

Sistemas Operativos

2º ano – 1º sem.

2022 – 2023

Conceitos básicos de UNIX

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Tópicos

Visão muito geral de sistema operativo Unix/Linux

Ambiente de execução UNIX

Comandos básicos para utilização do sistema de ficheiros

Particionamento do sistemas de ficheiros

Gestão de contas e password.

Mecanismos de segurança.

Elevação temporária de privilégios. `setuid` e `sudo`

Configuração de tarefas e de *boot loader*

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Visão esquemática de um sistema operativo

Utilizador

Aplicações utilizador

Sistema operativo

Hardware

- Cada camada interage apenas com as que estão imediatamente adjacentes
- As aplicações *utilizador*
 - Não fazem parte do sistema operativo
 - Inclui-se aqui a interface com o utilizador
 - Não têm capacidade de interação com o hardware
- O sistema operativo
 - Tem uma estrutura interna com maior ou menor complexidade, dependendo da arquitetura usada
 - Inclui os *device drivers*

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Visão esquemática de um sistema operativo

Utilizador

App1 | *App2* | *UI/Shell*

Sistema operativo

(vários componentes)

Hardware

- Cada camada interage apenas com as que estão imediatamente adjacentes
- As aplicações *utilizador*
 - Não fazem parte do sistema operativo
 - Inclui-se aqui a interface com o utilizador
 - Não têm capacidade de interação com o hardware
- O sistema operativo
 - Tem uma estrutura interna com maior ou menor complexidade, dependendo da arquitetura usada
 - Inclui os *device drivers*

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Visão muito geral de um sistema operativo

- O sistema operativo é constituído por um conjunto de programas
 - Núcleo (kernel).
 - É a parte mais central do sistema. Controla todos os aspetos da máquina
 - Corre em modo mais privilegiado (o processador reconhece mais instruções)
 - Pode ser um programa monolítico, ou uma coleção de vários programas, dependendo da arquitetura.
 - Gestores de dispositivos (device drivers).
 - Podem ou não ser considerados como parte do núcleo.
 - Correm também com privilégios elevados
 - Outros programas do sistema operativo.
 - Fazem parte do sistema mas não são o núcleo.
 - Em termos de privilégios, são “meras” aplicações
 - Exemplo: a Shell (consola ou gráfica)

Os restantes programas a executar no computador são meras aplicações. Correm num modo não privilegiado. Se quiserem aceder aos recursos da máquina têm que passar pelo sistema operativo (API)

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Ambiente de utilização Unix

- Após o *login* bem sucedido, o sistema lança uma *shell* em nome do utilizador
- A *shell* é um programa que interage com o utilizador e o sistema operativo, encaminhando as acções do utilizador para operações sobre o sistema
- Pode ser consola ou gráfica
 - **Consola**: o utilizador especifica o que pretende através de **comandos** escritos
 - Exemplo: bash
 - **Gráfica**: o utilizador especifica o que pretende através de ícones e menús
 - Exemplos: Gnome, KDE
- A *shell* é um programa como os outros (tecnicamente não faz parte do sistema). Corre “em nome” do utilizador e todas as operações que desencadeia e programas que lança são implicitamente em nome desse utilizador

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Ambiente de execução Unix

Processo vs. programa

- Cada **programa** que se executa corre no contexto de um **processo**.
 - Só se consegue correr um programa no contexto de um processo
- O **processo** define diversas características, afetando a forma como o programa no seu interior é executado

Isto inclui aspetos importantes **definidos pelo sistema**

- Zona de memória
- Prioridade
- Identificação do utilizador (e consequentemente os seus privilégios)
- Outros recursos controlados pelo sistema
- Inclui também **aspetos de ambiente, modificáveis pelo utilizador**
 - Diretoria actual / de trabalho
 - Variáveis de ambiente

-> ***O conceito de processo será analisado com cuidado mais adiante***

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Ambiente de execução Unix

- A única forma de criar um processo é através de uma relação pai-filho: um processo duplica-se criando um novo (“filho”)
 - Anterior (original) = “pai”, novo = “filho”
 - O processo filho adquire (herda) automaticamente algumas características do processo pai
 - Variáveis de ambiente
 - Ficheiros abertos (pode ser controlado)
 - Programa em execução
 - Identificação de utilizador
 - Diretoria actual

Esta relação processo pai -> processo filho é muito importante (slide seguinte)

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Ambiente de execução Unix

Processo pai -> processo filho - Relevância

- Muitas das características de um processo são definidas a partir das características do processo “pai”
 - Exemplo: a identificação do utilizador (e por inerência, os privilégios)**
 - O sistema apenas necessita de lançar a shell em nome do utilizador. A partir daí, os programas que o utilizador lança (usando a shell) correm em nome desse utilizador, com as suas permissões e recursos (pode ser modificado – exemplo: sudo)
- Um processo pode afetar o ambiente de execução do processo que vai lançar
 - Basta configurar as suas características antes de lançar o processo e o processo novo que lança herda essas características
 - Exemplo: diretoria de trabalho
- Um processo não consegue afetar as características do seu processo pai
 - Mesmo exemplo: a diretoria de trabalho

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Ambiente de execução Unix

- Ao criar um novo processo, o processo pai pode controlar várias das características que o processo filho vai ter
 - O processo filho não consegue controlar o ambiente do processo pai
 - Um exemplo onde isto se nota: propagação de variáveis de ambiente do processo pai para o filho e apenas neste sentido

Relevância

A *shell* (consola, terminal) pode definir a forma como os programas que invoca se executam pois é a *shell* que cria os processos onde esses programas correm.

