a tots els fitxers. No hi ha el concepte de propietat d'un fitxer i, per tant, no hi ha restriccions per a accedir-hi. En aquesta situació tampoc no faria falta un sistema de protecció.

En canvi, sí que es necessita un mecanisme de protecció que controli la identitat dels processos que intenten accedir als fitxers i la seva propietat, quan cada usuari només pot accedir als fitxers que ha creat, o en el cas general en què es permet als usuaris compartir fitxers mitjançant un accés controlat.

Ens centrarem en aquest últim supòsit més general per analitzar els diferents mecanismes de protecció.

Com hem vist, hi pot haver diferents necessitats a l'hora de protegir un objecte, les quals depenen de l'objecte en concret, dels usuaris que hi poden accedir i del tipus d'accions que hi poden fer. Per aquest motiu el sistema ha de proporcionar mecanismes que siguin prou flexibles a fi que els usuaris puguin implementar diferents polítiques de protecció segons les seves necessitats.

Abans de veure els mecanismes de protecció en els apartats que segueixen, hem d'analitzar els diferents **elements que intervenen en la protecció**. Són els següents:

1) Els **objectes**: són tots aquells elements gestionats pel SO sobre els quals es poden efectuar diferents accions de les quals poden haver de ser protegits. Per exemple, són objectes els fitxers, els dispositius, els directoris, els processos, etc.

2) Els **dominis**: són els diferents agents actius del sistema que poden actuar sobre un objecte. Per exemple, són dominis els processos que pertanyen a un usuari o a un grup predefinit d'usuaris, etc.

3) Els **drets**: són accions permeses per part d'un domini sobre un objecte. Per exemple, són drets llegir, escriure, executar, crear, destruir, etc.

Els mecanismes de protecció que analitzarem tot seguit són maneres diferents de relacionar els drets que cada domini té sobre cada objecte.

4.2. La matriu d'accessos

La matriu d'accessos està formada per tantes files com dominis (D_i) hi ha, i tantes columnes com objectes (O_j) hi ha. Cada cel·la D_{ij} de la matriu conté els drets d'accés del domini D_i sobre l'objecte O_j.

Matriu d'accessos											
Dominis	**Objectes**										
	exerciciA.txt	**exerciciB.txt**	**exercici1.txt**	**exercici2.txt**	**edit**	**Directori Joan**	**Domini Joan**	**Directori Maria**	**Domini Maria**	**Domini estudiant**	**Domini professors**
	Llegir	Llegir	Llegir	Llegir	Llegir	Llegir	Propietari	Llegir			
Joan			Escriure	Escriure	Executar	Escriure		Executar			
			Propietari	Propietari		Executar Propietari					
	Llegir	Llegir	Llegir	Llegir	Llegir	Llegir		Llegir	Propietari		
Maria	Escriure	Escriure			Executar			Escriure			
	Propietari	Propietari						Executar Propietari			
Estudiants	Llegir	Llegir	Llegir	Llegir	Llegir Executar	Llegir Executar		Llegir Executar		Propietari	
Professors					Llegir Executar						Propietari

En aquesta matriu, el domini `Joan` té drets de lectura i escriptura sobre l'objecte `exercici1.txt`, però, en canvi, només té drets de lectura sobre l'objecte `exerciciB.txt`, i de lectura i execució sobre `edit`. El fet de tenir drets d'escriptura i execució sobre el directori `Joan` li permetrà crear nous fitxers en aquest directori. El permís d'execució sobre un directori permet dur a terme accions sobre un directori, com per exemple canviar el directori actual de treball d'un procés a aquell directori, i sempre que coneguem el nom dels fitxers continguts al directori i tinguem permís per a accedir al fitxer concret, mostrar-ne el contingut. Si a més tenim permís d'escriptura sobre el directori els podrem reanomenar o crear-ne de nous. El permís de lectura sobre el directori indica que es permet demanar al sistema el contingut del directori i aquest retornarà informació sobre els fitxers i subdirectoris que conté. El dret

de propietari que té sobre el fitxer `exercici1.txt` li permet modificar qualsevol dret que estigui en la columna associada a aquest fitxer. Per exemple, pot donar el dret d'escriptura al domini `Maria` sobre aquest fitxer.

En la taula veiem que els dominis apareixen com a objectes. El motiu és que un domini també és un objecte sobre el qual es poden dur a terme accions i, per tant, han de ser sotmeses al mecanisme de control.

Per simplificar, no s'ha inclòs en la taula el domini *Administrador*. Cal recordar però, que l'administrador té tots els drets possibles sobre qualsevol objecte i domini del sistema.

La matriu d'accessos és una bona manera de veure les relacions que hi ha entre els diferents elements que intervenen en el mecanisme de protecció. No obstant això, no és una bona eina de treball pels motius següents:

- El seu volum creix ràpidament a mesura que augmenten el nombre d'usuaris i fitxers del sistema.

- El nombre d'objectes sobre els quals un domini té drets és petit en relació amb el nombre total d'objectes del sistema i, per tant, la matriu és una estructura molt gran amb la majoria de les cel·les buides.

Per a solucionar aquests problemes s'han proposat dos mètodes que apareixen en partir la matriu per columnes o per files. En els pròxims subapartats analitzarem els efectes que tenen aquestes solucions.

4.3. Les llistes de control d'accessos (*access control lists*)

Una llista de control d'accessos (LCA) és una llista de parelles (*drets*, *domini*) associada a un objecte. Aquesta llista és el resultat de partir la matriu d'accessos en columnes i eliminar totes les cel·les buides. D'aquesta manera el sistema estalvia espai i obté una estructura més dinàmica.

Amb aquest esquema, com que l'LCA es troba associada als objectes i no als processos, cada cop que un procés vol portar a terme un accés a un objecte s'ha de verificar si hi té dret recorrent a l'LCA. Això vol dir que en una sessió de treball amb un fitxer es verifiquen els drets a cada operació de lectura o escriptura que es vol fer. Això representa un volum de sobrecàrrega important per al sistema.

Vegeu també

En l'apartat 4.5 veurem com es pot resoldre el problema de la sobrecàrrega.

