ネットワーク
システム管理 #12

たかさきこうや

1限 (09:00-10:45)

1

おさらい：最終課題の内容

- 「コマンドを複数組み合わせて作った」「シェルスクリプト」で「何か自分の生活を便利にするようなスクリプト」を作り
- なにがことが起きた時に、必要な情報を自動的に送ってくれるようなプログラムを考えて、作ってみてください
- 基本的なUNIX上で動作するプログラムとします
  - Windows上で動くものは今回は対象外とします

2

第13回の講義2日後、1月28日23:59:59に

- これを提出していただきます
  - 該当のプログラムは、自分のホームディレクトリのbinの下に置いて「たかさきが」見れるようにしておくこと
- これとは別に、報告書を書いてもらいます
  - 「問題提起」
    - 俺こんなことに困ってたんだよね
  - 「作ったもの」
    - だからこういうものを作ったんだ
  - 「動かしてみた状況と考えたこと」
    - こんな風になったので、こう助かってるけど、こういう風にしたらもっと良くなったこと

3

報告書の内容

- PowerPointで数枚
  - 1:表紙には表題と氏名
  - 2:「問題提起」
    - 俺こんなことに困ってたんだよね
  - 3:「作ったもの」「動かしてみた状況」
    - こんな風になったので、こう助かってる
  - 4:「動かした結果考えたこと」
    - こういう方がよかったかもしれない
  - 5:その他語りたいことがあれば自由に
- を、各１ページ以上で語る
  (こちらはメール添付で提出)

4

ファイルの置き場所と名前

- ~/bin/nsa-final.sh としてください
  - 他のUNIX系課題同様、当面はファイルを置いておいてください
    - 僕から見られるようにしておかないと提出物が確認できないので
      「パーミッションにはくれぐれも注意」してください

5

ファイルの比較

- たとえば、あるWebサイトを監視していて

```
% curl https://www.cuc.ac.jp/ > cuc.txt
```

- こんな感じでコンテンツを取得し、cuc.txtというファイルに出力したとする
- これの変化を検出するために、例えばこういうやりかたがある

```
% mv cuc.txt cuc-old.txt
% curl https://www.cuc.ac.jp/ > cuc.txt
% diff cuc-old.txt cuc.txt
```

6

diff コマンド

- ファイルAとファイルBの差異を比較し、差がある場合その内容を出力するコマンド

```
% mv cuc.txt cuc-old.txt
% curl http://www.cuc.ac.jp/ > cuc.txt
% diff cuc-old.txt cuc.txt
```

- この場合、以前に取得したcuc.txt というファイルが既にあると言う前提でcuc-old.txtに名前を変更したうえで再度cuc.txtを作り、cuc-old.txtとcuc.txtの差異を比較している

7

差があるかどうかはどうやってわかる？

- wcコマンドにdiffの結果を食わせてやり、0行ならば「差がない」

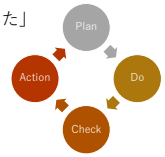
```
% mv cuc.txt cuc-old.txt
% curl http://www.cuc.ac.jp/ > cuc.txt
% diff cuc-old.txt cuc.txt | wc –l
```

- もし1以上の数字が入っているなら、「差がある」ので「更新された」とみなしてよい
- 差分を全部メールで送る必要はない。何らかの更新があったという情報だけでも有益

8

報告書を書くに際して

- 1枚目の表紙…はおいといて
- 2枚目の「こういう問題がありました」
  →「こういうものを作りました」
- 3枚目の「実際に動かしてみたらこんなでした」
- 4枚目以降の「思うところはこうです」
- これ、なんかみたことない？
  - こういう奴→



9

PDCAサイクル

- という概念そのものは、色々と弱点や欠点もあるので取り立てて「素晴らしいもの」というわけでもない
  - CUCなどの大学教育機関では「問題発見解決」というキーワードを用いることもあるよね
- ただし、「(自分にとって)問題があることに気付く」ことと「その問題をなんとかして(ITスキルで)解決すること」はこの講義のテーマでもあるので、ここが採点のキモになります

10

逆に言うと

- 技術的に稚拙か、優れているかはポイントではありません
- 泥臭くても「(自分の)問題が解決できているかどうか」がポイントです

11

今までやってきた、使えそうなコマンドの数々

無論ここに挙げた以外にも役立つコマンドは色々ある

- 出力加工関連
  - wc…文字数、行数カウント
  - nkf…文字コード変換
  - sort…行ごとに並べ替え
  - grep…特定行の抽出
  - sed…特定文字列置き換え
  - diff…ファイル比較
  - cut…文字切り出し
  - echo…文字出力
- UNIXログイン関連
  - whois…ユーザ確認
  - last…ユーザログイン履歴確認
  - ps…プロセス確認
- ファイル操作関連
  - mv…ファイル名変更
  - cp…ファイルコピー
  - rm…ファイル削除
  - mkdir…ディレクトリ作成
  - rmdir…ディレクトリ削除
  - chmod…ファイル権限変更
- ネットワーク問診関連
  - ifconfig…IPアドレス等調査
  - ping…疎通確認
  - traceroute…経路確認
  - netstat…通信確認
  - curl…Web通信/コンテンツ取り寄せ

12

シェルスクリプト

- 前回もやったけれど、今みんなは、UNIXにログインしている
- ログインした時に最初に起動されるプログラムを「シェル」と言う
- このシェルは、ログインしたユーザに様々な便利な環境を提供する(上矢印で以前のコマンドを呼び出したりとか)が、簡単なプログラミング環境としても機能する
  - 書いたプログラムをビルド（コンパイル）せずに使えるのでインタプリタなどと呼ばれる

13

一例

- 行頭に、どんな言語でスクリプトを書くかを宣言する
  - この場合はsh

```
#! /bin/sh

CNT=`ping -q -c 3 -i 2 www.google.com | grep "packet loss" | cut -f 3 -d"," | sed 's/% packet loss//' | sed 's/ //g`

echo  $CNT
```

  - ping～から始まるプログラムを実行し、結果をCNTという変数に入れる
  - 変数の内容をechoで出力する

14

for文とif文

- for文は「繰り返し文」
  - 同じ処理、あるいは似たような処理を、指定の回数だけ繰り返す
- if文は「条件分岐」
  - ある条件を満たす場合とそうでない場合で、振る舞いを変える
- いずれも、プログラミングの世界では基本中の基本だし人間も日常生活で同じことを無意識のうちにしている（はず）

15

for文の例

- 変数で事前にアクセスするサイトを決定しておき、for文で順番に処理していく例
  - do～doneの部分を繰り返し実行する数が多くなった場合はこれで見通しが良くなる

```
#! /bin/sh

STR1=https://www.cuc.ac.jp/
STR2=https://www.keio.ac.jp/
STR3=https://www.kaetsu.ac.jp/
STR4=https://www.wayo.ac.jp/
STR5=https://www.shodai.ac.jp/

for i in $STR1 $STR2 $STR3 $STR4 $STR5
do
  curl $i
  echo "-----"
done
```

16

if文の例「もし〇〇ならA　それ以外ならB」

- CNTに入っている数字が、0より大きければ「問題です」と表示。そうでなければ「問題ありません」を表示

```
#! /bin/sh

CNT=`ping -q -c 3 -i 2 www.google.com | grep "packet loss" | cut -f 3 -d"," | sed 's/% packet loss//' | sed 's/ //g`

echo $CNT

if [ $CNT -gt 0 ] ; then
 echo "問題です"
else
 echo "問題ありません"
fi
```

17

シェルスクリプトの亜種

- shの他に、bashやtcshなど、追加機能が使えるものも存在する
- もし得意であるなら、perl、python、rubyなどを使ってもいい
  - これらもスクリプト言語なのでコンパイルせずに使える
  - 僕はperlやphpを好んで使う
  - 何れの言語も、スクリプト言語の中からUNIXコマンドを呼び出して使えるので、コマンドをパイプで繋いで逐次処理させることが可能

18

自動実行と繰り返し実行

- みなさんに作って貰ったプログラムを
  - 定期的に実行する
  - 繰り返し実行する
- ための機能はUNIXに元々存在しているので、皆さんが「繰り返し機能」「自動実行機能」をスクリプトの中に仕込む必要はありません

19

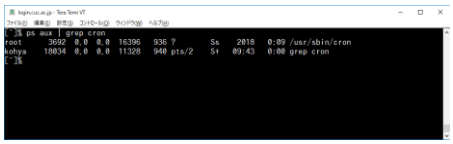
プログラムを自動的に動かす

- 何かを定期的に実行し、情報を持ってきて、それが有用ならお知らせをしてくれる…みたいなプログラムを作る場合、そのプログラムは
  - 常に動いている
    - デーモンとかサービスとかいう、という話は前にした
  - 定期的に起動してくれる
- のいずれかである必要がある

20

Cron(クロン/クーロン)

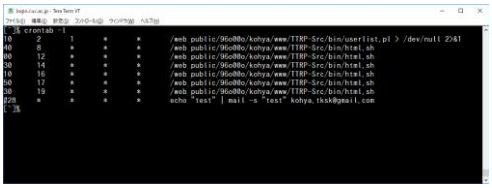
- UNIX系のサーバで、決まった時間にプログラムを起動してくれるサービス(デーモン)
- cronというプログラムが常時動いているサーバではこれが使える
  - Windowsでは「タスクスケジューラ」と呼ぶ



21

crontabコマンド

- crontab コマンドで、一般ユーザは自分の「自動実行プログラム」を登録できる



22

crontab –e

- % setenv EDITOR emacs
- % crontab –e
- で、cron登録用にemacsが立ち上がる
  - (setenv EDITOR emacs をやらないと、viが起動する)
- そこで、
- 10 * * * * echo “test” |sendmail 自分のメールアドレス
- とだけ書いて保存すると
- 「毎時10分に起動して自分にメールを送ってくれる」プログラムが起動する

23

10 * * * * の意味

- 順に「分」「時」「日」「月」「週」のパラメタ
- 2 30 1 10 * echo “ten-ichi” | sendmail メールアドレス
- とすると、「毎年10月1日2時30分」に「ten-ichi」と書かれたメールを飛ばすことができる

- 「30 4 * * 6」 とすれば、「毎週土曜日の午前4時30分」になる
- 仕掛けたcronは、crontab –l で表示可能

24

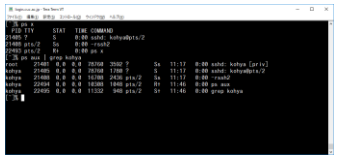
これを使うことで

- 例えば、とあるサイト(天気予報とか？)から持ってきた情報を、毎朝スマホにメールで送る、とかそういう機能が実装できる
- メール送信部分をスクリプトの中に入れて、なにか異常なときだけメールを送る、という手もある

25

cronやscreenコマンドを使うと

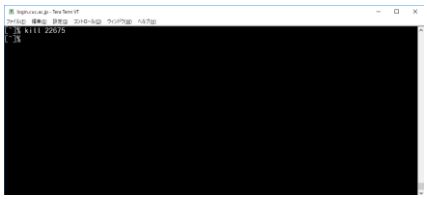
- 自分のプログラムがいつまでも走り続けてしまうことがある
- % ps aux | grep 自分のログイン名 で、自分の走らせたプログラムが暴走していないか確認
- プログラムには「プロセスID(PID)」がつくので、それを指定してプログラムを強制的に止めることができる



26

kill コマンド

- % kill PID で該当プロセスを強制的に終了可能



27

今日の残り時間は

- 次回提出するプログラムを作成するための実習時間とします
- どういうものが作りたいか
- どうすれば良いか
- を質問いただければ、レクチャーするので、遠慮なく聞いてください

28

最終回での提出に向けて

- 繰り返しになりますが、僕は非常勤なので、来週、最終回に大学に来たあとは学校に来ません
- もし提出していない過去の課題がある場合、次回までに、提出してください
  - 過去に提出した課題、**特にサーバ上のファイルは、成績評価が終わるまでは消さないでください**
- 最終課題提出締切=課題提出最終締切です
  - 最終課題のpptxファイルもここまでに(メールで)提出していただきます
  - ファイルはサーバ上に置いておいてください

29

最終課題の提出方法

- パワーポイントのファイルは**メール添付**でください
  - ファイル名： NSA3-c24XXXX.pptx
    - c24XXXXの部分はアカウント名としてください
- 両方ないと評価できないので、忘れずに出すように
- 件名: NSA03課題提出
- 本文: なんか思うところがあれば好きに書いてください

30