Exemplo de aplicação: redirecionamento (exemplo: *ls | more*)

O assunto de criação de processos vai ser visto mais adiante com o devido detalhe

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Variáveis de ambiente

- Definem aspectos operacionais do ambiente de trabalho do ambiente de execução dos programas e do ambiente do utilizador
- Forma: nome=valor
- Exemplos
 - PATH Identifica as directorias onde são procurados os programas
 - USER Contém o *username* do utilizador actual
 - PWD Contém o nome da directoria actual
- Podem ser criadas e modificadas por comandos. Exemplos:
 - declare Declara e atribui atributos de variáveis
 - export Marca variável para propagação para os programas seguintes
 - unset Elimina variável
- As variáveis afectam tanto a execução de programas como o ambiente do utilizador
- São propagadas apenas para a frente: para os programas lançados

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Comandos Unix – alguns exemplos

Forma: *comando opções alvo ls -la*

/tmp

- **cd** – muda de directoria (comando interno)
- **ls** – mostra ficheiros e direct. -a -l -d -s
- **mkdir** – cria directorias -p -m
- **cp** – copia ficheiros -u
- **mv** – mov e/rename ficheiros -f -i
- **rm** – apaga ficheiros -p -r -d
- **rmdir** – apaga directorias -p
- **cat** – mostra ficheiros
- **chmod** – muda permissões de ficheiros
- **chown** – muda dono do ficheiro
- **sudo** – executa em nome de outro utilizador
- redireccionamento: < > |

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Lógica de utilização de Unix

Ambiente consola - Linha de comandos

- Cada comando é um programa
- Faz uma tarefa muito simples
- Podem ser encadeados e construir funcionalidade complexa
- As operações podem ser automatizáveis em scripts – cenário ideal para administração e manutenção de tarefas

Ambiente gráfico – desktop

- Situação típica com janelas e ícones
- Existem dezenas de alternativas
- Podem ser configuradas várias em simultâneo
- Menos interessantes para administração e gestão

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Comandos Unix:

- A maior parte dos comandos são simples programas executáveis que se encontram no disco e são executados quando invocados
 - Isto significa que não são funcionalidades internas à *shell*
 - Por isso são designados de “comandos externos”
 - O sistema é facilmente extensível porque a qualquer altura se podem acrescentar comandos novos
- A variável PATH indica as directorias onde são procurados os programas a executar.
 - Exemplo: *whereis ls* indica onde está o programa correspondente ao comando “ls”
 - *whereis* é também um comando externo
 - Normalmente a própria directoria não é pesquisada para executar programas (não está na PATH) e isso é por uma questão de segurança

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Comandos Unix:

- Há “comandos” que não correspondem a ficheiros executáveis
- Exemplos: `cd`, `set`, `export`, `declare`
- O comando **`cd`** não é um comando externo
 - Está implementado directamente no interpretador de comandos (“ex. *bash*”) e ***nem podia ser de outra forma***
- Exercício/desafio: por que é que não podia ser de outra forma?

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Ficheiros em Unix – output do comando `ls`

Comando `ls`

Exemplo de *output*

```
drwxr-xr-x  2 joao joao      4096 Set 19 20:41 Documents
```

- Uma letra indicativa do tipo de ficheiro
 - `d` → directoria
 - `-` → ficheiro regular
 - `p` → *named pipe* (mecanismo de comunicação)
 - `s` → *socket* (mecanismo de comunicação em rede)
 - `l` → link simbólico para outro ficheiro (indicado)
 - `b` → device driver de bloco (exemplo: de disco)
 - `c` → device driver de carácter (exemplo: porto série)

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Ficheiros em Unix – output do comando ls

Comando ls

- Três grupos de permissões (comando chmod)
 - **r** → **read**
 - **w** → **write**
 - **x** → **execute**

A letra aparece: tem permissão.
Aparece “-” no lugar da letra: não tem essa permissão

 - Organizados como **dono** (*user*), **grupo** (*group*), **outros** (*others*) (três letras para cada)
 - Podem ser descritos em octal: 1 bit para cada letra, 3 bits por grupo
 - Outras permissões possíveis:
 - **t** → eliminação restrita (só o dono pode apagar).
 - » Exercício: onde fará sentido usar esta opção?
 - **S** → (em conjunto com execução): bit setuid / setgid ligado

DEIS/ISEC

• T

Sistemas Operativos – 2022/23

João Durães

Ficheiros em Unix – output do comando ls

Comando ls

Exemplo de *output*

```
drwxr-xr-x  2 joao joao      4096 Set 19 20:41 Documents
```