Un avantatge d'aquest esquema és que tots els drets que tenen els dominis sobre un objecte es troben en un mateix lloc, l'LCA. Aquesta circumstància fa que l'assignació i la revocació de drets per part del propietari de l'objecte es puguin fer de manera senzilla.

4.4. Les llistes de *capabilities* (capacitats)

Una llista de *capabilities* és una llista de parelles (*objecte*, *drets*) associada a un domini, cadascuna de les quals s'anomena *capability*. Les llistes de *capabilities* resulten de partir la matriu d'accessos en files i eliminar, igual que en les llistes de control d'accés, totes les cel·les buides.

Les llistes de *capabilities* són estructures de dades associades a processos. El sistema les pot gestionar de les dues maneres següents:

a) Protegint-les en el seu espai i proporcionant crides per tal que els processos les puguin utilitzar.

b) Deixant que els processos les gestionin directament. En aquest cas les *capabilities* es protegeixen mitjançant tècniques de xifratge que eviten que es modifiquin els drets que contenen.

En qualsevol dels dos casos, quan un procés accedeix a un objecte, ha de presentar al sistema la *capability* que li dóna dret a portar-lo a terme. El sistema en verifica la validesa i efectua l'accés. L'accés a aquest mecanisme és més àgil que a les LCA.

En canvi, la modificació dels drets que tenen els dominis sobre un objecte per part del propietari no és trivial, ja que els drets es troben repartits per tots els processos del sistema i, per tant, en el cas de voler revocar un dret de tots els dominis, s'ha de fer un recorregut per tots i eliminar les *capabilities* associades.

4.5. Millores i models combinats

Com acabem de veure, els models d'LCA i de *capabilities* tenen alguns inconvenients. Per a solucionar-los la majoria de sistemes utilitzen una combinació dels dos models, cosa que permet gaudir dels avantatges de cada esquema.

Com a mecanisme bàsic utilitzen les LCA, que se solen presentar fent una reducció de tots els dominis possibles a uns quants. Això s'aconsegueix agrupant els dominis, com per exemple en UNIX, en què n'hi ha tres grups: el propietari (per exemple `Joan`), tots els dominis associats al grup de treball del propietari (per exemple `Alumnes`) i la resta de dominis del sistema.

Els primers accessos a cada objecte es verifiquen mitjançant les LCA. Aquest primer accés es pot concretar, per exemple, en l'operació *obrir* que s'efectua en iniciar una sessió de treball amb un fitxer o un dispositiu. Cal recordar que en el moment d'obrir l'objecte s'especifica el tipus d'accés que es vol fer. Per tant, és possible comprovar si el procés que fa l'operació té permisos per a fer aquest tipus d'accés. Un cop verificat el dret, el sistema genera una *capability* que s'utilitzarà durant la sessió de treball. La *capability* generada permetrà fer únicament el tipus d'operació indicat en el moment d'obrir l'objecte. Normalment, aquesta *capability* s'associa al dispositiu virtual que es genera durant l'operació *obrir*. Cada cop que es fa una operació sobre el dispositiu virtual es comprova si la *capability* associada ho permet. Això, per exemple, permet evitar que s'intenti escriure un fitxer que s'ha obert només per a lectura. Un cop finalitzada la sessió de treball, la *capability* es destrueix amb l'operació *tancar*. D'aquesta manera es comprova cadascun dels accessos de lectura o escriptura que es fan durant la sessió de treball sense necessitat d'accedir a l'LCA per a cada accés[12].

Vegeu també

En el mòdul didàctic 4 vegeu l'operació *obrir* (en el subapartat 5.2.2) i els dispositius virtuals (en el subapartat 4.3).

(12)Accedir a la *capability* associada al dispositiu virtual sobre la qual s'intenta fer una operació de lectura o escriptura és molt més ràpid que accedir a l'LCA associada al fitxer.

Una altra millora, en aquest cas en el mecanisme de *capabilities*, consisteix a dotar-lo d'una eina de revocació eficient. Una possibilitat és l'anomenat **mecanisme de pany i clau**. En aquest esquema cada *capability* té associat un nombre anomenat **clau**. Els objectes tenen associada una col·lecció de nombres anomenats **panys**. Perquè una *capability* sigui correcta la clau ha de coincidir amb algun dels panys que té l'objecte. Per a revocar un conjunt de drets l'únic que ha de fer el propietari de l'objecte és eliminar un o més panys. Evidentment, aquest mecanisme no té la flexibilitat de les LCA, però soluciona parcialment el problema d'haver de recórrer tots els dominis per a eliminar les *capabilities* associades.

Una altra alternativa consisteix a associar informació temporal a una *capability* de manera que s'estableixi una data de caducitat.

5. Exemples de sistema de fitxers i protecció

5.1. UNIX

5.1.1. El sistema de fitxers en UNIX

En UNIX hi ha els tipus de fitxers següents:

1) Els **fitxers ordinaris**, que són els fitxers tal com els hem estudiat en aquest mòdul.

2) Els **fitxers directori**, que són els que configuren la funció de traducció de l'espai de noms.

3) Els **fitxers especials** o **dispositius**, que, com el seu nom indica, són els dispositius del sistema.

4) Els ***soft links*** o ***symbolic links***, que, com el seu nom indica, permeten fer enllaços simbòlics a altres fitxers.

El sistema operatiu UNIX reconeix un sol tipus de fitxer ordinari, que percep tots els fitxers com una seqüència de bytes, als quals s'accedeix mitjançant operacions específiques. Aquesta regla només té una excepció: els fitxers executables. Un fitxer executable té una estructura molt concreta que el SO reconeix i utilitza a l'hora de carregar un programa a la memòria. UNIX utilitza els primers bytes d'aquesta estructura del fitxer per a deixar-hi una marca que distingeix un fitxer normal d'un d'executable:

Vegeu també

Vegeu les operacions d'accés als fitxers en el subapartat 5.2.1 del mòdul didàctic 4 i els fitxers executables en el subapartat 1.3 del mòdul didàctic 2.

a) El fet que no hi hagi la marca garanteix que el fitxer no és un executable.

b) L'existència de la marca, però, no garanteix que el fitxer sigui executable, ja que hi pot haver fitxers binaris que per atzar tinguin la mateixa combinació de bytes al principi.

