情報科学演習 資料 5
ファイルのアクセス権とシンボリックリンク

令和 6 年 5 月 8 日

目 次

1　ファイルの詳細情報とアクセス権

1.1　ユーザーとグループ

UNIX システムの利用者は，そのシステム内で重複しないユーザー名 (ログイン名) でシステムに登録されている必要があります。利用者は正しいユーザー名とパスワードを入力することで，システムの利用が許可されます (ログイン)。

さらに UNIX には**グループ** *(group)* という概念があります。通常，UNIX のシステムには複数のグループが登録されており，各ユーザーは少なくともそのうちの一つに属しています。自分が所属するグループは

```
groups
```

というコマンドで調べることができます。試してみましょう。

1.2　ファイルの詳細情報

UNIX のファイルには，ファイル名の他にも種々の情報が付与されています。ls にオプション `-l` を付けて実行することにより，ファイルの詳細情報 (long format) を表示することができます。また，ls の引数にディレクトリやファイルの名前を与えると，ls はそれらの情報のみを出力します。複数のファイル名やディレクトリ名を ls に与えることも可能です。

したがって，例えばファイル .bashrc が存在するディレクトリにおいて

```
ls -l .bashrc
```

を実行すると[1]，ファイル .bashrc のみの詳細情報を得ることができます[2]。

```
種類          リンク数  グループ          更新日時
↓             ↓         ↓             <---------->
-rw-r--r--   1 user1   group1   500 Nov 25 16:55 .bashrc
 <------->     ↑          ↑                      ↑
 アクセス権    所有者     サイズ                 ファイル名
```

種類　ファイルの詳細情報における一番左の記号はファイルの種類であり，`-` は通常のファイルを，`d` はディレクトリであることを表します[3]。

所有者　そのファイルを所有するユーザーの名前です。デフォルトでは[4]，ファイルを作成したユーザーが，その所有者となります[5]。

---

[1] ls の引数にファイル名を指定した場合には，. (ドット) で始まる名前のファイル情報を出力する場合であってもオプション`-a` を付ける必要はありません。

[2] 引数にファイル名やディレクトリ名を与えなければ，カレントディレクトリ内のファイルに関する詳細情報一覧が得られます。

[3] ファイルの種類はこれだけではないので，この欄が `-` や `d` 以外になることもあります。ここに表示される可能性のある記号は ls のマニュアルに掲載されています。

[4] 「特に何もしなければ (by default)」

[5] ファイル .bashrc は各ユーザーのホームディレクトリに予め用意されていたファイルであり，所有者も予め各ユーザーに設定されています。

グループ　この項目はファイルの所属グループを表します[6] 。ファイルに所有者があるのと同様に，ファイルは第 1.1 節で説明したグループにも属します。デフォルトでは，ファイルの所有者が所属する第一番目のグループが，そのファイルの所属グループとなります。

サイズ　byte 単位でのファイルサイズです。テキストファイルでは，ファイル中の英数字や記号 1 文字のサイズは 1 byte です。

更新日時・ファイル名　ファイルが最後に変更された日時とファイル名です。

1.3　ファイルのアクセス権

UNIX にはファイルに対する読み書き等の操作の権限を，ファイルの所有者や，ファイルが属するグループ内のユーザーなどに対して設定するための仕組みがあります。それらの権限をファイルの**アクセス権** *(permission)* といいます。アクセス権をパーミッションやアクセス許可，使用許可等と呼ぶこともあります。

1.3.1　アクセス権の読み方

ファイルのアクセス権は `ls -l` で表示されるファイルの詳細情報から調べることができます[7]。そのために必要なのはアクセス権と所有者およびグループです。本節では，第 1.2 節で例示したファイル .bashrc の詳細情報

