ЛЕКЦІЯ 16. ФАЙЛОВА СИСТЕМА LINUX (ПРОДОВЖЕННЯ)

Розміщення компонентів системи: стандарт FHS

Спробуємо розібратися, як побудоване дерево *каталогів* Linux, де й що в ньому можна знайти. Фрагмент дерева *каталогів* типової *файлової системи* Linux наведений на рис. 16.1.

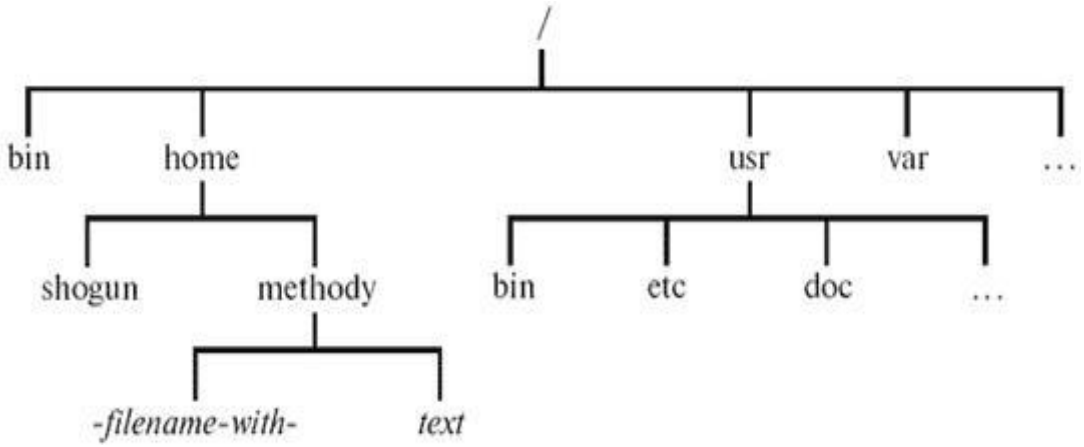

Рис 16.1. Фрагмент дерева каталогів Linux

Обстежимо файлову систему, починаючи з кореневого каталогу: використовуємо для цього команду ls /, де каталог - це повний шлях до каталогу: утиліта ls виведе список усього, що в цьому каталозі втримується. Цей список буде приблизно таким же в будь-якому дистрибутиві Linux. У кореневому каталозі Linux-системи звичайно знаходяться тільки підкаталоги зі стандартними іменами. Більше того, не тільки імена, але й тип даних, які можуть потрапити в той або інший каталог, також регламентовані цим стандартом. Даний стандарт називається Filesystem Hierarchy Standard ("стандартна структура файлових систем").

В табл. 16.1 наведено опис підкаталогів кореневого каталогу.

Таблиця 16.1. Підкаталоги кореневого каталогу

Каталог	Опис
/bin	Назва цього каталогу походить від слова "binaries". У цьому каталозі знаходяться файли, самих необхідних утиліт. Сюди попадають такі програми, які можуть знадобитися системному адміністраторові або іншим користувачам для усунення неполадок у системі або при відновленні після збою.
/boot	"boot" – завантаження системи. У цьому каталозі знаходяться файли, необхідні для найпершого етапу - завантаження ядра - і, звичайно, саме ядро. Користувачеві практично ніколи не потрібно безпосередньо працювати із цими файлами.
/dev	У цьому каталозі знаходяться всі наявні в системі файли