El sistema de fitxers d'UNIX té una estructura de graf dirigit organitzada en directoris, tal com hem vist. Cada fitxer pot tenir més d'un nom emprant el mecanisme d'enllaços físics o simbòlics. A l'hora d'adreçar un fitxer es poden utilitzar noms absoluts o relatius. Cadascun dels noms de subdirectoris que componen el nom d'un fitxer estan separats pel caràcter /.

Vegeu també

Vegeu l'SF amb estructura de graf dirigit en el subapartat 3.2 d'aquest mòdul didàctic.

Cada dispositiu d'emmagatzematge pot contenir un SF que es crea mitjançant la utilitat `mkfs`. Aquesta utilitat crea dintre del dispositiu l'estructura del directori, les estructures de dades necessàries per a gestionar l'espai lliure i també les estructures per a emmagatzemar les característiques dels fitxers (anomenades *inodes*), com ara l'espai que ocuparà un fitxer, o quin *major* i *minor* s'ha d'utilitzar per a localitzar un dispositiu, qui n'és el propietari, etc.

Vegeu també

Vegeu el *major* i el *minor* en l'apartat 7.1 del mòdul didàctic 4.

Malgrat que cada dispositiu que conté un SF té la seva estructura de directoris pròpia, UNIX dóna la visió d'un espai en forma de graf totalment connectat. Això significa que hi ha un únic directori arrel, i que sempre hi ha un camí des d'aquest directori arrel fins a qualsevol fitxer, com es veu en la figura 9.

Per a poder donar aquesta visió única, el sistema té un SF permanentment accessible[13] sobre el qual s'aniran afegint la resta d'SF abans de poder-hi accedir. L'operació `mount` és l'encarregada d'associar el directori arrel de l'SF que es vol muntar amb un directori d'un SF que ja estigui muntat. A part d'això, l'operació `mount` porta a terme operacions d'inicialització d'estructures internes del SO que agiliten els accessos.

[13]L'SF del sistema s'ha muntat en el temps d'inicialització del sistema.

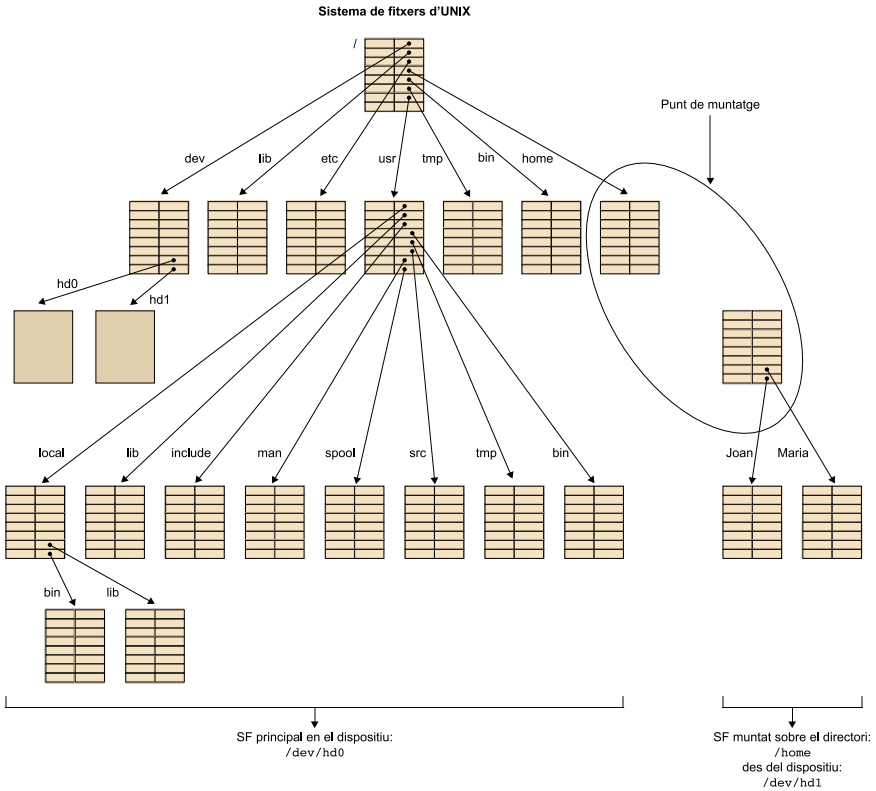

Figura 9

Les crides al sistema i les ordres que ens ofereix UNIX per gestionar l'SF segueixen la mateixa estructura que hem descrit en l'apartat de l'espai de noms. És la que podeu veure en la taula següent:

Vegeu també

Vegeu l'espai de noms en l'apartat 3 d'aquest mòdul didàctic.

Operació	Crida a l'SO	Ordre	Comentaris
Crear_SF	–	`mkfs`	–

Operació	Crida a l'SO	Ordre	Comentaris
Muntar_SF	`mount`	`mount`	–
Desmuntar_SF	`umount`	`umount`	–
Verificar_SF	–	`fsck`	–
Localitzar_objecte	`stat`	`stat`	–

Operació	Crida al SO	Ordre	Comentaris
Modificar_nom	`rename`	`mv`	–
Crear_nom	`link`	`ln`	Creen enllaços físics
	`symlink`	`ln -s`	Creen enllaços simbòlics
	mkdir	mkdir	Creen directoris
	mknod	mknod	Creen dispositius
Destruir_nom	unlink	rm	Esborra fitxers
	rmdir	rmdir	Esborra directori
Veure_noms	getdents	ls	Es veu el contingut d'un directori
Canvi_de_directori	chdir	cd	–
Directori_actual	getcwd	pwd	–

5.1.2. El mecanisme de protecció d'UNIX

Les proteccions en UNIX es basen en el model mixt que ja hem descrit. Nosaltres ens centrarem en les LCA i les operacions que s'ofereixen per a gestionar-les.

