Информатика, вычислительная техника и автоматизация.

Ротань О.С.,
студент,
факультет інформатики, математики та економіки,
Мелітопольський державний педагогічний університет
імені Богдана Хмельницького
Науковий керівник:
Сердюк І.М.,
асистент,
кафедра інформатики і кібернетики,
Мелітопольський державний педагогічний університет
імені Богдана Хмельницького

ОРГАНІЗАЦІЯ КЕРУВАННЯ ПРАВАМИ ДОСТУПУ ДО ФАЙЛІВ У UNIX-СИСТЕМАХ

Анотація: у статті розглянуто деякі аспекти централізованого управління налаштуванням прав доступу до файлів у Unix-системах, основні атрибути, які керують доступом до файлів та наведено приклади режимів запису прав доступу до файлів.

Ключові слова: адміністрування, права доступу, файлова система, Unix-система, користувач.

Keywords: administration, permissions, file system, Unix system, user.

Як відомо, UNIX-подібні операційні системи є багатокористувацькими, тому проблема організації розмежування доступу до файлів і каталогів – одне з головних питань, які повинна вирішувати операційна система. А оскільки Unix із самого початку розроблялася як багатокористувацька, то такий механізм, як права доступу до файлів і каталогів, що дозволяє розмежувати повноваження користувачів, які працюють в системі, тут передбачено. Додавання нових і супровід існуючих облікових записів користувачів (user accounts) є звичайною задачею системного адміністрування. До того ж зовнішні пристрої у UNIX-подібних операційних системах також є об'єктами файлової системи, цей механізм можна застосовувати і для управління доступом до пристроїв [3].

У зв’язку з поширенням на персональних комп’ютерах та на мобільних пристроях ОС Linux, від майбутніх інженерів-програмістів потрібна наявність поглиблених знань з адміністрування UNIX-подібних операційних систем.

Методи захисту файлів від несанкціонованого доступу описано С.Д. Кузнецовим, який стверджував, що в UNIX підтримується однаковий механізм контролю доступу до файлів і довідників файлової системи. Будь-який користувач або процес може отримати доступ до деякого файлу в тому і тільки в тому випадку, якщо права доступу, описані при файлі, відповідають можливостям даного користувача або процесу [1]. Механізм контроля доступу до файлів розглядали Е. Таненбаум [6, 872.] та А.М. Робачевский [5, 36]. Прийомам системного адміністрування присвячено роботи Рассела Г. Сейджа (Russel G. Sage), Митчела Уэйта (Mitchell Waite), Дональда Мартина (Donald Martin) и Стефена Прата (Stephen Prata) [4].

У статті було поставлено за мету розглянути централізоване управління налаштуванням прав доступу до файлів у Unix-системах.

Визначити права доступу до файлу – визначити для кожного користувача набір операцій, які він може застосувати до даного файлу. У різних файлових системах може бути визначений свій список диференційовних операцій доступу. У найзагальнішому випадку права доступу можуть бути описані матрицею прав доступу, де стовпчики відповідають усім файлам системи, рядки – всім користувачам, а на перетині рядків і стовпців вказуються дозволені операції [2, 272]

В основі механізмів розмежування доступу лежать імена користувачів та імена груп користувачів. В Unix кожен користувач має унікальне ім'я (ідентифікатор користувача UID), під яким він входить в систему (авторизується). Крім того, в системі створюється певна кількість груп користувачів, які також мають свій ідентифікатор (GID), причому кожен користувач може бути включений в одну або кілька груп. Створювати і

видаляти групи може привілейований користувач, він також може змінювати і склад учасників тієї чи іншої групи. Члени різних груп можуть мати різні права щодо доступу до файлів.

Основний механізм безпеки в операційних Unix-подібних системах простий. Кожен процес несе на собі UID і GID свого власника. Коли створюється файл, він отримує UID і GID того процесу, що створив його. Файл також отримує набір дозволів доступу, що визначаються створюючим процесом. Ці дозволи визначають доступ до цього файлу як для власника файлу, так для інших членів групи власника файлу і для всіх інших користувачів. Для кожної з цих трьох категорій визначається три види доступу: читання, запис і виконання файлу, що позначається буквами r, w і x (read, write, execute) відповідно [6, 872].

Право на читання (r) файлу означає, що користувач може переглядати вміст файлу за допомогою різних команд перегляду, наприклад, командою more або за допомогою будь-якого текстового редактора. Але, зробивши зміні змісту файлу в текстовому редакторі, він не зможе зберегти зміни у файлі на диску, якщо не має права на запис (w) у цей файл. Право на виконання (x) означає, що можна завантажити файл в пам’ять й спробувати запустити його на виконання як програму, що виконується. Звичайно, якщо в дійсності файл не є програмою (або скріптом shell), то запустити цей файл на виконання не вдасться, але, з іншого боку, навіть якщо файл дійсно є програмою, але право на виконання для нього не встановлене, то він теж не запуститься.

Кожен файл та директорія містить у собі особливі метадані: певний ідентифікатор власника файлу, ідентифікатор групи файлу, дев'ять бітів прав доступу власника, групи та інших користувачів, інші адміністративні біти.

Оскільки існують три категорії користувачів і три біти для кожної категорії, всі режими доступу до файлу можна закодувати 9 бітами [6, 873]. Розглянемо деякі з них (табл. 1).

Таблиця 1

Приклади режимів запису прав доступу до файлів

Двійковий	Вісімковий	Символьний	Дозволений доступ
111000000	700	rwx------	Власник може читати, писати і виконувати
111111000	770	rwxrwx---	Власник і група можуть читати, писати і виконувати
110100000	640	rw-r-----	Власник може читати і писати, група може читати
110100100	644	rw-r – r--	Власник може читати і писати, всі інші можуть читати
111101101	755	rwxr-xr-x	Власник має всі права, всі інші можуть читати і виконувати
000000000	000	---------	Ні у кого немає доступу
000000111	007	------rwx	Доступ є тільки у сторонніх
111111111	777	rwxrwxrwx	Необмежений доступ

Перші два приклади зрозумілі. У них надається повний доступ до файлу для власника файлу і його групи відповідно. У третьому прикладі групі власника дозволяється читати файл, але не дозволяється його змінювати, а всім стороннім забороняється будь-який доступ. Варіант з четвертого прикладу часто застосовується в тих випадках, коли власник файлу бажає зробити його публічним. П'ятий приклад показує режим захисту файлу, що представляє собою загальнодоступну програму. У шостому прикладі доступ заборонений всім. Такий режим іноді використовується для файлів-пустушок, які застосовуються для реалізації взаємних виключень, так як будь-яка спроба створення такого файлу призведе до помилки, якщо такий файл вже існує. Тобто якщо кілька програм одночасно спробують створити такий файл у якості блокування, тільки першій з них це вдасться. Режим, показаний в сьомому прикладі, надає всім стороннім користувачам більше доступу, ніж власнику файлу. Проте такий режим допустимо [6, 873]. Останній приклад ілюструє необмежені права доступу до файлів. Будь-хто

може опрацьовувати файли будь-яким чином. Але у власника файлу завжди є спосіб змінити в подальшому режим доступу до файлу, навіть якщо йому буде заборонений будь-який доступ до самого файлу.

Для зміни прав доступу до файлу використовується команда chmod. Її можна використовувати у двох варіантах. У першому варіанті потрібно явно вказати, кому яке право дається або кого цього права позбавлено [5, 38]:

[user]$ chmod wXp ім'я-файлу

де замість символу w ставиться:

- символ u (тобто користувач, що є власником);
- g (група);
- o (всі користувачі, що не входять у групу, якій належить даний файл);
- a (всі користувачі системи, тобто й власник, і група, і всі інші).

Замість X ставиться:

- + (надаємо право);
- – (лишаєм відповідного права);
- = (установити зазначені права замість наявних).

Замість p – символ, що позначає відповідне право:

- r (читання);
- w (запис);
- x (виконання).

Кілька прикладів використання команди chmod:

- [user]$ chmod a+x file_name

  надає всім користувачам системи право на виконання даного файлу.
- [user]$ chmod go-rw file_name

  видаляє право на читання й запис для всіх, крім власника файлу.
- [user]$ chmod ugo+rwx file_name

  дає всім права на читання, запис і виконання.

Нами розглянуто тільки основні атрибути, які керують доступом до файлу. Існує ще декілька атрибутів, що змінюють стандартне виконання

різних операцій. Як і у випадку прав доступу, ці атрибути по-різному інтерпретуются для каталогів та інших типів файлів.

Отже, правильне та поетапне налаштування прав доступу дозволяє підвищити надійність системи, при цьому захистивши від зміни або видалення важливі системні файли.

Література:

1. Кузнецов С.Д. Операционная система UNIX [Електронний ресурс] // IT-портал CITForum.ru. – Режим доступу: http://citforum.ru/operating_systems/unix/glava_15.shtml. (Дата звернення: 20.12.2017).

2. Олифер В.Г., Олифер Н.А. Сетевые операционные системы: Учебник для вузов. 2е издание / В.Г. Олифер, Н.А. Олифер. – СПб.: Питер, 2009. – 669 с.

3. Права доступу [Електронний ресурс] // WIKITNEU: Портал спільноти – Режим доступу: http://wiki.tneu.edu.ua/index.php?title=Права доступу (Дата звернення: 20.12.2017).

4. Приемы профессиональной работы в UNIX Перевод "Tricks of the UNIX Masters" by Russel G. Sage, Библиотека М. Мошкова [Електронний ресурс] // IT-портал CITForum.ru. – Режим доступу: http://citforum.ru/operating_systems/unixprofi/index.shtml. (Дата звернення: 20.12.2017).

5. Робачевский А.М. Операционная система UNIX. 2-еизд., перераб. и доп. / А.М. Робачевский, С.А. Немнюгин, О.Л. Стесик. – СПб.: БХВ-Петербург, 2010. – 656 с : ил.

6. Таненбаум Э. Современные операционные системы. 3-е изд. / Э. Таненбаум, Х. Бос. – СПб.: Питер, 2015. – 1120 с.: ил. – (Серия «Классика computer science»).