U.PORTO

FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

[dcc]

INTRODUÇÃO

ADMINISTRAÇÃO DE SISTEMAS

2022/2023

ROLANDO MARTINS

(ADAPTADOS DE PEDRO BRANDÃO)

Referências dos slides

- O conteúdo destes slides é baseado no livro da disciplina: “Unix and Linux System Administration Handbook (4ªEd)” por Evi Nemeth, Garth Snyder, Trent R. Hein e Ben Whaley, Prentice Hall, ISBN: 0-13-148005-7
- Alguns slides baseados em Admin. Sistemas do Prof Manuel Eduardo Correia.
- As imagens usadas têm a atribuição aos autores e/ou são de uso livre.

XKCD 705 – Devotion to Duty

Ver também o BOFH

UserFriendly – SysAdmin Day

SysAdmin

Pensar como um SysAdmin

- “I'd thank you but system administration is a thankless job”
- “System administration is like keeping the trains on time; no one notices except when they're late”.
- Envolve uma tensão constante entre a Autoridade e Responsabilidade e o espírito de serviço e cooperação que devem estar sempre subjacente no desempenho das tarefas.

```
# kill -9 `ps aux | awk '$1=="auser" {print $2}'`
$ write auser
Tens uma série de processos a correr
Precisas de ajuda?
^D
```

Deveres do SysAdmin

- Provisão de contas (cap. 7 do Livro)
- Adicionar e remover hardware (cap. 8, 13, 16 e 26)
- Fazer backups (cap. 10)
- Instalar e atualizar software (cap. 12)
- Monitorizar o Sistema (cap. 11 e 29)
- Descobrir a razão dos problemas (Troubleshooting) (cap. 21 para rede)
- Manutenção de documentação
- Monitorização de segurança (cap. 22)
- Apagar fogos

Conhecimentos Base

- Sistemas Operativos.
  - Processos, sinais, sistemas de ficheiros, device drivers, etc.
- Editar ficheiros texto (`emacs`, `vi`,`nano`, etc.).
- Shell
  - Redireccionamento de I/O, pipes, variáveis locais e de ambiente, ciclos, controlo de "jobs", etc.
- Comandos Unix de gestão corrente básicos.
  - `cd`, `pwd`, `ls`, `ps`, `grep`, `egrep`, `fgrep`, `mkdir`, `chown`, `chgrp`, `wc`, `chmod`, etc.
- Utilitários Unix
  - `cut`, `awk`, `find`, `su`, `sudo`, etc.

Aonde encontrar ajuda

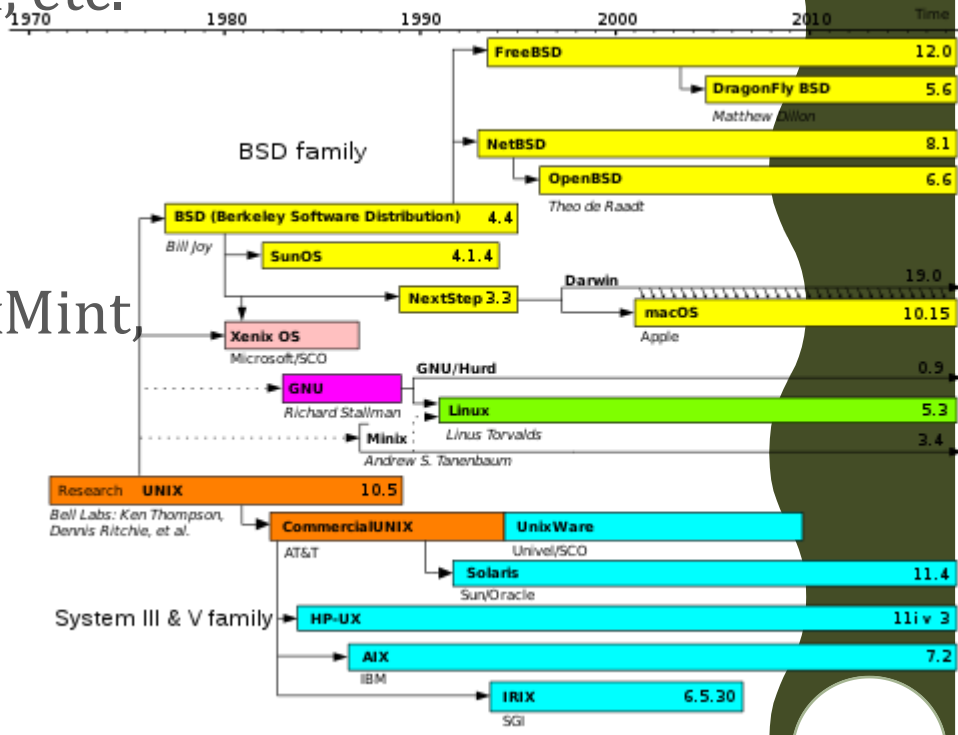
- Sim: “google is your friend”;
  - Ou outro motor de busca.
  - É razão de vergonha receber a seguinte resposta à pergunta:
    - Q: Como posso mudar a password de um utilizador em Linux?
    - R (amigo): Podes ver isso aqui.
- Páginas do manual (RTM, com um F para os menos simpáticos);
- Página de suporte da distribuição usada;
- Fóruns online (outra vez motor de busca);
- LDP (Linux Documentation Project)