Dominis de protecció i bits de permís

En UNIX els dominis de protecció estan associats a usuaris (UID) i a grups d'usuaris (GID). Els grups d'usuaris són dominis en què s'organitzen els usuaris del sistema en funció de les seves característiques, de la feina que han de fer, etc. Un usuari s'identifica en el sistema mitjançant un nom d'usuari i una contrasenya. Un cop dins el sistema, tots els seus processos tindran associats els dominis d'usuari i del grup d'usuaris al qual pertanyi. En un moment determinat, un usuari només pot pertànyer a un grup; tanmateix, pot estar autoritzat a canviar de grup mitjançant l'ordre `newgrp`.

Vegeu també

Vegeu el model mixt en el subapartat 4.5 d'aquest mòdul didàctic.

Els dominis dels estudiants

Tots els estudiants que estiguin donats d'alta en un sistema tindran un domini d'usuari propi, i podrien estar organitzats en un domini conjunt anomenat *estudiants*.

Tots els fitxers estan associats al domini d'usuari i de grup d'usuaris del seu propietari. Per a determinar els drets d'un usuari sobre un fitxer, UNIX fa servir una LCA amb tres dominis: l'usuari propietari del fitxer, els usuaris que

pertanyen al mateix grup que el propietari i, finalment, qualsevol altre usuari. Per a cadascun preveu bàsicament tres drets diferents: de lectura, d'escriptura i d'execució.

La visualització de les LCA es fa mitjançant tres grups de tres caràcters cadascun, un per a cada dret, com es mostra en la figura 10. El primer grup fa referència al propietari; el segon, al grup del propietari, i el tercer, a la resta. Les lletres `r`, `w` i `x` (lectura, escriptura i execució) determinen si s'hi té el dret. L'ordre `chmod` permet al propietari concedir o revocar aquest drets.

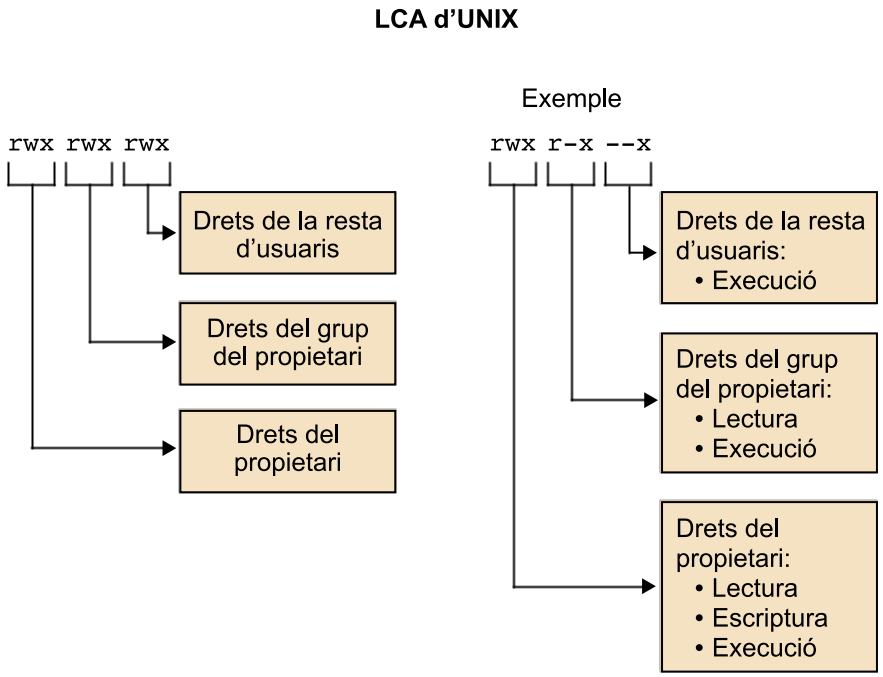

Figura 10

El dret d'execució pot ser reforçat amb el dret `setuid` o amb el dret `setgid`, els quals permeten que durant l'execució d'un programa el procés que l'executa canviï respectivament al domini de l'usuari propietari o al del grup de l'usuari propietari del fitxer executable. Aquests drets permeten construir aplicacions que accedeixin de manera controlada a bases de dades o a fitxers en general sobre els quals no es vol donar un dret d'escriptura generalitzat.

Com hem comentat en l'apartat 4.2, el significat dels drets `r`, `w` i `x` varia si el fitxer és un directori. En aquest cas, el dret de lectura permet visualitzar els noms continguts en el directori, el dret d'escriptura (condicionat a tenir també permís d'execució sobre el directori) permet afegir noms nous al directori, reanomenar fitxers o esborrar-los i, finalment, el dret d'execució fa possible que l'usuari utilitzi el directori com un directori de treball.

Per defecte, el valor dels bits assignats a un fitxer o directori en el moment de la seva creació correspon als bits resultants d'aplicar una màscara anomenada *umask* que forma part de l'entorn d'execució de cada procés i que pot ser con-

sultada o modificada per mitjà de l'ordre `umask` i la crida al sistema homònima. Posteriorment, els bits que controlen els permisos poden ser modificats, si es tenen drets per a fer-ho, mitjançant la crida i l'ordre `chmod`.

LCA de POSIX

L'estàndard POSIX defineix també LCA que són un superconjunt dels permisos especificats pels bits de permís associats als fitxers per als dominis propietari, grup i resta d'usuaris. Per una banda, hi ha una correspondència directa respecte als permisos indicats pels bits de permís per al propietari, el seu grup i la resta d'usuaris. Un canvi en un resulta en la modificació de les entrades de l'LCA de POSIX i viceversa. Però per una altra banda, les LCA de POSIX permeten especificar de manera molt més fina els permisos que el propietari d'un fitxer o directori vol concedir a altres usuaris o grups concrets.

5.1.3. Integració en el sistema de fitxers de dispositius i informació del sistema

Hi ha alguns objectes que apareixen a l'SF d'algunes variants del sistema operatiu UNIX, amb aparença de ser fitxers ordinaris però que en realitat no ho són. Per exemple, molts dispositius poden ser accedits i manipulats en UNIX per mitjà d'uns fitxers especials anomenats *fitxers de dispositiu* (*device files*). Aquests fitxers de dispositiu es troben al directori `/dev` i permeten accedir al codi gestor del dispositiu (*device driver*) emprant crides a sistema d'entrada/sortida. Es pot accedir a dispositius que existeixen físicament, com per exemple particions d'un disc (com pot ser `/dev/sda1`), una impressora (com pot ser `/dev/lp0`) o un terminal (`/dev/tty1`). I també es pot accedir a pseudodispositius que no existeixen físicament però que serveixen per a proporcionar una determinada funcionalitat. Per exemple, el dispositiu nul `/dev/null` que accepta i descarta tot allò que se li envia, o el dispositiu `/dev/zero`, que genera zeros.

Sovint hi ha el pseudosistema de fitxers `procfs`, que permet obtenir informació sobre els processos del sistema, i en alguns casos manipular-los. Per mitjà dels subdirectoris i fitxers que es troben a partir de `/proc` l'usuari pot accedir a molta informació sobre els processos que s'estan executant a la màquina. Cal notar que tots aquests directoris i fitxers no s'emmagatzemen en cap disc, sinó que es tracta d'informació que proporciona el sistema operatiu en el moment en què se sol·licita.

Un altre cas d'interès és el `sysfs`, que permet configurar diferents aspectes del sistema, principalment pel que fa a controladors (*drivers*) i dispositius per mitjà dels directoris que apareixen a partir de `/sys`. En aquest cas tampoc no es desa res a disc, sinó que tot està emmagatzemat a la memòria.

5.2. Windows

Hi ha diversos formats per a emmagatzemar la informació en un sistema Windows. El més habitual avui dia en els discos dels ordinadors que usen Windows és el format NTFS. Es continua usant bastant el format FAT, però sobretot per a memòries USB, ja que aquest format pot ser utilitzat en molts dispositius multimèdia. El format FAT és més simple que NTFS: permet mides de fitxers més petites que NTFS i no incorpora gaires mecanismes de protecció. Tot i que algunes característiques són comunes a tots dos formats, en aquest apartat assumirem que treballem amb NTFS.

A diferència d'UNIX, en el cas de Windows l'usuari no té la visió d'un únic sistema de fitxers. Hi ha un espai de noms lineal per a indicar diferents volums, en què un volum dóna accés a un dispositiu concret. Els noms possibles s'identifiquen amb una lletra majúscula seguida del caràcter :. Per exemple A:, C:, etc. L'usuari indica explícitament el volum amb què vol treballar.

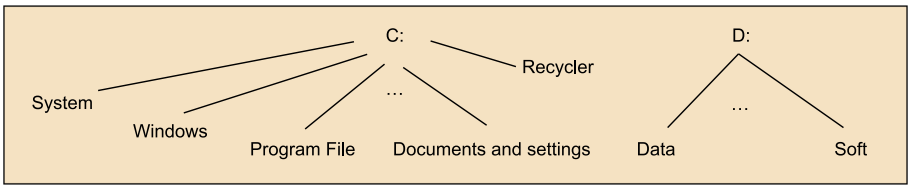

Figura 11

Els noms possibles per als fitxers i directoris són similars al cas d'UNIX. En Windows, però, els directoris se separen pel caràcter \. Per exemple, C:\Windows\Documents and Settings.

És possible crear enllaços físics (*hard links*) amb l'ordre `fsutil` o la crida `CreateHardLink` de l'API estàndard de Windows. Es poden crear enllaços simbòlics (*soft links*) creant accessos directes (*shortcuts*) emprant l'opció que apareix als menús contextuals.

El sistema de proteccions de fitxers i directoris (carpetes) està basat en LCA. Es pot manipular des de l'intèrpret d'ordres amb l'ordre `cacls` o per mitjà de la pestanya corresponent a la configuració de seguretat a partir de les propietats d'un fitxer o directori. L'administrador pot també definir quotes de disc per tal de limitar l'espai de disc màxim que un usuari pot arribar a ocupar.

NTFS incorpora també compressió i xifratge de fitxers, que es poden trobar accedint a les opcions avançades de les propietats dels fitxers. Altres aspectes inclosos són el tractament de fitxers dispersos per tal de reduir l'espai ocupat en disc quan hi ha cadenes molt llargues de zeros i la implementació de mecanismes de *journaling*[14] que milloren la consistència i recuperació del sistema de fitxers en cas de fallades o caigudes del sistema, com quan hi ha una interrupció del subministrament elèctric.

(14)Molts sistemes de fitxers de Linux també suporten *journaling*.

Resum

En aquest mòdul hem estudiat el concepte de **sistema de fitxers**, mitjançant el qual hem vist que el SO ofereix als usuaris els aspectes següents:

- El concepte de *fitxer*.
- La gestió dels noms dels fitxers.
- El mecanisme de protecció que tenen associat els fitxers.

Aquests dos últims punts els hem estès a tots els objectes que són gestionats pel SO. Respecte als fitxers, hem vist que el SO pot distingir els diferents tipus de fitxers, i que les operacions d'accés que s'hi poden fer no sempre són les mateixes.

Hem vist que la missió bàsica dels espais de noms és implementar una funció de traducció que permeti utilitzar noms estructurats i pròxims als nostres costums. Per a aconseguir-ho podem utilitzar bàsicament dos esquemes: l'**espai lineal** i l'**espai jeràrquic**. Aquest últim és el més utilitzat en la seva variant de graf dirigit.

Finalment, hem analitzat dos **mecanismes de protecció**: les llistes de control d'accessos (LCA) i les *capabilities*. Hem vist que els dos mecanismes presenten inconvenients, i que la millor solució és combinar-los, emprant l'LCA en obrir un fitxer i generant una *capability* associada al canal (*file descriptor*) obert per a comprovacions posteriors en fer operacions de lectura o escriptura.

Hem vist també que per mitjà del sistema de fitxers de vegades es pot accedir a certs dispositius, a informació sobre els processos i fins i tot a configurar el sistema.

Finalment, hem comentat alguns aspectes concrets dels sistemes de fitxers més habituals dels sistemes UNIX i Windows.