- **Outros campos**
 - Dono do ficheiro
 - Grupo do ficheiro
 - Tamanho
 - Data e hora
 - Nome

-> Os ficheiros são um aspecto central no sistema Unix. Muitos recursos são usados como se fossem ficheiros

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Sistemas de ficheiros

Organizado em directorias de carácter bem definido

/

- Ponto inicial do sistemas de ficheiros

/bin

- Ficheiros executáveis (ex. comandos) (link para /usr/bin)

/dev

- Contém os gestores dos dispositivos (“device drivers”)

/etc

- Contém bibliotecas e ficheiros de configuração (ex.: passwd,)

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Sistemas de ficheiros

/lib

- Bibliotecas do sistema e de software de carácter geral

/boot

- Contém os ficheiros de arranque (executáveis e configuração) e o kernel

/home

- Contém as directorias pessoais dos utilizadores

/mnt

- Usado para “montar” outras partições ou dispositivos (ex: cdrom, pen usb)

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Sistemas de ficheiros

/opt

- Contém software de carácter opcional, específico a um sistema particular
- Exemplo: um servidor ou programa instalado apenas nesta máquina

/proc

- Contém pseudo-ficheiros com informação dinâmica (*runtime*) do sistema, por exemplo, acerca dos processos em execução

/tmp

- Ficheiros de carácter temporário. Normalmente acessível a todos os utiliz.

Sistemas de ficheiros

/usr

- Ficheiros de carácter geral (“não categorizados”), mas de âmbito do sistema e não de utilizadores em particular

/var

- Usado para ficheiros de tamanho variável (ex.: ficheiros log)
- Em alguns casos usado como se fosse /opt

/sbin

- Programas (ficheiros executáveis) de uso restrito (para o super utilizador), normalmente para administração. Normalmente link para /usr/sbin

/srv

- Para instalação de programas de carácter “servidor” (exemplo: servidores web)

Sistema de ficheiros Unix

Normalmente organizado em partições independentes com mountpoints em directorias específicas e de uso bem definido

Existe uma única estrutura de directorias e sub-directorias e determinadas sub-directorias mapeam para outras partições (mesmo que noutro disco)

A directoria onde a partição está mapeada é o mountpoint dessa partição

É possível que cada partição tenha o seu próprio sistema de ficheiros, diferente dos restantes, desde que suportados pelo sistema

Exemplo: CDROM -> Mapeado em /mnt/cdrom
(o mountpoint /mnt/cdrom poderia ser outro)

Comandos envolvidos (uso restrito)

mount **umount**

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Sistema de ficheiros Unix

Muitas vezes algumas directorias centrais estão mesmo mapeadas em partições independentes

Casos típicos

Razão

- | | |
|------------|---------------------------------------------------------------------------------|
| • /boot | Para garantir que tem sempre algum espaço livre |
| • /home | Para poder levar a partição/disco para outro sistema |
| • /tmp | Tem um sistema de ficheiros afinado para fich. temporários |
| • /var | Tem um sistema de ficheiros otimizados para ficheiros de tamanho muito variável |
| • /mnt/xxx | /mnt serve especificamente para agrupar mountpoints |

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Sistemas de ficheiros

Organização em partições independentes montadas (“penduradas”) no sistema principal (o sistema com a directoria principal “/”)

Razões típicas para esta organização

- Garantir que os ficheiros de uma directoria (partição/dispositivo) não extravasam para o resto do sistema de ficheiros (ex., /tmp),
- Garantir que o excesso de ficheiros no resto do disco não passa para a directoria (ex., /boot)
- Implementações específicas de partes do sistema de ficheiros (ex: /proc)
 - O conteúdo desta directoria são pseudo-ficheiros
- Portabilidade dos dados para outra máquinas (ex., /home)
- Em geral: permitir a capacidade de gestão de cada parte do sistema de ficheiros de forma independente das outras partes
 - Exemplo: necessidade de formas diferenciadas de segurança ou fiabilidade consoante cada parte do sistema

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

fstab (file systems table)

/etc/fstab

Auxilia a forma como os vários dispositivos (partições, etc.) são montados no sistema de ficheiros

Exemplo

# device name	mount point	fs-type	options	dump-freq	pass-n
LABEL= /	/	ext3	defaults	1	1
/dev/hda6	swap	swap	defaults	0	0
none	/dev/pts	devpts	gid=5,mode=620	0	0
none	/proc	proc	defaults	0	0
none	/dev/shm	tmpfs	defaults	0	0
# my removable media					
/dev/cdrom	/mnt/cdrom	udf,iso9660	noauto,owner,ro	0	0
/dev/fd0	/mnt/floppy	auto	noauto,owner	0	0
# my NTFS Windows XP partition					
/dev/hda1	/mnt/WinXP	ntfs	ro,defaults	0	0
# my files partition shared by windows and linux					
/dev/hda7	/mnt/shared	vfat	umask=000	0	0