> アクセス権: `rw-r--r--`，所有者: user1，グループ: group1

を例に，アクセス権の読み方を説明します。

1. アクセス権を構成する 9 文字を 3 文字ずつの組に分けます。.bashrc の場合は `rw-` と `r--` と `r--` の 3 組です。

   各組は左から順に

   (a) ファイルの所有者 (user1) に対するアクセス権 (`rw-`)

   (b) ファイルのグループ (group1) に属する，所有者 (user1) 以外のユーザーに対するアクセス権 (`r--`)

   (c) 上記以外のユーザーに対するアクセス権 (`r--`)

   を表しています。

2. 3 つの組それぞれについて，各組を構成する 3 つの文字からファイルのアクセス権を調べます。3 つの文字は左から順に，次に示す 3 種類のアクセス権を表しています。

   (a) ファイルの内容の読み取り権[8](r または -)

   `r`：読み取り可能 (readable)

   `-`：読み取り不可能

[6] グループを表示するために，ls にオプション `-g` が必要な UNIX システムもあります

[7] ここで紹介するのは UNIX 系 OS に共通の基本的なアクセス権のみです。最近の UNIX 系 OS ではアクセス権をさらに細かく設定できるように独自の拡張が行われています。

[8] ディレクトリの場合，r はディレクトリ内のファイル一覧を得る権限

(b) ファイルへの書き込み権[9](w または -)

`w`：書き込み可能 (writable)

`-`：書き込み不可能

(c) ファイルの実行権[10] (x または -)

`x`：実行可能 (executable)

`-`：実行不可能

以上より，.bashrc のアクセス権は

1. 所有者 user1 に対して，読み取り可・書き込み可・実行不可 (`rw-`)
2. グループ group1 に属する，user1 以外のユーザーに対して，読き取り可・書き込み不可・実行不可 (`r--`)
3. その他のユーザーに対して，読み取り可・書き込み不可・実行不可 (`r--`)

であることがわかります。すなわち，.bashrc の所有者 user1 は，このファイルの内容を読んだり，ファイルに書き込みを行うことができます。グループ group1 に属する user1 以外のユーザーと，その他のユーザーは，このファイルの内容を読み取ることのみが許されています。

1.3.2 アクセス権の変更 — chmod コマンド

ファイルの所有者は，chmod (change file mode) コマンドを使って，ファイルのアクセス権を変更することができます。chmod の基本的な使い方は

`chmod` *mode file …*

であり，*mode* には *file …* の現在のアクセス権をどのように変更するのかを指定します。*mode* に指定する要素は「誰に」「どの権限を」「与える (または取り除く)」の三つですが，記述の順番は次のとおりです。

1. 「誰に」：u, g, o, a の何れかを指定します[11]。
   - u：ファイルの所有者 (user[12])
   - g：所有者を除くファイルのグループに属するユーザー (group)
   - o：その他のユーザー (other)
   - a：u, g, o の全て (all)
2. 「与える」または「取り除く」：+ か - を指定します。
   - +：与える
   - -：取り除く
3. 「どの権限を」; r, w, x の何れかを指定します。

[9] ディレクトリの場合，w はディレクトリ内にファイルを作ったり，ファイルを削除する権限
[10] ディレクトリの場合，x はディレクトリ内のファイルにアクセスする権限 (検索権)
[11] u, g, o を組み合わせるなど，より複雑な記述も可能ですが，何れか一つを指定する方法を知っていれば困らないでしょう。詳細を知りたければ man コマンドで調べてください。
[12] owner と呼ぶべきですが，other と区別するために user と記します

- r : 読み取り権
- w : 書き込み権
- x : 実行権

例えば

`chmod g+w` *file*

だと，*file* のグループに所属する (所有者以外の) ユーザーに，書き込みの権限を与えることになります。また，

`chmod a-w` *file*

では，全てのユーザーに対して，file への書き込みを禁止します。

1.3.3 アクセス権の 8 進数指定（絶対指定）

chmod における *mode* を `g+w` などの記号で指定する代りに 8 進数で指定することも可能です。8 進数による指定では，まず次の表にあるように，読み取り・書き込み・実行の可または不可を表す 3 文字の記号を，– のときは 0 に，それ以外のときは 1 に対応させます。そうしてできあがった 0 と 1 の並びを 3 桁の 2 進数として解釈し，8 進数で表現します。