	призначені для роботи з різними системними ресурсами і пристроями (англ. "devices" - "пристрої", звідси й скорочена назва каталогу). Наприклад, файли /dev/ttyN відповідають віртуальним консолям, де N - номер віртуальної консолі. Дані, введені користувачем на першій віртуальній консолі, система зчитує з файлу /dev/tty1; у цей же файл записуються дані, які потрібно вивести користувачеві.
/etc	Каталог для системних конфігураційних файлів. Тут зберігається інформація про специфічні настроювання даної системи: інформація про зареєстрованих користувачів, доступні ресурси, настроювання різних програм.
/home	Тут розташовані каталоги, що належать користувачам системи – домашні каталоги, звідси й назва "home". Відділення всіх файлів, що створюються користувачами, від інших системних файлів дає очевидну перевагу: серйозне ушкодження системи або необхідність відновлення не торкнеться найціннішої інформації - користувальницьких файлів.
/lib	Назва цього каталогу – скорочення від "libraries" (англ. "бібліотеки"). Бібліотеки - це набір стандартних функцій, необхідних багатьом програмам: операцій вводу/виводу, малювання елементів графічного інтерфейсу й т.д. Щоб не включати ці функції в текст кожної програми, використовуються стандартні функції бібліотек - це значно заощаджує місце на диску й спрощує написання програм. У цьому каталозі знаходяться бібліотеки, необхідні для роботи найбільш важливих системних утиліт (розміщених в /bin і /sbin).
/media	Каталог для монтування (від англ. "mount") - тимчасового підключення файлових систем, наприклад, на знімних носіях (CD-ROM і др.).
/proc	У цьому каталозі всі файли “віртуальні” - вони розташовуються не на диску, а в оперативній пам’яті. У цих файлах знаходиться інформація про програми (процеси), що виконуються у цей момент у системі.
/root	Домашній каталог адміністратора системи - користувача root. Розміщення його окремо від домашніх каталогів інших користувачів необхідно тому, що /home може розташовуватися на окремому пристрої, що не завжди доступно (наприклад, на мережному диску), а домашній каталог root повинен бути присутнім у будь-якій ситуації.
/sbin	Каталог для найважливіших системних утиліт (назва каталогу – скорочення від "system binaries"): на додаток до утиліт /bin тут знаходяться програми, необхідні для завантаження, резервного копіювання, відновлення системи. Повноваження на виконання цих програм є тільки в системного адміністратора.
/tmp	Цей каталог призначений для тимчасових файлів: у таких

	файлах програми зберігають необхідні для роботи проміжні дані. Після завершення роботи програми тимчасові файли втрачають сенс і повинні бути видалені. Як правило, каталог /tmp очищається при кожному завантаженні системи.
/usr	Каталог /usr- це "держава в державі". Тут можна знайти такі ж підкаталоги bin, etc, lib, sbin, як і в кореневому каталозі. Однак у кореневий каталог попадають тільки утиліти, необхідні для завантаження й відновлення системи в аварійній ситуації, а всі інші програми й дані розташовуються в підкаталогах /usr. Прикладних програм у сучасних системах звичайно встановлено дуже багато, тому цей розділ файлової системи може бути дуже великим.
/var	Назва цього каталогу – скорочення від "variable" ("змінні" дані). Тут розміщаються ті дані, які створюються в процесі роботи різними програмами й призначені для передачі іншим програмам і системам (черги друку, електронної пошти та ін.) або для відомості системного адміністратора (системні журнали, що містять протоколи роботи системи). На відміну від каталогу /tmp сюди попадають ті дані, які можуть знадобитися після того, що як створила їхня програма завершила роботу.

Стандарт *FHS* регламентує не тільки перераховані *каталоги* , але і їхні *підкаталоги* , а іноді навіть приводить список конкретних *файлів*, які повинні бути присутнім у певних каталогах. Цей стандарт послідовно дотримується у всіх Linux-системах.

Стандартне розміщення *файлів* дозволяє й людині, і навіть програмі визначати, де перебуває той або інший компонент системи. Для людини це означає, що він зможе швидко зорієнтуватися в будь- якій системі Linux (де *файлова система* організована у відповідності зі стандартом) і знайти те, що йому потрібно. Для програм стандартне розташування *файлів* - це можливість організації автоматичної взаємодії між різними компонентами системи.

