Engenharia da Segurança em SI

Universidade do Minho

2023/2024

Tópico 2 - Controlo de Acesso

TP3

PG53680 — Beatriz Campos da Silva Reis — MEGSI-SIOS
PG53772 — Diogo Almeida Esteves — MEGSI-ESI
PG53844 — Gonçalo Ribeiro Regadas — MEGSI-ESI
PG53921 — João Martins Dias — MEGSI-ESI
PG54275 — Vítor Pedro Barbosa Magalhães — MEGSI-ESI

Introdução

O Tópico 2 do projeto exige que o grupo domine vários conhecimentos essenciais relacionados ao Controlo de Acesso. Isto inclui compreender o que é controlo de acesso, entender o papel das políticas de acesso, explorar o funcionamento da política MAC (Controle de Acesso Mandatório), entre outros aspectos fundamentais. Além disso, é necessário criar uma rede de etiquetas de segurança (lattice) de acordo com a estrutura estabelecida pelos docentes e desenvolver sobre um possível processo de implementação automática deste modelo.

Conceitos [3) e 4)]

O controlo de acesso é um componente fundamental da segurança da informação, responsável por gerir e regular o acesso a recursos de computação, sistemas e dados. O controlo de acesso assegura que apenas utilizadores autorizados tenham permissão para aceder a recursos específicos, enquanto os não autorizados são impedidos de fazê-lo.

De destacar que o controlo de acesso inclui autenticação, autorização e *accounting/auditing*. Assim, no processo de controlo de acesso, a autenticação é a primeira linha de defesa, envolvendo a verificação da identidade do utilizador por meio de, por exemplo, credenciais como nome de utilizador e senhas. Após a autenticação, vem a autorização, na qual o sistema determina se o utilizador autenticado tem permissão para aceder a um determinado recurso e quais operações pode realizar no mesmo. Estas permissões são concedidas ou negadas com base nas políticas de segurança predefinidas, levando em conta os direitos atribuídos ao utilizador e as restrições impostas pelo sistema. Além disto, o *accounting/auditing* desempenha um papel crucial no controlo de acesso, envolvendo o seguimento e registo de atividades de acesso a recursos. Isso inclui quem acedeu a quais recursos, quando os acessou e quais ações foram realizadas. Com isto em mente, o *accounting/auditing* é essencial para monitorar a conformidade com políticas de segurança, detectar atividades suspeitas e investigar incidentes de segurança.

Em relação às políticas de seguranças, estas são frequentemente utilizadas para implementar controlo de acesso e em sistemas de computador. Apresenta-se de seguida as principais políticas:

- No modelo de Controlo de Acesso Obrigatório (*Mandatory Access Control* - MAC), o acesso é determinado por uma política de segurança imposta pelo sistema (atendendo à forma como foi configurado pelo seu administrador), com permissões baseadas em níveis de segurança e categorias de utilizadores.
- No modelo de Controlo de Acesso Discricionário (*Discretionary Access Control* - DAC), o proprietário do "objeto\recurso" tem controlo total sobre quem pode aceder ao mesmo e quais permissões são concedidas a terceiros sobre o recurso ().
- Controlo de Acesso Baseado em Função (*Role-Based Access Control* - RBAC), de igual forma ao MAC, a política de acesso é definida pelo sistema. Mas em vez de terem permissões associadas aos níveis de segurança do utilizador, as permissões estão associadas às funções do utilizador no sistema. Ou seja, por exemplo em um contexto organizacional, esta política

determina o acesso ao “objeto\recurso” com base nas funções\categoria que os utilizadores desempenham dentro da organização, com permissões atribuídas a papeis específicos.