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Significado das colunas

1. Nome do dispositivo ou a sua localização
2. mount point – em que ponto é que a partição vai ser inserida no sistema de ficheiros global
3. Tipo do sistema de ficheiros
4. Opções (slide seguinte)
5. Opções para arquivo da partição com utilitário *dump*.
6. Ordem pela qual o utilitário *fsck* (file system check) analisa as partições à procura de erros quando o sistema arranca

Tipos de sistemas (coluna 3) – exemplos:

- | | |
|-----------|----------------------------------|
| – fat | MSDOS / Win85/98 |
| – ext2 | Linux |
| – iso9660 | CDROM |
| – Ntfs | WindowsNT (2000,xp, vista, 7-10) |

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Exemplo de opções para a coluna 4

- nosuid – impede bit setuid nos binários nesse dispositivo
- auto / noauto – controla o mounting automático do dispositivo
- exec / noexec – controla a permissão de executar binários no dispositivo
- ro (read-only) / rw (read-write) – controla o acesso read/write
- uid=xxx, gid=xxx especifica user/group id para a posse dos ficheiros nesse dispositivo
- dmask=xxx, fmask=xxx – especifica bits de permissão (octal) para directorias e ficheiros tal como umask (ver umask)

-> Algumas opções não são suportadas em todos os dispositivos

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Exemplo: opções numa partição NTFS montada num sistema linux:

```
nosuid,noexec,rw,auto,uid=0,gid=999, dmask=007,fmask=117
```

- Não permite bit suid
- Não permite executar programas nesse dispositivo
- Montado como read/write
- Automaticamente montado no arranque
- Ficheiros são vistos como pertencendo ao user com ID 0 (root) e ao grupo com ID 999
- Directorias adquirem permissão `rw-rw---`
- Ficheiros adquirem permissões `rw-rw---`
- EM algumas situações o sistema consegue entender o tipo de sistema de ficheiros (NTFS, FAT, vboxsf, etc)

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Exemplo: `fstab` para directoria partilhada numa máquina VirtualBox

A informação é apresentada com as linhas quebradas por falta de espaço no slide

o nome da directoria não é o mesmo que o dos documentos de instalação

```
share /mnt/dirpart vboxsf
nosuid,noexec,rw,auto,uid=0,gid=998,dmask=007,fmask=117
0 0
```

Significado dos campos e opções principais

- **share** -> identificador do **dispositivo** (nome definido no virtualbox)
- **/mnt/partilhado** -> Directoria no FS do linux onde aparece o conteúdo partilhado
- **vboxsf** -> Tipo de sistema de ficheiros (Virtual Box Shared Folder)
- **uid=0** -> Vê os ficheiros como pertencendo ao user 0 (root)
- **gid=998** -> Vê ficheiros como pertencendo ao grupo 998 (vboxsf)
Nesta configuração a directoria é vista como pertencendo ao root. Os utilizadores “normais” que queiram aceder aos ficheiros deverão pertencer ao grupo vboxsf (tipicamente: GID 998)
- **rw** -> Os ficheiros em share (mnt/dirpart) vistos como tendo permissões **read** e **write**
- **auto** -> O sistema tenta fazer o mount automaticamente no arranque
- **noexec** -> Não permite execução de ficheiros existentes em share (/mnt/dirpart)
- **nosuid** -> Não permite bit setuid ligado (sem necessidade uma vez que “noexec”)
- **dmask=007** -> Definem permissões `rw-rw---` para as subdirectorias
- **fmask=117** -> Define permissões `rw-rw---` para os ficheiros

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Gestão de utilizadores

Organizados em grupos

- Um utilizador tem um grupo primário mas pode pertencer a mais do que um grupo

Ficheiros envolvidos

- /etc/passwd
- /etc/group
- /etc/shadow
- /etc/gshadow

Alguns comandos envolvidos

useradd userdel whoami id passwd

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

/etc/passwd

- Contém a identificação das contas de utilizador

Tem, entre outros dados

- Nome da conta (*login*)
- Password (codificada)
- Directoria pessoal (*home dir*)
- ID no sistema

- Este ficheiro está em *clear text*

- Pode ser lido e escrito pelo administrador
- Pode ser lido pelos utilizadores → para que fim?
 - Este aspecto é um ponto fraco de segurança → por que razão?

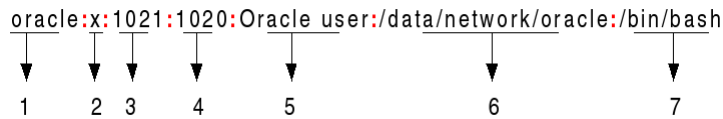
DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

/etc/passwd

oracle:x:1021:1020:Oracle user:/data/network/oracle:/bin/bash



1 2 3 4 5 6 7

1. **Username:** "login" do utilizador. Entre 1 e 32 caracteres.
2. **Password:** Password encriptada, ou então um **X** que indica que a password (encriptada) está no ficheiro /etc/shadow
3. **User ID (UID):** Identificação numérica do utilizador (única a nível do sistema). 0 = root, 1-99 = reservados para contas predefinidas. 100-999 = reservadas para contas e grupos administrativos.
4. **Group ID (GID):** ID do grupo principal (do utiliz.) (no ficheiro /etc/group)
5. **User ID Info:** Campo de comentário acerca do utilizador, por exemplo, nome completo e telefone. O comando **finger** usa esta informação.
6. **Home directory:** Caminho absoluto da directoria pessoal do utilizador. Se não for especificada será a raiz ("/"), mas isso não implica direitos de escrita
7. **Command/shell:** Pathname para o programa que serve de interpretador de comandos para este utilizador. Exemplo:/bin/bash). (Tipicamente uma shell; tipicamente a bash, mas pode ser qualquer programa.