Ми вже встигли скористатися деякими перевагами, які дає стандартне розташування файлів: на попередніх лекціях ми запускали утиліти, не вказуючи повний шлях до файлу, що виконується, наприклад, cat замість /bin/cat. Командна оболонка "знає", що файли, що виконуються, розташовуються в каталогах /bin, /usr/bin і т.д. - саме в цих каталогах вона шукає програму cat. Завдяки цьому кожна знову встановлена в системі програма негайно виявляється доступна користувачеві з командного рядка. Для цього не потрібно не перезавантажувати систему, не запускати які-небудь процедури - досить просто помістити файл, що виконується, в один з відповідних каталогів.

Рекомендації стандарту по розміщенню файлів і каталогів ґрунтуються на принципі розміщення файлів, які по-різному використовуються в системі, у різних підкаталогах. По типу використання файли можна розділити на наступні групи:

1. користувальницькі/системні файли;

2. що змінюються/незмінні файли
3. поділювані/неподілювані файли

Користувальницькі *файли* - це всі *файли*, створені користувачем і не приналежні жодному з компонентів системи. Про користь розмежування користувальницьких і системних *файлів* мова вже йшла вище.

До незмінених *файлів* ставляться всі статичні компоненти програмного забезпечення: бібліотеки, програми й т.д. - усе, що не змінюється саме без втручання системного адміністратора. Ф*айли, що* змінюються, - це ті, які змінюються без втручання людини *в* процесі роботи системи*: системні* журнали, черги друку та ін. Виділення незмінних *файлів* в окрему структуру (наприклад, /usr) дозволяє використовувати
відповідну частину *файлової системи* в режимі "тільки читання", що зменшує ймовірність випадкового ушкодження даних і дозволяє застосовувати для зберігання цієї частини *файлової системи CD*-ROM і інші носії, доступні тільки для читання.

Це розмежування стає корисним, якщо мова йде про мережу, у якій працює кілька комп'ютерів. Значна частина інформації при цьому може зберігатися на одному з комп'ютерів і використовуватися всіма іншими по мережі (до такої інформації відносяться, наприклад, багато програм і *домашні каталоги* користувачів). Однак частина *файлів* не можна розділяти між системами (наприклад, *файли* для початкового завантаження системи).

Команди архівування файлів

Основним засобом архівування в UNIX (а, отже, і в Linux) є комплекс із двох програм — tar і gzip. Саме в такому форматі поширюється більша частина програмного забезпечення для Unix.

Програма tar

Назва tar розшифровується як Tape ARchiver, він не стискає дані, а лише поєднує їх у єдиний файл із послідовним доступом для наступного запису на стрічку. За замовчуванням цей архівний файл створюється на стрічковому накопичувачі, точніше на пристрої /dev/rmt0. Якщо ви хочете створити архівний файл на диску, то необхідно використовувати команду tar з опцією f, після якої вказується ім'я архівного файлу.

У програми tar є 8 опцій, що відрізняються від інших тим, що при виклику програми повинна обов'язково задаватися одна із цих опцій. Ці опції визначають основні функції програми.

Таблиця 16.2. Основні опції програми tar

Опція	Значення
-A, --concatenate	Додає файли в існуючий архів
-c, -create	Створює новий архів
-d, -diff,--compare	Знайти розходження між архівом і файловою системою
--delete	Видалити із архіву
-r, -append	Додає файли в кінець архіву

-t, -list	Виводить список файлів архіву
-u, -update	Додає тільки файли, які новіші, ніж наявна в архіві копія
-x, -extract,--get	Витягти файли з архіву

Якщо ви працюєте з файлами архівів на дисках, а не зі стрічковим пристроєм, то, обов'язкової буде й опція f. Інші опції не є обов'язковими, вони служать тільки для конкретизації завдання програмі. Наприклад, опція v змушує програму виводити список оброблюваних файлів. Однобуквені опції програми tar можуть перераховуватися один за одним. Наведемо кілька командних рядків для виконання самих необхідних дій з архівами. Щоб створити один tar-архів з декількох файлів, використовується команда:

[user]$ tar -cf ім'я_архіву файл1 файл2 ...,

де опція -c повідомляє програму, що необхідно створити (create) архів, а опція f говорить про те, що архів повинен створюватися у вигляді файлу (ім'я якого повинне йти відразу за цією опцією). В іменах файлів, які зберігаються в архіві, можна використовувати шаблони імен файлів, у тому числі просто символи-шаблони * і ?. Завдяки цьому можна дуже короткою командою відправити в архів відразу багато файлів. Наприклад, для того, щоб створити архів, що містить всі файли одного з підкаталогів (нехай це буде sub_dir) поточного каталогу, досить дати команду:

[user]$ tar -cvf ім'я_архіву ./sub_dir/*

або навіть просто **[user]$ tar -cvf ім'я_архіву sub_dir**

Програма gzip

Хоча програма tar створює архіви, вона, як було сказано, не стискає архіви, а просто з'єднує окремі файли в єдиний архівний файл. Для стискання цього файлу часто застосовують команду gzip. У найпростішому випадку вона викликається в наступному форматі: **[user]$ gzip файл**

У командному рядку можна вказати відразу кілька імен файлів або шаблон ім'я файлу. Але в цьому випадку кожний із зазначених файлів буде заархівований окремо (загальний архів не створюється). Для того, щоб розпакувати архів, використовуйте команду: **[user]$ gzip -d файл_архіву** або **[user]$ gunzip файл_архіву**

Вихідні файли після стискання видаляються, залишається тільки архівний файл (файли переміщаються в архів), а при розархивуванні видаляється архів. В табл. 16.3 перелічені корисні опції програми gzip.

Таблиця 16.3. Опції програми gzip

Опція	Значення
-h, -help	Виводить короткої справку по використанню програми
-l, -list	Виводить ім'я файлу, що знаходиться в архіві, його обсяг і ступінь стиску
-L, -license	Відображає номер версії й ліцензію на програму
-N, -name	Зберігає (або відновлює) вихідне ім’я й час створення файлу
-q, ---quiet	Подавляє видачу на екран попереджуючих повідомлень

-r, ---recursive	Рекурсивно обробляє підкаталоги (використовується у випадку. коли заданий шаблон імен оброблюваних файлів)
-S .suf, ---isuffix .suf	Додає суфікс .suf до імені стислого файлу (замість суфікса, що додається по замовчанню - gz)
-t, ---test	Протестувати архівний файл
-v, ---verbose	Вивід додаткових повідомлень у процесі роботи програми
-V, ---version	Відобразити версію програми
-1, ---fast	Найменший рівень стиску
-9, ---best	Найбільший рівень стиску

Оскільки програма gzip не вміє зберігати в одному архіві кілька файлів, то звичайно неї застосовують для стиску архівів, створених програмою tar. Більше того, серед опцій програми tar є спеціальна опція -z, що дозволяє відразу після створення стиснути його за допомогою програми gzip. Для виконання такого стиску треба використовувати команду
tar приблизно в такий спосіб:

[user]$ tar -czf ім'я_архіву шаблон_імен_файлів

Програма bzip2

Останнім часом всі частіше замість програми gzip використовується архіватор bzip2, що забезпечує більше високий ступінь стиску й працює трохи швидше. Працює bzip2 приблизно так само, як команда gzip, тобто заміщає кожний файл, ім'я якого задано в командному рядку, стислим версією, додаючи до ім'я файлу суфікс .bz2.
Стислий файл має той же самий час модифікації, права доступу й, по можливості, того ж власника, що й вихідний файл, що дає можливість відновити ці атрибути при добуванні файлів з архіву. Команда bunzip2 (або bzip2 -d) розпаковує зазначені в командному рядку файли. Опції командного рядка для bzip2 дуже схожі на опції команди gzip, але все-таки вони не ідентичні. Детальну інформацію про опціїї bzip2 можна отримати за допомогою команди man.

Створення й монтування файлових систем

У попередніх розділах ми коротко розглянули основні команди для роботи із уже сформованою файловою системою. Тепер треба зупинитися на питанні про те, як створити файлову систему й модифікувати її.