- Por fim, temos o Controlo de Acesso Baseado em Atributo (*Attribute-Based Access Control* - ABAC) onde define o acesso a “objetos\recursos” com base em atributos específicos do utilizador, recursos e ambiente, combinando informações como identidade do utilizador, localização e tempo. Exemplo: Permitir que pessoas de um determinado departamento tenham acesso a determinados documentos apenas em determinado horário. [5)]

É Importante ainda referir o modelo de Bell-LaPadula (BLP), também conhecido como modelo multinível. Este caracteriza-se por se basear numa arquitetura MAC enquanto oferece suporte para DAC. Este modelo possui um conjunto de regras que o distinguem de outros como [6)]:

- “*No read up*”: os utilizadores não podem aceder a informação/ficheiros de níveis superiores a aquele a que foi atribuído
- “*No write down*”: os utilizadores não podem escrever informação/ficheiros de níveis superiores a aquele a que foi atribuído. Assumindo a *“star property*” se o label do objeto dominar sobre o do utilizador, apenas lhe é permitido realizar operações de *append* enquanto a escrita limita-se a objetos e a utilizadores com exatamente o mesmo label
- Princípio “*Need-to-Know”*: só se deve ser dado acesso à informação estritamente necessária para cada utilizador realizar as suas funções

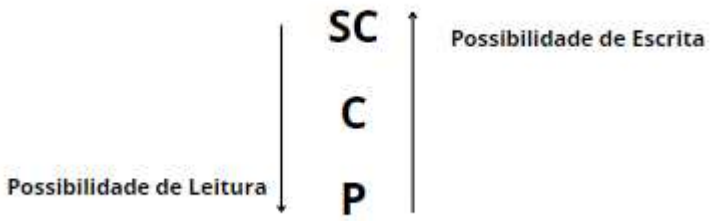

Figura 1 - Regras de BLP

Nota:
O modelo de BLP garante confidencialidade mas não integridade, pois permite escrita em objetos de nível superior ao do utilizador. Apesar de não os conseguir acessar, tem a capacidade de escrever. Isto é denominado por “escrita cega”.

Pergunta 1.

Estrutura fornecida pelos docentes:

Assumindo um contexto universitário, é necessário construir a rede de rótulos de segurança para os níveis de segurança **P** (público), **C** (confidencial) e **SC** (estritamente confidencial) e categorias **AS** (Serviços Académicos) e **ScS** (Serviços Científicos).

i) as propriedades fundamentais do modelo BLP
ii) os professores são classificados no nível (rótulo) (C, {AS, ScS}), enquanto os alunos são classificados no nível (C, {AS})
iii) a implementação habitual do modelo de Controlo de Acessos (multinível) em sistemas informáticos
Em apêndice encontra-se uma pequena explicação sobre as ligações entre rótulos.
Ou seja, os níveis de segurança seriam representados, pelo nível de restrição, da seguinte forma :

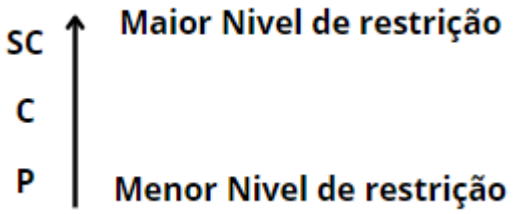

Figura 2 -Níveis de restrição

Tendo em mente a estrutura proposta o grupo concebeu a seguinte árvore de Lattice [7) e 8)] :

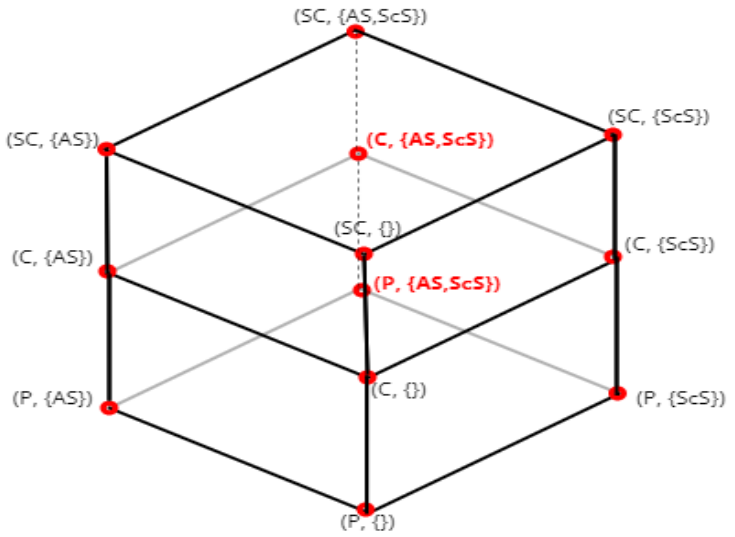

Figura 3 -Árvore de Lattice

Risco de fraude por parte de alunos

Partindo do exemplo prático disponibilizado pela equipa docente e da arquitetura desenvolvida pelo grupo foram identificados alguns elementos relevantes para a questão sobre a possibilidade de fraude tais como:

- O aluno e o docente encontram-se com o mesmo nível de *clearance* (nível **C** - confidencial)
- Ambos podem ler qualquer elemento que seja tomado como público atendendo às suas respectivas características (Público ≤ Confidencial)
- Professor = (C, {AS, ScS}) e Aluno = (C, {AS}) em formato de Lx = (Sx, Cx) [1) e 2)]
    - $S_{Professor} = S_{Aluno}$

- $C_{Professor} \supseteq C_{Aluno}$
- Logo $L_{Professor} \geq L_{Aluno}$ O *label* do Professor é mais restrito que o do Aluno

	Professor S(C, {AS, ScS})	**Aluno S(C, {AS})**

O(C, {})	• L(S) > L(O)
	  ○ Ler
		• L(S) > L(O)
		  ○ Ler

O(C, {AS})	• L(S) > L(O)
	  ○ Ler
		• L(S) = L(O)
		  ○ Ler
		  ○ Escrever

O(C, {ScS})	• L(S) > L(O) ○ Ler	• L(S) ≠ L(O)

O(C, {AS, ScS})	• L(S) = L(O)
	  ○ Ler
	  ○ Escrever
		• L(S) < L(O)
		  ○ Anexar

Tabela 1 - Permissões para o nível C (Confidencial)

Legenda:

- O: Objeto
- S: Sujeito/Utilizador

Assumindo um cenário onde não está presente nenhum controlo de acesso do tipo DAC nem uma política de "*Need-to-Know"* na arquitetura descrita previamente apresentada. Toma-se a existência de um ficheiro do professor com os resultados de todas as avaliações realizadas pela turma (o label dessa mesma pasta corresponde a **O(C, {AS, ScS})**).

Observa-se que ambos os elementos possuem o mesmo nível $S_{Ficheiro} = S_{Aluno}$, porém o conjunto de características do ficheiro contém as características do aluno $C_{Ficheiro} \supseteq C_{Aluno}$ o que leva a que seja tomado **L(Aluno) < L(Ficheiro)**, desta forma verifica-se que o aluno possui um label menos restrito quando comparado ao do ficheiro do docente o que, segundo os princípios base de Bell-LaPadula, permite que o Aluno anexe/acrescente informação/dados a este mesmo ficheiro, embora de forma "cega" pode causar problemas para o docente e funcionar a favor do próprio aluno indevidamente, uma vez que esta arquitetura limita a escrita em níveis inferiores, mas não limita a escrita ou anexo em níveis superiores.

Para que este cenário não se realize desta ou de formas semelhantes podem ser adotados outros controlos como o DAC (*Discretionary Access Control*) de forma a complementar a base do modelo de Bell-LaPadula. Ao criar um documento, o docente pode limitar quaisquer interações que os restantes utilizadores têm para com o seu ficheiro evitando que outros utilizadores mal intencionados, que se encontrem no mesmo nível de confidencialidade que este, possam alterar o seu documento.

Pergunta 2. 9) *e* 10)

Implementação teórica do SELinux

Uma das tecnologias mais conhecidas para implementação de controlos de acesso do tipo MLS (Multi Level Security) trata-se do SELinux.

SELinux, também conhecido como *Security Enhanced* Linux, é um módulo de segurança para o kernel de Linux que oferece ferramentas para o controlo de acessos. Desenvolvido pela NSA (*National Security Agency*), este módulo nasce a partir da necessidade crescente por garantir que a informação em sistemas Linux e após a transição de sistemas POSIX para LINUX apenas era acedida por quem possuísse clearance para tal, evitando que indivíduos não autorizados possam ler informação que não lhes era permitida ("*No read up*") nem que houvesse fugas de informação de níveis superiores para utilizadores determinada *clearance* ("*No write down"*) . Para conseguir tal, este módulo oferece funcionalidades para implementar uma arquitetura de MLS onde MAC e DAC podiam coexistir de forma a garantir o fluxo pretendido de informação dentro do sistema.