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

/etc/shadow

- Contém a password dos utilizadores
 - Pode ser lido e escrito pelo administrador
 - Não pode ser lido (nem escrito) pelos utilizadores
 - Isto dificulta ataques de força bruta para obter passwords
 - Contém informação acerca das regras de modificação de password (prazos)
- **Exercício/Desafio**
 - Como é que se podia montar um ataque se um utilizador normal pudesse ler o ficheiro /etc/shadow (ou seja, ler a password codificada)?

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

/etc/shadow

Exemplo

smithj:Ep6mckrOLChF.:10063:0:99999:7:::

1. **Username:** "login" do utilizador (igual à do ficheiro etc/passwd)
2. **Password:** Password encriptada ou um * que indica que não é necessário password (má ideia)
3. Número de dias desde que a password foi modificada.
4. Número de dias até poder ser modificada.
5. Número de dias após qual a password deve mesmo ser modificada.
6. Número de dias até avisar que a password está a expirar.
7. Número de dias após password expirada que causa a conta ficar cancelada
8. Número de dias desde que a conta está cancelada.
9. Reservado.

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

/etc/shadow

Método de encriptação

Outro exemplo (focar apenas no campo relativo à password e nos \$)

smithj:\$1\$Etg2ExUZ\$F9NTP7omafhKllqaBMqng1:10063:0:99999:7:::

O primeiro \$... indica o algoritmo

- \$1 -> Hashing MD5
- \$2 -> Blowfish
- \$2a -> eksblowfish
- \$5 -> SHA-256
- \$6 -> SHA-512

Por omissão: Algoritmo DES com o programa crypt (não é o mais seguro)

Nota: não é para decorar esta tabela mas é para saber a lógica de como isto funciona

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

/etc/shadow

Método de encriptação

Outro exemplo (focar apenas no campo relativo à password e nos \$)

smithj:\$1\$Etg2ExUZ\$F9NTP7omafhKllqaBMqng1:10063:0:99999:7:::

Restantes \$...

- 2º -> Valor "salt" para configuração do algoritmo (valor aleatório para dificultar descriptação)
 - 3º -> resultado da "encriptação" da password com o salt usando o algoritmo especificado
 - A análise dos algoritmos de encriptação é importante mas seria noutras disciplinas. Exemplos: "fundamentos teóricos de encriptação" (basicamente = matemática), ou "segurança"
- Novamente: Exercício/Desafio
- Desde o último desafio, que novas ideias lhe ocorrem para montar um ataque se pudesse ler o ficheiro /etc/shadow?

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Validação de permissões de operações

- O UNIX valida as operações com base em
 - Identificação dos utilizadores
 - Pertença dos recursos (ex.: ficheiros de configuração)
 - Atribuição de permissões
 - RWX | RWX | RWX
 - Dono | grupo | outros

R = Read, W = Write, X = eXecute

Comando para mudança das permissões: comando **chmod**

Em unix, os recursos manifestam-se como ficheiros ou pseudo-ficheiros. Assim, a operações dos utilizadores são verificadas analisado as permissões expressas nos (pseudo-)ficheiros correspondentes aos recursos manipulados

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Permissões e privilégios de programas

- Quem valida as permissões e privilégios?
- Será o programa?

```
main() {  
    if (...user-pode-fazer-isso...)  
        /* faz a operação pretendida */  
    else {  
        printf("lamento, EU não deixo fazer isso");  
    }
```

- Ou será o sistema?

```
main() {  
    if (executaOperaçãoPretendida(...) == 0) /* função sistema */  
        printf("lamento, O SISTEMA não deixa fazer isso");  
    else  
        printf("ok, operação concluída");  
}
```

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Permissões e privilégios de programas

- É **obviamente** o sistema que faz a validação
- Isto pode ser testado com um simples programa
 - Exemplo, um programa que tenta ler o ficheiro /etc/shadow
- Porquê “obviamente”?