В ОС Linux прийнята концепція, у якій зовнішнє устаткування (дискети, жорсткі диски, SSD носії, CD/DVD дисководи, флеш пам'ять і т.п.) розглядаються як файлові пристрої з певною структурою файлів для читання й запису. В табл 3.4 наведено відповідність імен пристроїв в Windows та Linux.

Таблиця 16.4. Типовий состав зовнішніх дискових пристроїв і їхні імена

Назва пристрою в ОС Windows	Тип пристрою	Назва пристрою в ОС Linux	Примітки
A	Флоппі диск	/dev/fd0	

C	1-й жорсткий диск (master)	/dev/sda	На 1-му IDE каналі
D	2-й жорсткий диск (slave)	/dev/sdb	На 1-му IDE каналі
E	3-й жорсткий диск (master)	/dev/sdc	На 2-му IDE каналі
F	4-й жорсткий диск (slave)	/dev/sdd	На 2-му IDE каналі
G	SCSI CD-ROM	/dev/scd0	

Загальне дерево файлів і каталогів системи Linux формується з окремих "гілок", що відповідають різним фізичним носіям. Часто говорять, що воно формується з окремих файлових систем. Говорити так дозволяє той факт, що в UNIX немає поняття "форматування диска" (і команди форматування), а використовується поняття "створення файлової системи". Коли ми одержуємо новий носій, наприклад, жорсткий диск, ми повинні створити на ньому файлову систему. Тобто кожному носію ставиться у відповідність окрема файлова система. Щоб цю файлову систему використовувати для запису в неї файлів, треба її спочатку підключити в загальне дерево каталогів ("змонтувати").

Можна також відзначити, що звичайно жорсткий диск попередньо розбивається на розділи (особливо сучасні диски, що мають ємність, обчислювальну десятками гігабайт). Створення розділів полегшує виконання резервного копіювання, рішення завдань розмежування повноважень, підвищує продуктивність. Тому надалі будемо говорити про створення файлової структури в одному розділі (диск, що не має розділів, можна теж розглядати як один розділ).

Ще один момент, істотний у контексті цього розділу, пов'язаний з тим, що Linux може працювати з різними типами файлових систем. "Рідною" файловою системою для нього в цей час є "друга розширена файлова система" (second extended filesystem) ext2fs або "третя розширена файлова система" (third extended filesystem) ext3fs. Но в Linux можна працювати й з 16-розрядною файловою системою FAT, створюваної в MS-DOS, і з 32-розрядної FAT32, розробленої для MS Windows 95, і з файловою системою ISO9660, використовуваної для запису інформації на CD-ROM, і з іншими типами файлових систем. Тобто, при розгляді питань створення й монтування файлових систем треба постійно пам'ятати про те, що типи файлових систем на різних носіях можуть розрізнятися.

Отже, спочатку розглянемо випадок, коли потрібно створити в якомусь розділі диска файлову систему. Будемо припускати, що створюється файлова система типу ext2fs (створення файлових систем інших типів — тема для книг, присвячених іншим операційним системам). Створення файлової системи типу ext2fs має на увазі створення в даному розділі на диску суперблоку, таблиці індексних дескрипторів і сукупності блоків даних. Робиться все це все за допомогою команди mkfs. У найпростішому випадку досить дати цю команду в наступному форматі:

 # mkfs -t ext2 /dev/sda5

де /dev/sda5 - відповідний пристрій або розділ. Наприклад, якщо ви хочете створити файлову систему на дискеті, то команда прийме вид:

 # mkfs -t ext2 /dev/fd0

Після виконання команди mkfs у зазначеному розділі буде створена файлова система ext2fs. У новій файловій системі автоматично створюється один каталог з ім'ям lost+found. Він використовується в екстрених випадках програмою fsck, тому не видаляйте його. Для того, щоб почати працювати з новою файловою системою (наприклад, переписати якісь файли на новий носій), необхідно підключити її в загальне дерево каталогів, що робиться за допомогою команди mount.