Atualmente o SELinux pode ser instalado em determinadas distribuições de Linux, contudo existem algumas que já possuem este módulo por default como Fedora, CentOS e RHEL (RedHat Distros).

Para poder implantar uma arquitetura próxima àquela apresentada neste trabalho de forma automática recomenda-se uma das distribuições de Linux previamente referidas uma vez que não requer instalação do módulo nem cria a possibilidade de conflitos entre outros módulos de segurança como o AppArmor (default em distros baseadas em Ubuntu entre outras).
Antes de criar os respectivos níveis bem como as suas categorias, recomenda-se verificar o estado do SELinux de forma a ver em que modo este se encontra (enabled, permissive ou disabled), para tal pode ser utilizado o comando getenforce. Para criação de novas políticas de acesso recomenda-se que o SELinux se encontre em modo *permissive* (não restringe o acesso, mas guarda logs), para tal recorremos ao comando setenforce. Após o sistema se encontrar em modo *permissive* devem ser criados os respectivos níveis de segurança (**P**, **C** e **SC**) e as categorias para cada uma das layers (**AS** e **ScS**) utilizando as ferramentas/comandos seinfo e semanage respetivamente.
Com os níveis e as categorias criados recorre-se ao comando semanage para associar os respectivos labels aos professores e alunos e, para finalizar, deve ser alterado o modo do SELinux para *enable* através do comando setenforce.

A metodologia descrita reflete uma um conjunto de instruções possível para conseguir reproduzir a arquitetura desenvolvida neste trabalho de forma exata o que exige ao administrador do sistema algum trabalho para customizar e desenvolver as políticas que pretende, contudo se o objetivo passa por simplesmente por implementar uma arquitetura de BLP de forma automática recomenda-se apenas a utilização de distribuições Linux que já possuam o módulo de kernel SELinux uma vez que não requer a instalação de qualquer novo módulo fornecendo uma solução "*straight out of the box*" para uma arquitetura BLP.

Nota:
Também deve ser verificada a referência para objetos sem nenhuma característica ({})

Implementação prática do modelo BLP [11)]

Neste capítulo será apresentada uma implementação possível e simplificada do modelo BLP desenvolvida em Ubuntu 22.04 a executar em ambiente de Windows WSL.

Esta implementação parte de dois sujeitos descritos no enunciado deste exercício: professores e alunos. Desta forma foram criados dois grupos para cada um destes tipos de sujeito com recurso ao comando groupadd como ilustra a seguinte figura:

```
~$ sudo groupadd professores
~$ sudo groupadd alunos
```

Figura 4 - Criação dos dois grupos

Após a criação de cada um dos grupos procedemos à verificação da criação dos mesmos.

```
dias@DESKTOP-      :~$ getent group | grep -E 'professores'
professores:x:1001:
dias@DESKTOP-      :~$ getent group | grep -E 'alunos'
alunos:x:1002:
```

Figura 5 - Verificação da criação dos grupos

Em seguida, com recurso à família de comandos useradd, foram adicionados dois novos utilizadores com um intuito de conseguir testar as configurações numa fase mais tardia da implementação. Após a criação de um utilizador denominado de prof1, que representa um professor que utiliza este sistema, e um outro utilizador (aluno1) que parte da mesma linha de raciocínio foi verificada a sua respetiva criação através do comando awk -F':' '{ print $1}' /etc/passwd.

```
~$ sudo useradd -g professores prof1
~$ sudo useradd -g alunos aluno1
```

Figura 6 - Criação dos users para o professor e para o aluno

```
dias
prof1
aluno1
dias@DESKTOP-      :~$
```

Figura 7 - Verificação da criação dos users

Baseando-nos nas boas práticas de criação de novos utilizadores e de forma a introduzir um outro tipo de controlo de acessos à sessão de cada um dos utilizadores criados foram definidas palavras passes para cada um com recurso aos comandos sudo passwd prof1 e sudo passwd aluno1.