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Permissões e privilégios de programas

- É **obviamente** o sistema que faz a validação
- Isto pode ser testado com um simples programa
 - Exemplo, um programa que tenta ler o ficheiro /etc/shadow
- Porquê “obviamente”?
 - Porque se fosse o programa a testar as permissões, qualquer um com conhecimentos de programação poderia fazer o seu próprio programa (sem testes nenhuns) e fazer o que bem entendesse (*mais ou menos*)
 - Cada processo está associado a um utilizador e o sistema conhece essa associação. As operações desencadeadas por esse processo são validadas pelo sistema com base nas permissões associadas ao utilizador associado a esse processo (há mais detalhes sobre isto)

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Permissões e privilégios de programas

*“Porque se fosse o programa a testar as permissões, qualquer um com conhecimentos de programação poderia fazer o seu próprio programa (sem testes nenhuns) e fazer o que bem entendesse (*mais ou menos*)”*

Acerca do “mais ou menos”

- Em última análise, o programador poderia tentar replicar o código do sistema dentro do seu programa e remover os testes e fazer o que bem entendesse na máquina.
- O que o impede?
 - > o **sistema operativo** e o **hardware**, usando o **mecanismo que permite ao processador distinguir código do sistema de código de aplicações utilizador** (já mencionado atrás e a abordar novamente mais adiante)

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Permissões e privilégios de programas

O que impede o programa de replica as operações feitas pelo sistema e fazer ele próprio as operações? -> o **sistema operativo** e o **hardware** (detalhe a abordar novamente mais adiante)

- O código do **núcleo** do sistema corre numa **zona de memória** reconhecida pelo **processador** como podendo conter determinadas instruções críticas
 - Essas instruções são necessárias para (exemplos)
 - Ver/mudar a memória toda do computador
 - Por e tirar programas em execução
 - Interagir directamente com o hardware
- Se essas instruções aparecerem em zonas de memória que não as do núcleo, o **processador** recusa-se a executá-las e **avisa** o sistema operativo
 - Assim, os processos “normais” não conseguem fazer nada que ponha em causa a máquina e outros utilizadores. A única forma é **passar pelo sistema (núcleo)**, o qual valida tudo em função da identificação do utilizador

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Permissões e privilégios de programas

Exemplo / demonstração

-> **Fazer e depois executar um programa que tenta usar a instrução HLT**

Essa instrução pára o processador (ou um núcleo do processador)

Description

HALT stops instruction execution and places the 80386 in a HALT state. An enabled interrupt, NMI, or a reset will resume execution. If an interrupt (including NMI) is used to resume execution after HLT, the saved CS:IP (or CS:EIP) value points to the instruction following HLT.

Flags Affected

None

Protected Mode Exceptions

HLT is a privileged instruction; #GP(0) if the current privilege level is not 0

A execução dessa instrução afectaria **toda a máquina e todos os outros processos e utilizadores**. Não parece ser boa ideia permitir a sua execução em “meros” processos-utilizador. O que acontecerá se se executar tal programa?

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Permissões e privilégios de programas

Exemplo / demonstração

→ Código do programa

```
#include <stdio.h>

int main(int argc, char ** argv) {
    printf("\n\nantes da instrução\n");
    __asm__ ( "hlt;" ); /* tenta parar o processador */
    printf("\nde pois da instrução\n\n");
    return 0;
}
```

→ Resultado da sua execução

```
antes da instrução
Segmentation fault
```

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Permissões e privilégios de programas

Exemplo / demonstração

E se o programa fosse executado como root ou com sudo?

→ O resultado seria o mesmo

Obviamente

- O utilizador *root* continua a ser um utilizador e os seus programas são executados em “meros” processos-utilizador.
- Não deve ser confundido privilégios de utilizador (o que é que o utilizador pode ou não fazer) com privilégios de código (o que é que o código privilegiado no núcleo do sistema pode fazer quando comparado com código de processos-utilizador)

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Permissões e privilégios de programas

*“A única forma é **passar pelo sistema (núcleo)**, o qual valida tudo em função da identificação do utilizador”*

O que significa isso?

- Qualquer operação desencadeada pelo código do programa que envolva recursos do sistema **passam sempre** pelo sistema, que valida tudo.
- Exemplo: **um simples “printf” passa pelo sistema**
Afinal de contas, usa o recurso “ecrã”, que pertence ao Sistema e não ao programa

printf → **fprintf** → **fwrite** → **write** → **device driver da gráfica** → ecrã

Permissões e privilégios de programas

um simples “printf” passa pelo sistema

Afinal de contas, usa o recurso “ecrã”, que pertence ao Sistema e não ao programa

printf → **fprintf** → **fwrite** → **write** → **device driver da gráfica** → ecrã
Sistema / núcleo so sistema

- As funções a negro são meras funções biblioteca, no espaço de memória do programa “utilizador”
- As funções a azul pertencem ao sistema operativo, no núcleo, numa zona de memória à qual o processador reconhece e aceita instruções necessárias para interagir com o periférico
 - O processo utilizador não consegue ver nem alterar essa memória

Permissões e privilégios de programas

- Execução de programas
 - É criado um novo processo para conter o código do programa pretendido
 - O novo processo tem as mesmas permissões e privilégios do processo que o lançou

Assim, quando um utilizador lança a execução de um programa, o novo programa corre com as mesmas permissões desse utilizador

 - Esta é a situação *default*
 - **Pergunta de revisão:** Como é que isso ocorre?