記号	0,1 の並び (2 進数表記)	8 進数	意味
`---`	000	0	読取不可・書込不可・実行不可
`--x`	001	1	読取不可・書込不可・実行可
`-w-`	010	2	読取不可・書込可・実行不可
`-wx`	011	3	読取不可・書込可・実行可
`r--`	100	4	読取可・書込不可・実行不可
`r-x`	101	5	読取可・書込不可・実行可
`rw-`	110	6	読取可・書込可・実行不可
`rwx`	111	7	読取可・書込可・実行可

上記の一桁の 8 進数を，所有者・グループ・その他のユーザーに対するアクセス権として順に並べると，アクセス権の 8 進数表記ができあがります。この方法による chmod の例を次に示します.

- `chmod 644` *file* (すべてのユーザーが読み取り可能，所有者のみ書き込み可能)
- `chmod 755` *file* (すべてのユーザーが読み取りと実行可能，所有者のみ書き込み可能)

この方法では，現在のアクセス権とは無関係に，指定したアクセス権が設定されることになります (絶対指定)。

1.4 特権ユーザーとセキュリティ

UNIX には「特権ユーザー」,「スーパーユーザー」等と呼ばれるシステム管理のための特別なユーザーが存在します。UNIX の場合，通常，特権ユーザーのユーザー名は root です。ユーザー

root にはシステムに対するすべての操作が許されていて，root はシステム内のどんなファイルでも見ることができるし，変更したり削除することもできます[13]。

システムが動作するのに必要な諸設定は，一般のユーザーが書き込めないファイルに保存されているので，管理者がシステムの設定変更を行うためには root として作業する必要があります。一方で，root が操作を誤ればシステムが壊れることもあり得ます[14]。

さて，もし root のパスワードが悪意ある人に知られてしまったら，どういう事が起こり得るか，想像できますね。

root を含むユーザーのログイン用パスワードは，暗号化されてパスワードファイルに格納されています。したがって，パスワードファイルの中を見ることができたとしても，暗号を解読できなければパスワードはわかりませんが，一般利用者のパスワードが漏れて悪意ある人がシステムに侵入すると，root のパスワードを解読される危険性は高まります。すなわち皆さんのパスワードが漏ることによって被害を被るのは，皆さん自身だけとは限らないということです。

パスワードの取り扱いには注意しましょう。また，安易なパスワードを設定しないでください。

2 問題

1. ホームディレクトリに存在するファイルの詳細情報を一覧表示してください。通常のファイルとディレクトリの違いや更新日時，所有者，アクセス権等を確認しましょう。

2. ファイル /bin/cat と /bin/ls のアクセス権が，コマンドの実体としてのファイルが持つべき適切なアクセス権になっていることを確認してください。

3. まず，は echo コマンドとリダイレクトを使って

   ```
   echo > now
   ```

   を実行し，空行（改行文字）が一つだけ入ったファイル now を作ってください。

   続いて，ファイル now だけの詳細情報を表示し，アクセス権やファイルサイズ，更新時刻などを確認してください。

4. date コマンドが出力する現在の日付や時刻をファイル now に書き込んだ後，再度 now のファイルサイズや更新時刻などを確認してください。ファイル now の内容が正しく書き換わったことも確認してください。

5. chmod コマンドを使って now のアクセス権を `-w-r--r--` に変更してください。(ファイル所有者の読み取り権を外します)

6. now の中を cat コマンドなどで見てみましょう。

   見えないはずですね。

7. now のアクセス権を `r--r--r--` に変更してから中身を見てみましょう。（ファイル所有者に読み取り権を与えてから書き込み権を外すなど）

8. 再度，now に現在の日付や時刻を書き込みましょう。

   できないはずですね。書き込み権を外すと，誤操作からファイルを保護できます。

[13] 従って，他人に知られては本当に困る情報を，共用の計算機システムに保管するのは避けるべきです。
[14] そのため，システム管理者が管理目的以外で root としてシステムを利用するのは極力避けるべきです。

9. now の削除を試みましょう。

   書き込み許可のないファイルは，すぐには削除されず，

   > rm: 書き込み保護されたファイル 通常ファイル 'now' を削除しますか?