Para terminar a configuração referente aos utilizadores e aos respectivos grupos, associamos cada um dos utilizadores aos grupos a que pertencem.

```
~$ sudo usermod -aG professores prof1
~$ sudo usermod -aG alunos aluno1
```

Figura 8 - Associação dos *users* aos respectivos grupos

Da mesma forma que foram configurados grupos e utilizadores partimos para a criação e configuração das diretorias destinadas a cada utilizador. Em primeiro lugar foram criadas as diretorias que compõem este modelo.

```
~$ sudo mkdir acessos
~$ sudo mkdir acessos/dirProfs
~$ sudo mkdir acessos/dirAlunos
```

Figura 9 - Criação de diretorias

Foram executados os comandos `ls -l` e `ls acessos/ -l` de forma a verificar a criação das diretorias em questão. Após a criação das mesmas foram retiradas todas as permissões associadas às diretorias pertencentes à principal denominada de "acessos" através dos comandos `sudo chmod 000 acessos/dirProfs/` e `sudo chmod 000 acessos/dirAlunos/`.

Uma vez que, por default, o Ubuntu 20.04 em WSL não possui o módulo acl instalado (*Access Control List*) foi necessário adquirir o módulo de forma a poder atribuir permissões a cada um dos grupos previamente criados. De forma a conseguir instalar este módulo foi utilizado o seguinte comando: `sudo apt-get install acl`. Após a instalação dos pacotes em questão foram cedidas permissões a cada um dos grupos para cada uma das diretorias (dirProfs e dirAlunos). A partir do enunciado e das regras base da arquitetura do modelo BLP foram definidas as seguintes permissões:

```
~$ sudo setfacl -m g:professores:rwx acessos/dirProfs/
~$ sudo setfacl -m g:alunos:-wx acessos/dirProfs/
~$ sudo setfacl -m g:professores:r-x acessos/dirAlunos/
~$ sudo setfacl -m g:alunos:rwx acessos/dirAlunos/
```

Figura 10 - Definição das permissões de acesso para as diretorias

As permissões foram atribuídas aos grupos e não aos utilizadores em particular de forma a facilitar a escalabilidade do sistema onde apenas se inclui um novo user a um grupo em vez de configurar a lista de acesso para cada novo utilizador.

```
dias@DESKTOP-H4V6K8U:~$ getfacl acessos/dirProfs/
# file: acessos/dirProfs/
# owner: root
# group: root
user::---
group::---
group:professores:rwx
group:alunos:-wx
mask::rwx
other::---

dias@DESKTOP-H4V6K8U:~$ getfacl acessos/dirAlunos/
# file: acessos/dirAlunos/
# owner: root
# group: root
user::---
group::---
group:professores:r-x
group:alunos:rwx
mask::rwx
other::---
```

Figura 11 - Verificação das permissões dos grupos para cada diretoria

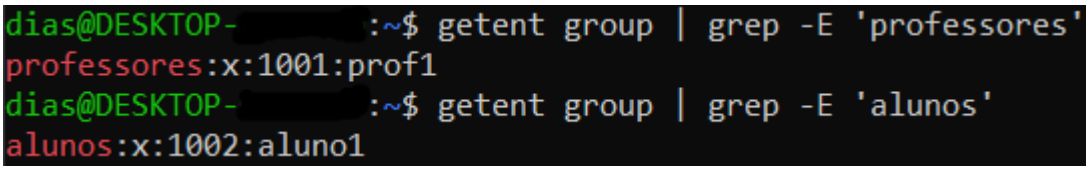

Figura 12 - Verificação dos *users* para cada grupo

Teste da arquitetura implementada

De forma a validar a eficácia da arquitetura desenvolvida nesta fase partimos para a demonstração prática da arquitetura. A partir desta tabela serão definidas as condições de acesso, verificado se o comportamento dos utilizadores corresponde àquele permitido pela arquitetura concebida pelo grupo:

Diretorias (Users) / (Grupo)	dirProfs	dirAlunos
prof1 / professores	RWX	R-X
aluno1 / alunos	-WX	RWX

Tabela 2 - Matriz de acessos

Primeiramente foi verificado se o prof1 e o aluno1 possuem todas as permissões dentro das diretorias destinadas a cada um. Para tal será feito o login a ambos os *users* e cada um escreverá um ficheiro para a sua respectiva diretoria, de seguida cada um irá ler esse mesmo ficheiro.