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Permissões e privilégios de programas

- Execução de programas
 - É criado um novo processo para conter o código do programa pretendido
 - O novo processo tem as mesmas permissões e privilégios do processo que o lançou

Assim, quando um utilizador lança a execução de um programa, o novo programa corre com as mesmas permissões desse utilizador

 - Esta é a situação *default*
 - **Pergunta de revisão:** Como é que isso ocorre?
 - O utilizador está a usar a sua shell, com permissões associadas a si
 - A shell é quem lança o novo processo
 - Logo, o novo programa (processo) tem as mesmas permissões e privilégios que o utilizador

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Execução privilegiada temporária

Problema:

- Como permitir a alguns utilizadores privilégio elevados sem partilhar a totalidade dos direitos de administração?
- Para, por exemplo: ***permitir mudar a sua password***
 - Implica escrever em /etc/shadow
 - Os utilizadores normais não têm permissões para tal
 - No entanto podem mudar a password -> **como?**
- Pretende-se dizer caso a caso (comando a comando) quem pode executar
- Solução: setuid/setgid e sudo

DEIS/SEC

Sistemas Operativos – 2022/23

João Durães

setuid – **Set User ID** upon execution

- Cada processo (“programa”) corre normalmente com os privilégios:
 - **setuid ligado** → ***do dono do executável***: o processo pode fazer o que o utilizador dono também pode
 - **setuid desligado** → ***de quem lança o processo***: o processo só pode fazer o que esse utilizador também pode

Mudança do bit setuid/setgid

- `chmod u+s` (adquire permissões do utilizador dono - setuid)
- `chmod g+s` (adquire permissões do grupo dono - setgid)
- Os programas que têm o bit setuid/setgid ligado devem ser cuidadosamente construídos de forma a não terem vulnerabilidades → **Pergunta-desafio: “por que razão?”**

DEIS/SEC

Sistemas Operativos – 2022/23

João Durães

setuid – **Set User ID** upon execution

- Exemplo de `ls -l` para um programa com setuid ligado

```
-rwsr-xr-x 1 root root 163988 Jun  5 2017 /usr/bin/exemplo
```

Os programas com o bit setuid são potencialmente perigosos (*porquê?*)
As *shells* habituais apresentam esses programas com cores bem visíveis para assinalar esse perigo

Exemplo:

```
lrwxrwxrwx 1 root root      22 May 10 2017 strings -> i686-linux-gnu-strings
lrwxrwxrwx 1 root root      20 May 10 2017 strip  -> i686-linux-gnu-strip
-rwsr-xr-x 1 root root 163988 Jun  5 2017 sudo
lrwxrwxrwx 1 root root       4 Jun  5 2017 sudoedit -> sudo
-rwxr-xr-x 1 root root    42728 Jun  5 2017 sudoreplay
```

Pergunta de raciocínio: por que é que os programas com setuid ligado são mais perigosos (mesmo os que pertencem ao sistema)?

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

setuid – **Set User ID** upon execution

- Voltando à questão inicial

Pergunta de avaliação de atenção na aula

Como é que o comando ***passwd***, que ***pode ser executado pelo utilizador***, consegue mudar a password se isso implica modificar o ficheiro protegido `/etc/shadow`?

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

setuid – **Set User ID** upon execution

- Voltando à questão inicial – mudar a password

Pergunta de avaliação de atenção na aula

*Como é que o comando **passwd**, que **pode ser executado pelo utilizador**, consegue mudar a password se implica modificar o ficheiro protegido `/etc/shadow`?*

-> (Obviamente), o programa **passwd** é “dos tais” que têm o bit setuid ligado e pertencem ao root

```
ls -la /usr/bin/passwd  
-rwsr-xr-x 1 root root 63736 Jul 27 2018 /usr/bin/passwd
```

Trata-se de um programa que vem com o sistema, cujo código é conhecido (sabe-se o que faz / é de confiança) e faz apenas aquilo que está previsto que faça - o facto de correr como root não significa que faça qualquer coisa que apeteça ao utilizador

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

setuid – **Set User ID** upon execution

Desafio:

- Assuma que *pode oferecer programas a outros utilizadores* e que *pode ligar o bit setuid*.
 - Mudar permissões incluindo bit setuid: **chmod**
 - Mudar o dono de um ficheiro: **chown**

(afinal, se calhar, pode mesmo fazer isso)

Questão: Como poderia então montar um ataque para obter o controlo total da máquina?

(Nota: não pode oferecer programas a outros utilizadores a não ser que seja *root*)

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

/etc/sudoers

Comando sudo

- Permite a execução de comandos específicos a utilizadores específicos
 - O utilizador executa o comando sudo, especificando por parâmetro o comando que deseja correr
 - Exemplo: sudo fdisk
- O comando sudo corre com privilégios de administrador
 - **Pergunta de revisão:**→ Como é que isso ocorre?

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

/etc/sudoers

Comando sudo

- Permite a execução de comandos específicos a utilizadores específicos
 - O utilizador executa o comando sudo, especificando por parâmetro o comando que deseja correr
 - Exemplo: sudo fdisk
 - O comando sudo corre com privilégios de administrador (root)
 - **Pergunta de revisão:**→ Como é que isso ocorre?