Dicas

- Planear antes de atuar
- Ter backups
- Documentar todo os passos das alterações/ instalações/ correções feitas
  - Ex.: ir copiando os comandos para um ficheiro indicando eventuais erros e como recuperar
- Ter backups das configurações
- Testar
  - Se possível ter um sistema para teste e outro para operação
- Usar um sistema de controlo de versões (~~SVN, CVS,~~ GIT, etc.) para configurações
  - Também serve de backup ;-)
- …

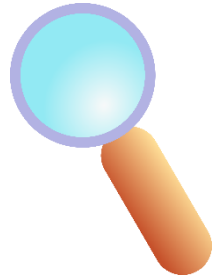
UNIX e Linux

- Linux deriva de UNIX, mas tem algumas diferenças:
  - Localização de ficheiros, secções do manual, diferenças nos comandos, sistema de gestão de software, kernel, etc.
  - Por vezes mesmo dentro das distribuições Linux
- Exemplo distribuições Linux:
  - Red Hat, Fedora, Ubuntu, CentoOS, Debian, LinuxMint, Pop!_OS, SUSE, etc.
- Exemplos distribuições UNIX:
  - BSD, HP-UX, Solaris (OpenSolaris), AIX



Know thy commands

- Localizar
  - `which`, `whereis`, `locate`
- Instalar software:
  - `dnf` (antes `yum`), `apt-get`, `yast`, `pkutil`, `swinstall`…
  - `configure` + `make install`



[dcc]

U.PORTO

FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

SHELL E SCRIPTING

Edição

- A edição da shell usa o modo emacs ou vi
  - `set -o vi`
  - `set -o emacs`
- Testar depois de efetuar alguns comandos:
  - `history`
  - `!2`
  - `Ctrl+r`
    - (e depois colocar algum texto de um comando usado)

[[]] Ver outros em 15 Examples To Master Linux Command Line History by Ramesh Natarajan

Shell redireccionamento

- O que faz?

```
$ find / -name core > /tmp/corefiles 2> /dev/null
```

  - Redireccionamento para STDOUT (>) e para STDERR (2>)
  - E se for ">>" ?
- E o que fazem?

```
$ cut –d: -f7 < /etc/passwd
$ ps aux | grep `whoami`
```

- Qual a diferença de | e < ou > ?

Imprimir páginas do Manual

- Aonde está o ficheiro:
  - `man -w man`
- Como ver o ficheiro do manual:
  - Está em gzip:

```
gunzip -c /usr/share/man/man1/man.1.gz
gunzip -c /usr/share/man/man1/man.1.gz | groff -mandoc
```

  - Output do `groff` é por defeito em ps

```
gunzip -c /usr/share/man/man1/man.1.gz | groff -mandoc > man.ps
evince man.ps
```

- Existe um erro na tabela
  - `gunzip -c /usr/share/man/man1/man.1.gz | groff -t -mandoc > man.ps`
- Outros:

[ ] Usar na linha acima o `ps2pdf` para gerar um man.pdf

[ ] Gerar um man.html

Vários comandos

- Resultado de?
  - `$ lpr man.ps && rm man.ps`
  - `$ lpr man.ps || lpr.cups man.ps`
- E de?
  - `$ df –h ; ls -lha`

Variáveis

- ```
  a=1
  b=$((2))
  c=$a+$b
  d=$(($a+$b))
  echo "$a + $b = $c / $a + $b = $d"
  ```
- Variáveis de ambiente definem algum funcionamento.
  - `env`
- Alias permitem definir alias para comandos
  - `alias`
  ```

Comandos de filtragem

- `cut, tee, sort, uniq, tee, wc, head, tail, grep`

- Exemplos:

```
$ cat /etc/passwd | tee /dev/tty | wc –l
$ tail –f /var/log/messages
```

find

- Estrutura:

`find <pastas_de_partida> <critérios_e_acções>`

`pastas_de_partida` – Pastas âncora da pesquisa

`critérios_e_acções` – Podem estar misturados e são avaliados da esquerda para a direita.

find - critérios

-atime n	ficheiro foi acedido n dias atrás
-mtime n	ficheiro foi modificado n dias atrás
-size n	ficheiro tem n blocks de tamanho (1 bloco = 512 bytes)
-type c	"ficheiro" é do tipo: f=regular, d=diretório, etc.
-fstype type	Tipo de sistema de ficheiros: 4.2 ou nfs, etc.
-name nome	nome do ficheiro é nome
-user usr	Dono do ficheiro é usr
-group grp	Grupo dono do ficheiro é grp
-perm p	Permissões do ficheiro são p

Baseado em Indiana Univ.

Find – Exemplos de Critérios

- `-mtime +7`: Modificado há mais de 7 dias
- `-atime -2`: Acedido há menos de dois dias
- `-size +5M`: Maior do que 5M
- `-atime +60 -mtime +120`
- `\( -atime +7 -o -mtime +30 \)`
- `! -name gold.dat -name \*.dat`
- `-perm 755`: Permissões = `rwxr-xr-x`
- `-perm -002`: Permissões de escrita para todos
- `-perm -4000`: Ficheiros setuid
- `-perm -2000`: Ficheiros setguid

Find – Ações e opções.

- Ações
  - `-print` - Escreve o caminho do ficheiro que está a ser avaliado
  - `-ls` - Versão detalhada (`ls -l`) do comando `print`.
  - `-exec cmd` – Executa o comando sobre o ficheiro.
  - `-ok cmd` – O mesmo que o anterior, só que pergunta antes de executar o comando.
- Opções
  - `-xdev` – Restringe a pesquisa ao sistema de ficheiros da pasta inicial que serve de raiz à pesquisa
  - `-prune` – Não desce abaixo da pasta encontrada.

AWK

- Criada por Al **A**ho, Peter **W**einberger, and Brian **K**ernighan
- linguagem simples orientada para descoberta de padrões.

Ver AWK, por Grimoire

AWK exemplos

```
$ ps -ef | grep "firefox" | awk '{print $1}'

$ ps -ef | grep "firefox" | awk '{print $1 " [" $7 "]"}'

find /home -user rmartins -type f -ls | \
      awk '{sum+=$7}; END {print "Total de gasto \
      em disco = " sum}'
```

Repetição de comandos - xargs

- `xargs` – automatizar a aplicação de comandos sobre um grupo de objetos.
  - Normalmente elemento final de uma pipe para aplicar repetidamente um comando aos objetos produzidos pela pipe

```
# ps -ef | grep "[fF]irefox" | awk '{print $2}' |\
    xargs kill
# find / -name "core*" –print0 | xargs -0 rm -f
```

  - Ver a opção `–delete` do `find`

Repetição de comandos - xargs

```
$ echo a b c d e f | xargs -n3 -I '{}' echo 'before {} after'
before a b c d e f after

$ echo a b c d e f | xargs -I '{}' -n3 echo before {} after
before {} after a b c
before {} after d e f

$ echo a b c d e f | xargs -n3 | xargs -I '{}' echo before {} \
      after
before a b c after
before d e f after
```

Bash Scripts

- Shebang line:

```
#!/bin/bash
```

- Podem ter qualquer comando interpretado pela Shell
- Ver exemplo do livro (pag. 39) para ir construindo na shell e depois usar `fc` para gravar num ficheiro.

Exemplo do livro

```
#!/bin/bash
function show_usage {
        echo "Usage: $0 source_dir dest_dir"
        exit 1
}
# Main program starts here
if [ $# -ne 2 ]; then
        show_usage
else # There are two arguments
        if [ -d $1 ]; then
                source_dir=$1
        else
                echo 'Invalid source directory'
                show_usage
        fi
        if [ -d $2 ]; then
                dest_dir=$2
        else
                echo 'Invalid dest. directory'
        show_usage
        fi
fi
printf "Source directory is ${source_dir}\n"
printf "Destination directory is ${dest_dir}\n“
```

Scripting não termina aqui

- Mais operadores, `while`, `for`, `arrays`
- Expressões regulares

- Perl

- Python

Boas práticas em scripts (e programas em geral)

- Ter uma mensagem de como usar (`show_usage()`)
- Validar inputs e valores calculados
- Retornar códigos de saída apropriados
- Convenções de nomes (variáveis e scripts)
- Comentários sobre objetivo e parâmetros
- Comentar o código
- Usar `-x` e `-n` da `bash` para ver os comandos e testar a sintaxe respetivamente (ver set builtin)
- Erros devem ir para o `STDERR` e não `STDOUT`

Resumo

- O SysAdmin
- Pontos a saber
- Shell e Scripting