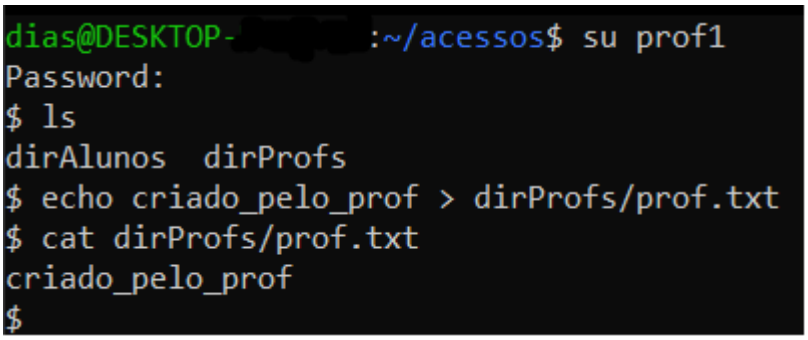

Figura 13 - Escrita e leitura do user prof1 na sua diretoria

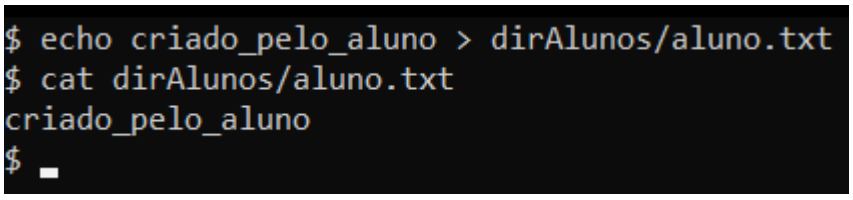

Figura 14 - Escrita e leitura do user aluno1 na sua diretoria

Após se verificar que ambos os utilizadores conseguem ler e escrever ficheiros na respetiva diretoria foram testadas as permissões do utilizador prof1. Como ilustra a Figura 15, observamos que

o professor tem permissões para ler o conteúdo da diretoria “dirAlunos”, porém é incapaz de escrever na mesma.

```
dias@DESKTOP-      :~/acessos$ su prof1
Password:
$ cat dirAlunos/aluno.txt
criado_pelo_aluno
$ echo prof_w_dirAluno > dirAlunos/aluno.txt
sh: 2: cannot create dirAlunos/aluno.txt: Permission denied
$ exit
```

Figura 15 - Verificação de acessos do prof1 na diretoria do aluno

Seguindo a validação das permissões do professor, testámos as permissões associadas ao aluno. Neste teste foi utilizado o editor de texto Nano para ler o conteúdo do ficheiro criado pelo docente (user prof1) onde se observa que o aluno1 não consegue ler esse mesmo ficheiro.

```
dias@DESKTOP-      :~/acessos$ su aluno1
Password:
$ ls
dirAlunos  dirProfs
$ nano dirProfs/prof.txt
Unable to create directory /home/aluno1/.local/share/nano/: No such file or directory
It is required for saving/loading search history or cursor positions.
```

Figura 16 - Verificação de acessos do aluno1 na diretoria do professor

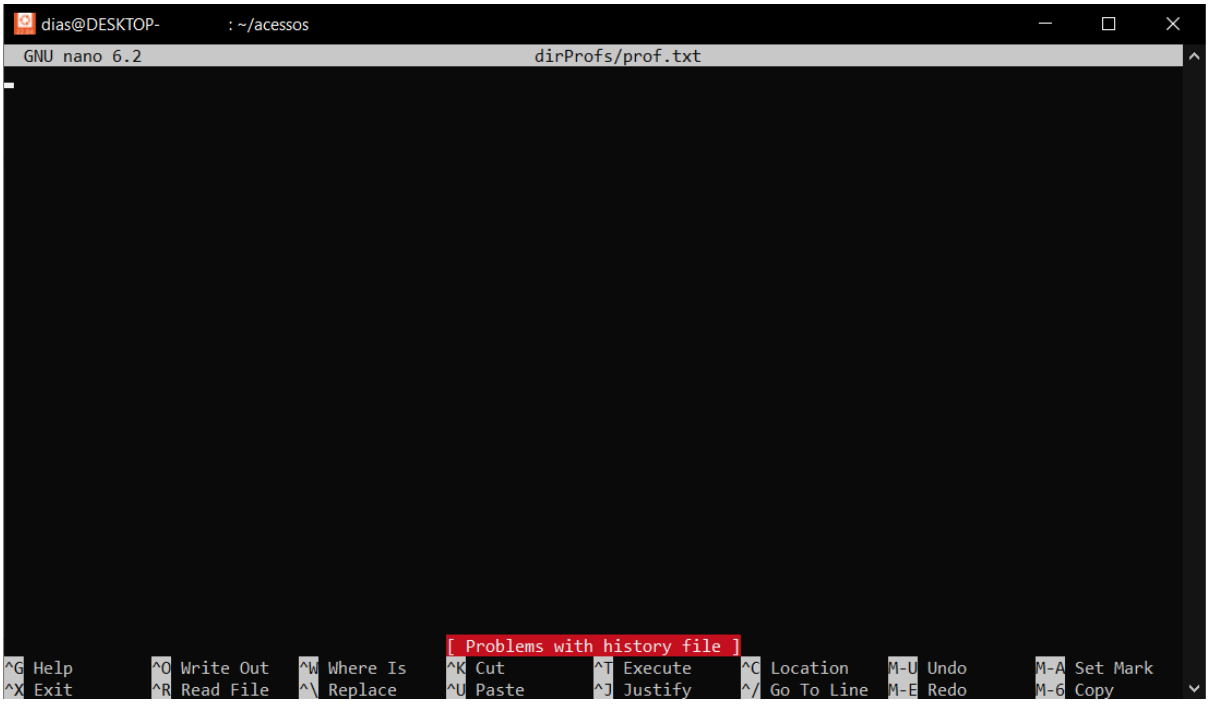

Figura 17 - Tentativa de leitura do user aluno1 do ficheiro na diretoria do professor

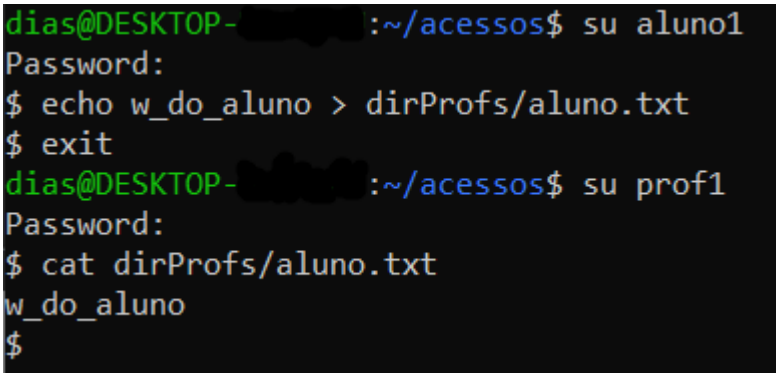

Figura 18 - Verificação de escrita do user aluno1 na diretoria do professor

Entretanto verificamos que o aluno consegue efetivamente escrever na diretoria destinada ao professor. Assim fica provado que a arquitetura desenvolvida neste capítulo vai ao encontro das características que definem a arquitetura de Bell-LaPadula o que permite classificar o produto do trabalho desenvolvido pelo grupo como uma implementação eficaz desta arquitetura.

Conclusão

Este projeto teve como objetivo fundamental o aprofundamento do conhecimento e modulação sobre controlo de acesso em sistemas de informáticos a partir do modelo Bell-LaPadula (BLP). Este desempenha um papel crucial na proteção e segurança de informações, especialmente em ambientes militares onde a confidencialidade é um dos principais pilares. O relatório elaborado pelo grupo consiste em duas partes distintas: a criação da arquitetura teórica em contexto real e a investigação da implementação automatizada do modelo previamente desenvolvido. A primeira parte abordou os aspectos mais teóricos do modelo, assim como os conceitos fundamentais no controlo de acesso, complementada por uma representação esquemática da arquitetura BLP. De seguida, foi discutida uma possível implementação prática da arquitetura BLP, utilizando o módulo de kernel SELinux e apresentando uma implementação simplificada em ambiente Ubuntu WSL.

Assim, este trabalho não apenas ampliou a compreensão teórica do grupo nesta matéria, como também proporcionou uma visão prática e aplicável das estratégias de controlo de acesso destacando sua importância e relevância em ambientes reais.

Bibliografia

1) https://www.cs.cornell.edu/courses/cs5430/2011sp/NL.accessControl.html
2) https://www.cs.unc.edu/~dewan/242/f96/notes/prot/node13.html#SECTION000112000000000000000
3) https://e-learning.dsi.uminho.pt/pluginfile.php/147637/mod_resource/content/4/Topico-AC/ControloAcessos-En.pdf
4) https://www.techtarget.com/searchsecurity/definition/authentication-authorization-and-accounting
5) https://www.youtube.com/watch?v=5lunZm2gbmk&t=183s&ab_channel=NextLabs
6) https://en.wikipedia.org/wiki/Bell%E2%80%93LaPadula_model
7) https://www.cs.purdue.edu/homes/ninghui/courses/Spring18/handouts/05_blp.pdf
8) https://profsandhu.com/cs6393_s12/lbac-blp-biba.pdf
9) https://www.youtube.com/watch?v=HhydNtaLEK0
10) https://www.redhat.com/sysadmin/semanage-keep-selinux-enforcing
11) https://www.redhat.com/sysadmin/linux-access-control-lists

Apêndice

Nota - O que representa as ligações entre rótulos:
As ligações entre os rótulos na rede de árvore de lattice indicam as permissões de acesso entre esses níveis de segurança. Cada nó representa um rótulo de segurança e as ligações entre nós indicam as relações de controlo de acesso permitidas. Essas relações geralmente incluem:

- Descendência: Um nó mais alto pode aceder informações em um nó mais baixo. Isso representa uma permissão de leitura de informações menos confidenciais por parte de entidades com acesso a informações mais confidenciais.
- Ascendência: Um nó mais baixo não pode aceder informações em um nó mais alto. Isso reflete a política de "necessidade de saber", onde informações de um nível mais alto não são acessíveis a entidades em níveis mais baixos, a menos que tenham a permissão explícita.

Comandos utilizados:

>Criar grupos sudo groupadd professores sudo groupadd alunos **>Verificar criacao dos grupos** getent group | grep -E 'professores' getent group | grep -E 'alunos' **>Criar novos users para os respetivos grupos** sudo useradd -g professores prof1 sudo useradd -g alunos aluno1	**>Verificar diretorias** ls -l ls acessos/ -l **>Retirar todas as permissoes as diretorias** sudo chmod 000 acessos/dirProfs/ sudo chmod 000 acessos/dirAlunos/ ls acessos/ -l (verificacao) **>Instalar ACL** sudo apt-get install acl

>Listar users awk -F':' '{ print $1}' /etc/passwd **>Definir passwords para cada user** sudo passwd prof1 (>prof1<) sudo passwd aluno1 (>aluno1<) **>Associar users a grupos** sudo usermod -aG professores prof1 sudo usermod -aG alunos aluno1 **>Criar diretorias para cada grupos** sudo mkdir acessos sudo mkdir acessos/dirProfs sudo mkdir acessos/dirAlunos	**>Atribuir permissoes ao grupo para cada dir** sudo setfacl -m g:professores:rwx acessos/dirProfs/ sudo setfacl -m g:alunos:-wx acessos/dirProfs/ sudo setfacl -m g:professores:r-x acessos/dirAlunos/ sudo setfacl -m g:alunos:rwx acessos/dirAlunos/ **>verificacao das permissoes de acesso** getfacl acessos/dirProfs/ getfacl acessos/dirAlunos/