Resposta

O ficheiro sudo pertence ao root e tem o bit setuid ligado

```
lrwxrwxrwx 1 root root      22 May 10 2017 strings -> i686-linux-gnu-string-
lrwxrwxrwx 1 root root      20 May 10 2017 strip  -> i686-linux-gnu-strip
-rwsr-xr-x 1 root root 163988 Jun  5 2017 sudo
lrwxrwxrwx 1 root root      4 Jun  5 2017 sudoedit -> sudo
-rwxr-xr-x 1 root root    42728 Jun  5 2017 sudoreplay
```

Ou seja: quando é executado, corre com os privilégios do seu dono, que é o root

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

/etc/sudoers

Comando sudo (continuação)

- Posteriormente o comando valida a identificação do utilizador pedindo-lhe a password
 - Para garantir que é mesmo o utilizador autorizado quem está na consola
- De seguida verifica se o utilizador pertence ao grupo sudo
- Se o ficheiro sudoers existir, verifica também que comandos específicos podem ser usados com sudo por esse utilizador
- Por fim executa o comando desejado lançando-o a partir do seu próprio contexto de execução (=“executa-o no contexto do seu próprio processo”, que já tem privilégios elevados)

Pergunta-desafio: avaliar o entendimento de conceitos

→ Qual a diferença entre *setuid/setgid* e *sudo*?

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Sudoers

/etc/sudoers (em versões mais antigas)

Controla o acesso a programas privilegiados por parte de utilizadores não privilegiados (evita a necessidade de partilhar a *password* do *root*)

Para aceder a um programa privilegiado o utilizador comum deve escrever *sudo comando*

Exemplo

```
User_Alias    FULLTIMERS = millert, mikef, dowdy
User_Alias    PARTTIMERS = bostley, jwfox, crawl
Cmnd_Alias    KILL = /usr/bin/kill
```

```
# full time sysadmins can run anything on any machine without a password
```

```
FULLTIMERS ALL = NOPASSWD: ALL
```

```
# part time sysadmins may run anything but need a password
```

```
PARTTIMERS ALL = ALL
```

```
# matt needs to be able to kill things on his workstation when they get hung.
```

```
matt          valkyrie = KILL
```

```
# joe may su only to operator
```

```
joe  ALL = /usr/bin/su operator
```

Versões mais recentes: acrescentar o utilizador ao grupo *sudo* (GID 27)

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Outros ficheiros de configuração

- Existem inúmeros serviços e funcionalidades no Unix. Cada um é configurado por simples ficheiros de texto

Ficheiros de texto:

- Podem ser lidos facilmente sem necessidade de programas específicos (exemplo: *reedit* como no Windows)
- Em caso de problemas de configuração na máquina, um simples editor de texto é o suficiente pra consertar o problema

- Nos slides seguintes, dois exemplos:
 - Configuração de agendamento de tarefas
 - Configuração do bootloader

crontab (cron table)

/etc/crontab

Ficheiro que controla execução periódica de programas (tarefas)

Existe um utilitário com o mesmo nome para editar este ficheiro.

crontab.allow e crontab.deny → Controla os utilizadores que podem alterar a tabela de cron

cron – É o processo responsável por lançar os programas especificados na tabela

Campo	Função
1º	Minuto
2º	Hora
3º	Dia do mês
4º	Mês
5º	Dia da semana (0 = Domingo)
6º	Utilizador
7º	Programa a executar

Separador = *tab*

"*" = Todos

Pode-se usar mais que um valor desde que separado por ",",

Exemplo

01	*	*	*	*	root	run-parts /etc/cron.hourly
02	4	*	*	*	root	run-parts /etc/cron.daily
22	4	*	*	0	root	run-parts /etc/cron.weekly
42	4	1	*	*	root	run-parts /etc/cron.monthly

run-parts

Executa todos os *scripts* (executáveis) dentro de uma determinada diretoria. Usado para executar programas de hora a hora, diariamente, semanalmente ou mensalmente

Directoria	Período
/etc/cron.hourly	De hora a hora
/etc/cron.daily	Diariamente
/etc/cron.weekly	Semanalmente
/etc/cron.monthly	Mensalmente

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

Configuração do GRUB (GRand Unified Boot loader)

/boot/grub/menu.lst (pode variar por distribuição)

- Este **loader** é bastante poderoso e permite carregar sistemas em outros discos, ocultar partições, entre outras funcionalidades
- Identificação do dispositivo com o kernel: (disco, nº da partição)

Exemplo simples

```
title          Ubuntu, kernel 2.6.20-16-generic
root          (hd0,0)
kernel        /boot/vmlinuz-2.6.20-16-generic
              root=UUID=4a62f231-3ad9-4cf8-bbdd-27586ab374b6 ro quiet
              splash

title          Ubuntu, kernel 2.6.20-16-generic (recovery
mode)
root          (hd0,0)
kernel        /boot/vmlinuz-2.6.20-16-generic
              root=UUID=4a62f231-3ad9-4cf8-bbdd-27586ab374b6 ro single
```

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães