第 11 回の課題について

（1）A さんが B さんの公開鍵で暗号化した電子メールを、B さんと C さんに送信した結果のうち、適切なものはどれか。ここで、A さん、B さん、C さんのそれぞれの公開鍵は 3 人全員がもち、それぞれの秘密鍵は本人だけがもっているものとする。

ア　暗号化された電子メールを、B さんだけが、A さんの公開鍵で復号できる
イ　暗号化された電子メールを、B さんだけが、自身の秘密鍵で復号できる
ウ　暗号化された電子メールを、B さんも、C さんも、B さんの公開鍵で復号できる
エ　暗号化された電子メールを、B さんも、C さんも、自身の秘密鍵で復号できる

公開鍵暗号方式では暗号化するための公開鍵と復号するための秘密鍵のペアで暗号化と復号を行います。B さんの公開鍵を使って暗号化したので、それを復号できる唯一の鍵は B さんの秘密鍵です。

ア… A さんの公開鍵では復号できない、ウ… B さんの公開鍵では復号できない、
エ… B さんは自身の秘密鍵で復号は可能だが、 C さんは自身の秘密鍵では復号できない

したがって「イ」が適切であると考えられます。

『イ』ように記号だけを記載した解答もありましたが、適切でない選択肢に対して、その理由をつけていただくとよいのではと思います。

（2）次の pp.11-12 に示すアクセス制御方式をもつファイルシステムにおいて、ファイルに対して以下のアクセス権の条件すべてを満足する設定（アクセス権の設定状況）を示せ

- 全ての利用者が実行できる
- 所有者、および所有グループの利用者だけが読み出しできる
- 所有者だけが書き込みできる

```
pp.11-12 のように アクセス権は、○○○ ○○○ ○○○
                                 |      |      └ 他のユーザに対するアクセス権
                                 |      └ 所有グループに属するユーザに対するアクセス権
                                 └ 所有者自身に対するアクセス権
```

の形で、それぞれ左から 読出し：r、書込み：w、実行：x で許可を示し、 - で禁止を示します。

所有者は 読出し、書込み、実行 できるので rwx、所有グループの利用者は 読み出し、実行できるので r-x、 その他の利用者は実行だけできるので --x
となり、 これらをつなげて rwxr-x--x となります。

なお、先頭に -（ファイル種別を示す記号）を入れていただいた解答もありましたが、アクセス権としては、rwx のような 3 つの記号を 3 セット分並べていただければと思います。

chmod *xyz* <ファイル名>

のような解答をされた方もいらっしゃいました。chmod はアクセス権を設定するコマンドです。この問題は「アクセス権の設定状況」のつもりで、rwxr-x--x を期待するものでしたが、「満足する設定」という表現から、設定を行うコマンドを解答されたものかと思います。問題文の表現があいまいで失礼しました。

xyz は 3 桁の八進数で、*x* が所有者自身に対するアクセス権、*y* が所有グループに属するユーザに対するアクセス権、*z* が他のユーザに対するアクセス権、を表します。1 組の ○○○ を 3 ビットの二進数と考えると、r が指定されていると先頭ビットに 1 が立っていて数としての重みが 4、 w が指定されていると 2 番目のビットに 1 が立っていて数としての重みが 2、 x が指定されていると 2 番目のビットに 1 が立っていて数としての重みが 1、 - が指定されているとそのビットが 0、となります。それを八進表現して、 rwx では 7、 -r- では 2 などです。

アクセス権を設定せよ、ということですと chmod 751 <ファイル名> となります。

UNIX の（講義資料で示した）方式は、簡略化されたアクセス制御リストと考えられます。主体として、所有者/グループ/その他 という 3 つの単純な利用者群を定義し、そのそれぞれに read / write / execute を許可する/しない として行う枠組みです。ファイルの持つアクセス制御情報としても少なく、アクセス権限のチェックのオーバーヘッドも小さいといえます。あるファイルを公開する場合は、自分と同じグループに所属する利用者を想定して、その中でアクセス制御するという考え方です。
この方式の利点は単純であることですが、反面、細かい制御が難しいという欠点があります。たとえば、グループに所属しないその他の利用者 A だけにファイルの read を許したい場合には、A をグループに追加するなどグループ管理が煩雑になるなどの問題があります。

なお、UNIX での ACL は、講義資料で説明している部分を基本 ACL エントリとし、指定ユーザや指定グループなどに対するアクセス権を 拡張 ACL エントリとして保持する形式です。

以下は、 授業で補足した参考スライドです。（ノート資料には入っていませんのでここに置きます）

- **参考：LinuxでのACL**
  - setfacl コマンドでファイルやディレクトリにACLを設定/削除する
  - 特定のユーザやグループに対してファイルごとのアクセス制限が可能になる
  - getfacl コマンドで具体的な設定内容を表示

```
[study@localhost mydoc]$ ls -l test*.txt   ← 現在の設定を確認
-rw-rw-r--. 1 study study 18  8月 13 06:31 test1.txt
-rw-rw-r--. 1 study study 18  8月 13 06:31 test2.txt
[study@localhost mydoc]$ setfacl -m u:penguin:r test1.txt   ↓ユーザー「penguin」に「r」を許可
[study@localhost mydoc]$ setfacl -m o:- test*.txt   ←その他には許可を与えない
[study@localhost mydoc]$ ls -l test*.txt   ← 現在の設定を確認
-rw-rw----+ 1 study study 18  8月 13 06:31 test1.txt   ユーザーpenguinに対する設定が加
-rw-rw----. 1 study study 18  8月 13 06:31 test2.txt   わったため「+」が表示されている
[study@localhost mydoc]$ getfacl test*.txt ← getfctlでACL情報を確認
# file: test1.txt
# owner: study
# group: study
user::rw-
user:penguin:r--  ← penguinに対して読み出し許可が追加されている
group::rw-
```

ACLが設定されている場合、9桁のパーミッション表示の後に「+」が表示される

記入内容	意味
user:ユーザー:許可内容	ユーザーに対する許可
group:グループ:許可内容	グループに対する許可
other:許可内容	その他に対する許可

@IT https://atmarkit.itmedia.co.jp/ait/articles/1808/23/news026.html より