Відзначимо, що після монтування файлової системи, наприклад, в каталог /media/disk2 колишній зміст цього каталогу стане для вас недоступним (так само, як інформація про колишнього власника й права доступу до цього каталогу) доти, поки ви не размонтуєте знову підключену файлову систему. Колишній зміст не знищується, а просто стає тимчасово недоступним. Тому як точки монтування краще використовувати порожні каталоги (заздалегідь заготовлені).

Команда mount також може використовувати інформацію з файлу /etc/fstab, в якому вказуються параметри для монтування. Конфігураційний файл /etc/fstab використовується в основному для того, щоб забезпечити автоматичне монтування файлових систем у процесі завантаження. Кожний рядок цього файлу містить опис однієї файлової системи й складається з 6 полів, що розділюються пробілами:

1. ім'я пристрою. Як ім'я може використовуватися як ім'я локального пристрою, наприклад, /dev/hda5, так і шляхове ім'я мережної файлової системи NFS, наприклад, pc21:/home/jim, що вказує на каталог /home/jim на машині з ім'ям pc21;
2. точка монтування (повне ім'я каталогу, у який буде монтуватися файлова система);
3. тип файлової системи;
4. опції монтування (за замовчуванням мається на увазі rw — читання, запис);
5. рівень дампу. Це поле використовується програмою dump, призначеної для створення резервних копій. Якщо файлова система повинна брати участь у процесі резервного копіювання, то тут повинне стояти число 1, якщо ні — 0;
6. порядок (пріоритет) перевірки файлових систем програмою fsck. Системи з меншими значеннями цього поля перевіряються раніше. Системи з однаковими номерами перевіряються, якщо це можливо, паралельно.

Замість типу файлової системи в поле "тип файлової системи" (і в опції -t команди
mount) можна задати значення auto. У такому випадку команда mount спробує самостійно визначити тип монтуємої файлової системи. Однак це в багатьох випадках приводить до помилок, тому краще вказати тип явно. Можна перелічити кілька типів (через кому). У команді mount можна також спочатку задати список типів файлових систем, які не треба монтувати. Цей список задається за допомогою прапора no. Така можливість може виявитися корисної в тому випадку, коли використовується команда mount з аргументом - a. По цій

команді виробляється монтування всіх файлових систем, перерахованих у файлі /etc/fstab. За допомогою додаткового аргументу -t type у цьому випадку можна обмежитися монтуванням файлових систем тільки певного типу, а за допомогою прапора no можна вказати типи, які не треба монтувати. Наприклад, команда:

mount -a -t nomsdos,ext - монтує всі файлові системи, за винятком тих, які відносяться до типів msdos і ext. Коли монтується файлова система, згадана у файлі /etc/fstab, то в команді монтування досить вказати тільки один аргумент — або ім'я пристрою, або крапку монтування. Всі інші параметри команда mount візьме з файлу /etc/fstab. Звичайно монтувати файлові системи може тільки суперкористувач, але якщо в поле опцій монтування файлу /etc/fstab указати опцію user, то відповідну файлову систему зможуть змонтувати всі користувачі. Так, якщо в /etc/fstab є рядок

/dev/cdrom /cd iso9660 ro,user,noauto,unhide то будь-який користувач зможе змонтувати файлову систему на своєму CDROM, використовуючи команду:

[user]$ mount /dev/cdrom або **[user]$ mount /cd**

Команди mount і umount підтримують в актуальному стані таблицю (або перелік) змонтованих файлових систем. Цей перелік зберігається на диску у вигляді файлу /etc/mtab. Цей файл можна переглянути безпосередньо, або вивести на екран командою mount без аргументів. Якщо ви хочете монтувати якусь систему тільки для читання з її, то у відповідному рядку файлу /etc/fstab треба або вказати опцію r (read only, за замовчуванням мається на увазі rw, тобто й читання, і запис), або використовувати команду mount з параметром -r.