Activitats

1. Estudieu en el manual quins tipus de SF suporta Linux.

2. Analitzeu i proveu les ordres d'UNIX que hem presentat en aquest mòdul.

3. Penseu quines crides a l'SO hem d'utilitzar i com hem de fer les ordres analitzades a l'activitat anterior.

4. Com a usuaris de Windows, analitzeu quin espai de noms i quines operacions teniu per accedir al sistema de fitxers.

5. Proveu a crear un directori i algun fitxer a dintre. Emprant l'ordre `chmod` activeu i desactiveu els bits de protecció `r`, `w` i `x` del directori per al propietari. Per a cada combinació de valors intenteu llegir el directori, veure el contingut d'un fitxer contingut al directori del qual coneixem el nom, crear nous fitxers, moure'ls o esborrar-los, o canviar el directori de treball a aquell directori.

6. En un sistema Linux consulteu la informació disponible sobre les LCA de POSIX al manual en línia fent `man acl`.

7. En un sistema Windows i treballant com a administrador, definiu un nou usuari a partir del tauler de control i comptes d'usuari. Canvieu d'usuari anant al menú d'inici i anant a la part de canvi d'usuari. Per defecte el directori de connexió propi de cada usuari està protegit de manera que la resta d'usuaris no hi poden accedir. Si l'administrador desactiva l'opció *Use simple file sharing* de les opcions de carpeta, llavors els usuaris poden activar o desactivar els permisos d'accés per a usuaris concrets a través de la pestanya corresponent a la configuració de seguretat a partir de les propietats d'un fitxer o directori.

Exercicis d'autoavaluació

1. Quines són les característiques o propietats principals d'un fitxer? Comenteu quines poden ser les funcions.

2. Basant-vos en el que hem vist en aquest mòdul didàctic i en la vostra experiència, digueu quin tipus d'espai de noms tenen els sistemes de fitxers dels SO següents:

a) Windows.

b) UNIX.

3. Quin tipus d'enllaç representen els fitxers que tenen la característica "accés directe" de Windows? Justifiqueu la resposta.

4. En MS-DOS el directoris, a part de la funció de traducció, contenien informació pel que fa a les característiques dels fitxers a què fan referència, com ara el nombre de caràcters que conté el fitxer. Quins inconvenients podia tenir aquest fet?

5. Quin tipus d'espai de noms creieu que és el més adequat per als tipus d'objectes següents?

a) Fitxers.

b) Dispositius físics o lògics.

c) Dispositius virtuals.

d) Processos.

e) Usuaris.

6. Com seria la matriu d'accessos següent si...

a) cada usuari només pogués accedir als seus fitxers amb tots els drets (lectura, escriptura i execució)?

b) tots els usuaris poguessin accedir amb tots els drets a tots els fitxers?

Dominis	**Objectes**							
	Fitxer A	**Fitxer B**	**Fitxer C**	**Fitxer D**	**Fitxer E**	**Fitxer F**	**Fitxer G**	**Fitxer H**
Joan	Propietari	Propietari						
Maria			Propietari	Propietari				
Josep					Propietari	Propietari		
Marta							Propietari	Propietari

7. En els dos casos de l'exercici 6, cal tenir un mecanisme de protecció? I en cas que faci falta, té sentit el dret de propietari? Justifiqueu la resposta.

8. Digueu ens quins moments d'una sessió de treball amb un fitxer es verifiquen els drets d'accés si tenim un sistema operatiu amb un sistema de protecció basat en...

a) LCA.

b) llistes de *capabilities*.

Justifiqueu les respostes.

9. Quines restriccions té el mecanisme de protecció d'UNIX basat en dominis de protecció i bits de permís respecte d'un mecanisme d'LCA en què cada usuari constitueix en si mateix un domini? Justifiqueu la resposta.

Solucionari

Exercicis d'autoavaluació

1. Un fitxer té un conjunt de característiques o propietats que fan referència als aspectes següents:

a) La informació relativa al contingut del fitxer i a la seva modificació: grandària del fitxer, data de creació, última data d'accés, última data de modificació, tipus d'informació, etc. La seva funció és bàsicament estadística, a excepció de la grandària del fitxer i del tipus d'informació. El sistema utilitza la grandària del fitxer per a determinar quan ha de donar la marca de final de fitxer durant els accessos de lectura. El tipus d'informació que conté el fitxer pot condicionar les operacions que s'hi poden fer.

b) La ubicació d'aquesta informació dintre del dispositiu d'emmagatzematge. Conté informació necessària perquè el sistema pugui localitzar el fitxer dintre del dispositiu d'emmagatzematge.

c) L'accessibilitat del fitxer. Conté informació relacionada amb l'usuari propietari del fitxer i les operacions que s'hi poden fer: escriure, llegir, executar, etc. Totes aquestes informacions estan relacionades amb el mecanisme de protecció.

2.a) Windows té un espai de noms en forma de graf dirigit en què es poden formar cicles, perquè un fitxer pot tenir més d'un nom gràcies als enllaços. L'espai és disjunt: cada dispositiu que conté un sistema de fitxers té un espai de noms separat.

b) UNIX té un espai de noms amb característiques similars a les de Windows, amb la diferència que en UNIX l'espai no és disjunt, sinó que hi ha un sol espai que uneix tots els dispositius actius que contenen un sistema de fitxers.

3. Els fitxers d'accés directe de Windows presenten enllaços simbòlics. Aquests enllaços fan referència a un fitxer mitjançant un altre nom de l'espai de noms. Quan s'esborra el fitxer per mitjà d'un enllaç físic, els accessos directes es mantenen, ja que no es poden localitzar fàcilment a partir del fitxer esborrat. Quan s'accedeix pel nom d'accés directe, el sistema s'adona que ja no existeix el fitxer i avisa d'aquesta circumstància.

4. El principal inconvenient d'aquest fet és que es barreja el concepte de funció de traducció amb l'estructura del fitxer. Això fa que l'espai de noms perdi flexibilitat. Amb un sistema d'aquestes característiques no es podia assignar més d'un nom a un fitxer, ja que en cas de fer-ho es duplicaria la informació, amb el risc o el cost de mantenir-la coherent. Per exemple, en MS-DOS es desava el nombre de caràcters en el directori. Si tenim dos noms per al mateix fitxer, tindrem dues còpies d'aquest nombre, i si aquest s'actualitza en funció del nom pel qual s'accedeix al fitxer, ens podem trobar amb informacions incoherents.