   のような表示がでます。

   ここでは `n` で答えてください。ファイルは削除されません。

10. now のアクセス権を `rw-r--r--` に戻してから，now を削除しましょう。

11. 自分の生まれた月のカレンダーを表示するコマンドを作って実行しましょう。

    複雑な処理をするコマンドを作成するにはプログラミングの知識が必要ですが，簡単な処理をするコマンドであれば，既存のコマンドを組み合わせてテキストファイルに記入し，そのファイルに実行の許可を与えるだけで作成することができます[15]。

    以下の手順に従って作業をしてください。

    (a) 次のコマンドを順に実行して，結果を確認してください。

    ```
    echo name was born in
    cal month year
    ```

    ここで，*name* には自分の名前を，*month year* には生まれた月と西暦を与えてください。

    (b) 前項で実行したコマンド行の内容 (echo ... と cal ...) 2 行が入ったテキストファイルを作成してください。ファイル名は mybirth としましょう。ファイルの作成には vi などのテキストエディタを用いるのがいいでしょう。

    (c) そのファイルに，すべてのユーザーが実行できる実行権を与えてください。

    (d) ファイルに実行権を与えましたので，ファイルをコマンドとして実行できるようになりました。作成したコマンドを実行してください。

3 シンボリックリンク

UNIX ファイルシステムでは，既存のファイルに別の名前を付けて利用することができます。この仕組みがリンク (link) です[16]。UNIX ファイルシステムには，ハードリンク (hard link) とシンボリックリンク (symbolic link) がありますが，ここではシンボリックリンクのみを取り上げます[17]。

シンボリックリンクを作成するには，コマンド `ln` にオプション `-s` を付けて実行します。基本的な実行の書式は

> `ln -s` リンク先のファイル名 作成するリンク名

です。

---

[15] このテキストファイルを**シェルスクリプト**といいます。ここでは扱いませんが，シェルスクリプトでは制御構造や変数など，より高度なプログラミングの機能を使うことも可能です。

[16] ここでいうリンクはファイルシステムにおけるリンクであって，Web のハイパーリンク (hyperlink) とは異なるものです。

[17] Windows のショーカットは UNIX のシンボリックリンクに似ています。ただし，シンボリックリンクは ショートカットよりシンプルで動作が明快です。

3.1 通常ファイルへのシンボリックリンク

第 2 節の問題 11 で作成した誕生月カレンダーを表示するコマンドが，ホームディレクトリに mybirth という名前のファイルとして存在するとします。以下では，このファイルへのシンボリックリンクを作成して，その性質を見てみます。

1. まず，mybirth がどのディレクトリにあるかを確かめましょう。ホームディレクトリ以外の場所にあるならば，mv コマンドを使ってホームディレクトリに移動してください。

   ファイルの中身と実行結果も確認してください。

2. mybirth へのシンボリックリンク symlink をホームディレクトリに作成します。

   ```
   cd
   ln -s mybirth symlink
   ```

3. できたかどうか確認します。シンボリックリンクもファイルなので，ls コマンドが使えます。

   ```
   ls
   ls -F
   ```

   シンボリックリンクであることを表す印はわかりましたか。

4. もっと詳しい情報も見てみましょう。

   ```
   ls -l
   ```

   シンボリックリンクでは，ファイルの種類の表示（左端）が l になっていますね。また，ファイル名の表示欄からはリンク先もわかります。シンボリックリンクのアクセス権は，リンク先のものが引き継がれます。