5. Totes les decisions dependran en gran manera del volum d'objectes que es vol adreçar mitjançant els espais de noms i del tipus d'usuari d'aquests objectes.

a) Fitxers: tal com hem vist en el mòdul, el més adequat és un espai jeràrquic en forma de graf, que dóna la màxima flexibilitat a l'hora d'organitzar els objectes. Els noms han de ser formats per cadenes de caràcters a fi que es puguin adaptar a les nostres necessitats, ja que, en general, els usuaris som les persones, i el volum de fitxers que es vol adreçar és important.

b) Dispositius físics o lògics: en aquest cas pot ser suficient un espai amb estructura en arbre o un espai lineal, ja que les necessitats d'organització dels objectes no són tan importants com en el cas anterior, perquè el volum d'objectes és molt més petit. Com que els usuaris som les persones, igual que en el cas anterior, seria convenient que els noms els formessin cadenes de caràcters.

c) Dispositius virtuals: l'espai més indicat és l'espai lineal amb noms senzills, com ara nombres, a causa del fet que els usuaris d'aquest tipus de dispositius són els processos; a més a més, els dispositius virtuals que són visibles des d'un procés no necessiten ser visibles des dels altres. Es tracta d'un conjunt d'objectes reduït, i amb unes necessitats d'organització molt baixes.

d) Processos: amb un espai lineal amb noms senzills, com ara nombres, n'hi ha prou, ja que, com en el cas anterior, ens trobem amb un conjunt d'objectes adreçats quasi exclusivament pel SO o pels processos. El nombre d'objectes pot ser important, però no tenen necessitats d'organització.

e) Usuaris: el més adequat és un espai lineal. Els noms haurien de ser formats per cadenes de caràcters en consideració dels usuaris humans, ja que en aquest cas els usuaris són els

objectes domini i els usuaris d'aquests objectes domini poden ser les persones i el sistema. El nombre d'objectes pot ser molt gran, però la necessitat d'organització és petita.

Finalment, s'ha de tenir present que, en tots els casos en què s'ha proposat un espai amb noms formats per cadenes de caràcters, és necessari disposar d'un segon espai de noms intern, generalment lineal i amb noms basats en nombres, pròxim a les necessitats de gestió del SO. Els noms d'aquest espai intern són aquells als quals fa referència la funció de traducció un cop traduït un nom de l'espai extern.

6.a) Si cada usuari només pogués accedir als seus fitxers amb tots els drets (lectura, escriptura i execució), la matriu d'accessos seria la següent:

Dominis	**Objectes**							
	Fitxer A	**Fitxer B**	**Fitxer C**	**Fitxer D**	**Fitxer E**	**Fitxer F**	**Fitxer G**	**Fitxer H**
Joan	Propietari Lectura Escriptura Execució	Propietari Lectura Escriptura Execució						
Maria			Propietari Lectura Escriptura Execució	Propietari Lectura Escriptura Execució				
Josep					Propietari Lectura Escriptura Execució	Propietari Lectura Escriptura Execució		
Marta							Propietari Lectura Escriptura Execució	Propietari Lectura Escriptura Execució

b) Si tots els usuaris poguessin accedir amb tots els drets a tots els fitxers, la matriu d'accessos seria la següent:

Dominis	**Objectes**							
	Fitxer A	**Fitxer B**	**Fitxer C**	**Fitxer D**	**Fitxer E**	**Fitxer F**	**Fitxer G**	**Fitxer H**
Joan	Propietari Lectura Escriptura Execució	Propietari Lectura Escriptura Execució	Lectura Escriptura Execució	Lectura Escriptura Execució	Lectura Escriptura Execució	Lectura Escriptura Execució	Lectura Escriptura Execució	Lectura Escriptura Execució
Maria	Lectura Escriptura Execució	Lectura Escriptura Execució	Propietari Lectura Escriptura Execució	Propietari Lectura Escriptura Execució	Lectura Escriptura Execució	Lectura Escriptura Execució	Lectura Escriptura Execució	Lectura Escriptura Execució
Josep	Lectura Escriptura Execució	Lectura Escriptura Execució	Lectura Escriptura Execució	Lectura Escriptura Execució	Propietari Lectura Escriptura Execució	Propietari Lectura Escriptura Execució	Lectura Escriptura Execució	Lectura Escriptura Execució
Marta	Lectura Escriptura Execució	Lectura Escriptura Execució	Lectura Escriptura Execució	Lectura Escriptura Execució	Lectura Escriptura Execució	Lectura Escriptura Execució	Propietari Lectura Escriptura Execució	Propietari Lectura Escriptura Execució

7.a) En el cas que cada usuari només pugui accedir als seus fitxers amb tots els drets (lectura, escriptura i execució), efectivament cal un mecanisme de protecció. Seria tan restrictiu com

fos possible: no hi ha cap possibilitat de compartir fitxers entre usuaris. Per tant, el dret de propietari està implícit en l'estructura del mecanisme proposat i, per consegüent, la seva existència explícita no és important.

b) En el cas que tots els usuaris puguin accedir amb tots els drets a tots els fitxers, no hi ha cap mecanisme de protecció, ja que tots els usuaris poden fer el que vulguin sobre qualsevol fitxer. Així, com que no cal cap mecanisme de protecció, tampoc no es necessita el dret de propietari.

8.a) En el cas de l'LCA: en aquest cas la verificació de drets es fa en cada accés al dispositiu, ja que la informació de protecció es troba físicament en el dispositiu. Això vol dir que, en un SO amb un mecanisme de protecció basat exclusivament en LCA, tots els accessos al fitxer d'una sessió de treball han de ser verificats.

b) En el cas de les llistes de c*apabilities*: en aquest cas la verificació de drets està associada al procés que vol efectuar els accessos. Aquesta verificació es fa d'acord amb l'existència o no de la *capability* associada a l'objecte. Així, doncs, en un SO basat exclusivament en *capabilities*, en una sessió de treball només s'ha de verificar el dret d'accés en l'operació d'obrir, que és l'encarregada de verificar l'existència de la *capability*. De tota manera, es pot generar una nova *capability* i associar-la al canal per tal que es pugui comprovar ràpidament si es té dret a llegir o escriure sobre el canal quan s'intenta fer l'operació en qüestió.

9. En un mecanisme com el d'UNIX els dominis que recullen les LCA s'han restringit als tres següents: propietari, grup del propietari i la resta d'usuaris. Aquesta simplificació de tots els dominis possibles permet a UNIX fer una gestió de les proteccions més eficient respecte de l'espai i del temps de gestió. No obstant això, aquesta simplificació restringeix la flexibilitat a l'hora d'especificar els drets individuals de cada usuari. Per exemple, amb UNIX no és possible distingir entre els usuaris que pertanyen al grup del propietari i, per tant, no els podem donar drets diferents. Passa el mateix amb els usuaris que pertanyen al domini "la resta d'usuaris". De totes maneres, aquesta restricció es pot suavitzar amb una bona política de definició de grups d'usuaris, ja que un usuari d'UNIX pot estar autoritzat, dintre un conjunt predeterminat, a canviar de grup d'usuaris.

Glossari

directori *m* Conjunt de fitxers on s'emmagatzema l'estructura de dades que dóna lloc a la funció de traducció.

directori de treball *m* Directori on es troben els fitxers amb els quals un usuari està treballant en un instant concret. Quan un usuari comença la sessió de treball, el directori de treball és el directori inicial.

directori inicial *m* Directori des d'on un usuari por crear la seva estructura de directoris i on col·locarà els fitxers que ell creï.

domini *m* Agents actius del sistema que poden actuar sobre un objecte.

dret *m* Conjunt d'accions permeses per part d'un domini sobre un objecte.

enllaç físic *m* Enllaç que relaciona directament un nom del sistema de fitxers amb l'objecte al qual fa referència. O, el que és el mateix, relaciona directament un nom del sistema de fitxers amb el nom intern del SO de l'objecte al qual fa referència.
en hard link

enllaç simbòlic *m* Enllaç que no relaciona directament un nom del sistema de fitxers amb un objecte, sinó que ho fa indirectament mitjançant un enllaç a un altre nom del sistema de fitxers des del qual es pot localitzar l'objecte. Aquest tipus d'enllaços s'utilitzen fonamentalment per a donar un nom a un objecte que es troba en un dispositiu d'emmagatzematge diferent d'aquell on es troba el directori que conté l'enllaç simbòlic.
en symbolic link o soft link

espai de noms *m* Espai configurat pel conjunt de tots els noms possibles d'un objecte.

espai de noms jeràrquic *m* Espai format per una jerarquia de directoris, amb un directori arrel del qual pengen altres directoris o fitxers que configuren les fulles de l'arbre. Els espais jeràrquics poden tenir forma d'arbre o, més generalment, de graf dirigit.

espai de noms lineal *m* Espai de noms, amb una sola dimensió, en què tots els noms són al mateix nivell, en un únic directori. En aquest tipus d'espai no es poden fer classificacions entre els diferents objectes.

fitxer *m* Dispositiu lògic format per una agrupació lògica d'informació emmagatzemada en un dispositiu físic, com ara un disc, un llapis USB o la memòria, i que pot ser manipulada com un tot. La informació que inclou té en comú un conjunt de propietats que la caracteritzen.

funció de traducció *f* Funció unívoca que relaciona un nom amb un objecte.

hard link *m* Vegeu **enllaç físic**.

journaling *m* Tècnica emprada en sistemes de fitxers per a evitar la corrupció de l'SF. Es basa a desar informació en un lloc determinat, sigui intern o extern a l'SF, sobre tot un conjunt de canvis que es faran sobre l'SF. Aquesta informació es desa abans de fer els canvis. Això permet que en cas que es produeixi una fallada es pugui detectar el punt en què s'ha produït i es pugui recuperar la consistència de l'SF.

LCA *f* Vegeu **llista de control d'accessos**.

llista de *capabilities* *f* Llista de parelles (*objecte*, *drets*) associada a un domini. Cadascuna d'aquestes parelles s'anomena *capability*. Les llistes de *capabilities* són el resultat de partir la matriu d'accessos en files i eliminar, igual que en les llistes de control d'accés, totes les cel·les buides.

llista de control d'accessos *f* Llista de parelles (*drets*, *domini*) associada a un objecte. Aquesta llista és el resultat de partir la matriu d'accessos en columnes i eliminar totes les cel·les buides.
sigla **LCA**

matriu d'accessos *f* Matriu formada per tantes files com dominis (D_i), i tantes columnes com objectes (O_j), en què cada cel·la D_{ij} conté els drets d'accés del domini D_i sobre l'objecte O_j.

nom absolut *m* Nom format per la ruta que va des de l'arrel, passant pels diferents subdirectoris, fins a arribar al fitxer. Així, doncs, el nom absolut està compost per una seqüència de cadenes de caràcters, o components del nom, contingudes en els subdirectoris que configuren la ruta. Els components estan separats per un caràcter delimitador.

nom relatiu *m* Noms formats pel recorregut que va des del directori de treball fins al fitxer. Diversos fitxers poden tenir el mateix nom relatiu, sempre que aquest s'obtingui a partir de directoris de treball diferents.

objecte *m* Element gestionat pel SO sobre el qual es poden portar a terme accions en funció d'uns drets.

pany i clau *m* Mecanisme de protecció basat en l'ús d'unes parelles de nombres anomenades *pany* i *clau* que han de coincidir a fi de donar per vàlid un dret. Si aquests nombres no coincideixen el dret queda invalidat.

symbolic link *m* Vegeu **enllaç simbòlic**.

sistema de fitxers *m* Sistema encarregat de gestionar el conjunt de fitxers continguts en un mateix dispositiu d'emmagatzematge.

soft link *m* Vegeu **enllaç simbòlic**.

Bibliografia

Bibliografia bàsica

Silberschatz, A.; Galvin, P.; Gagne, G. (2008). *Operating Systems Concepts* (8a. ed.). John Wiley & Sons.

Tanembaum, A. (2009). *Modern Operating Systems*. Prentice Hall.