5. シンボリックリンクの内容はどのように表示されるでしょう。次のように，シンボリックリンクとリンク先ファイルに cat コマンドを適用して比較しましょう。

   ```
   cat symlink
   cat mybirth
   ```

6. 実行結果を比較しましょう。

   ```
   ./symlink
   ./mybirth
   ```

7. エディタで symlink を開き，ファイルの末尾に次の 1 行を加えてから保存してください。

   ```
   echo symlink changed
   ```

   続いて，cat コマンドで symlink とリンク先のファイル mybirth の内容を比較してください。また，symlink と mybirth をコマンドとして実行してください。

8. rm コマンドを使って symlink を削除してください。リンク先だった mybirth がどうなったかを確認してください。

3.2 ディレクトリへのシンボリックリンク

1. ディレクトリ /bin へのシンボリックリンク mybin を，ホームディレクトリに作成します[18]。

```
cd
ln -s /bin mybin
```

2. できたかどうか確認します。

```
ls -l
```

3. シンボリックリンクを使ってカレントディレクトリを /bin にします[19]。

```
cd mybin
pwd
ls
```

4. こんなこともできます。何をしているのか，ちゃんと考えながら実行してください。

```
cd
pwd
ls
ls mybin
ls -l mybin/pwd
mybin/pwd
mybin/echo now using symbolic link for /bin
```

5. シンボリックリンクを削除しても，リンク先のディレクトリに影響はありません。

```
rm mybin
ls /bin
```

[18] ここではわかりやすさを優先し，リンク先を絶対パス名で指定していますが，シンボリックリンクにおけるリンク先には相対パス名を使うことが推奨されます。

[19] pwd の出力を，リンク先ディレクトリの絶対パス名にするには pwd -P を実行します。なお，/bin の実体が他のディレクトリへのシンボリックリンクになっているシステムもあり，その場合 pwd -P を実行しても出力は /bin とはなりません。

4 問題

1. ホームディレクトリに存在するディレクトリ unix に，現在の日付・日時が入ったファイル now を作ってください。date コマンドを使います。

   できたかどうか，確認してください。

2. ホームディレクトリをカレントディレクトリとしてから，前項で作成したファイル now へのシンボリックリンクを，ホームディレクトリに作成してください。シンボリックリンクの名前は symnow とします。リンク先は，相対パス名を使えば unix/now ですね。

3. ファイル now の内容を相対パス名を使って出力してください。シンボリックリンク symnow を使って同じ出力を得てください。

   > シンボリックリンクを作ることで，カレントディレクトリとは異なるディレクトリのファイルを，パス名に代えファイル名だけで扱うことが可能になります。

4. リンク先のファイル now の名前を old に変更してから，ls コマンドでシンボリックリンク symnow の詳細情報を出力してください。さらに，ホームディレクトリで `cat symnow` を実行してみましょう。

   > しかられますね。リンク先の名前が変わるとシンボリックリンクは無効になります。

5. 再度，ホームディレクトリに存在するディレクトリ unix に，現在の日付・日時が入ったファイル now を作ってください。ls コマンドでシンボリックリンク symnow の詳細情報を出力し，ホームディレクトリで `cat symnow` を実行してみましょう。

   > リンク先が存在すれば，シンボリックリンクは有効です。

   結果を確認したら，symnow と now と old を削除しましょう。

6. ホームディレクトリに，ディレクトリ /usr/bin へのシンボリックリンク mybin を作成してください。ただし，絶対パス名 /usr/bin でなく，相対パス名を使ってください。作成した mybin の詳細情報を表示して，正しくできたことを確認してください。

7. /usr/bin に存在するファイル echo の詳細情報を出力してください。その際，ls コマンドの引数には，mybin を使ったファイル echo のパス名を与えてください。

8. mybin を削除してください。