Universidade do Minho
Escola de Engenharia

Mestrado em Engenharia Informática

Ano letivo 2024/2025

Tecnologias de Segurança

Trabalho Prático #3

Grupo 02

João Rodrigues - pg57880
Rúben Silva - pg57900
Miguel Guimarães - pg55986

Março 2025

Índice

Figuras

1 Introdução

O **controlo de acesso** é um dos pilares fundamentais da segurança informática, estabelecendo um conjunto de regras e mecanismos que garantem que só utilizadores autorizados conseguem aceder aos recursos.

Ao longo dos anos foram desenvolvidos vários modelos de autorização, sendo que em **1975** foi desenvolvido um dos modelos que se destacam como uma referência na proteção da confidencialidade, o **modelo Bell-LaPadula (BLP)**.

1.1 Introdução ao modelo Bell-LaPadula

O **Bell-LaPadula (BLP)** foi desenvolvido com objetivo de impor **controlo de acesso** em sistemas de instituições Militares e em sistemas do Governo, assentando numa **estrutura hierárquica de níveis de classificação** que visa garantir que informação sensível não seja divulgada a utilizadores que não possuam a autorização adequada.

Um dos conceitos centrais do modelo é a **"não-leitura-acima"** e a **"não-escrita-abaixo"**, isto porque os utilizadores não podem ler dados de um nível de classificação mais altos ou escrever dados em um nível de classificação mais baixo. O objetivo é impedir a **fuga de informações confidenciais** dos níveis mais altos para os mais baixos.

O modelo possui a **propriedade de segurança discricionária** que indica que no mesmo nível de segurança a autorização é definida por uma matriz de acesso, ou seja, os donos dos recursos definem as permissões dos mesmos para outros utilizadores no mesmo nível de segurança.

O **principio de tranquilidade** define quando é que os rótulos de segurança podem ser modificados, existindo dois tipos diferentes de tranquilidade. Existe a **tranquilidade forte** em que os rótulos não podem ser modificados enquanto o sistema se encontra em execução e a **tranquilidade fraca** em que os rótulos não podem ser modificados de forma a que uma das propriedade de segurança seja violada.

Embora o modelo Bell-LaPadula (BLP) aparente ser um modelo completo este **possui varias limitações** que impedem que este seja adotado de forma mais ampla. Algumas das suas limitações são por exemplo a natureza estática dos níveis de classificação, a ausência de mecanismos para uma gestão dinâmica dos utilizadores e a falta de flexibilidade em cenários mais complexos.

1.2 Objetivo do trabalho

Este trabalho prático tem como objectivo explorar e implementar uma adaptação ao modelo Bell--LaPadula que permita superar algumas das suas limitações, tornando-o mais adequado para aplicações práticas.

Pretende-se desenvolver uma solução que mantenha os princípios fundamentais de proteção da confidencialidade, introduzindo simultaneamente soluções para os problemas descritos anteriormente, fornecendo a flexibilidade necessária para uma adoção moderna e realista do modelo.

2 Estrutura geral

2.1 Visão Geral da Arquitetura

A aplicação desenvolvida segue uma arquitetura cliente-servidor de três camadas, implementando o modelo de controlo de acesso Bell-LaPadula. O sistema está estruturado de forma modular, separando claramente as responsabilidades entre apresentação, lógica de negócio e persistência de dados.

2.1.1 Cliente

A aplicação cliente permite que os utilizadores interajam com o sistema através de uma aplicação de linha de comandos, sendo que esta aplicação comunica com o servidor recorrendo **a pedidos HTTPS**, o que garante segurança à comunicação entre o cliente e o servidor. À medida que o utilizador comunica com o servidor são apresentados os vários resultados consoante a ação realizada.

A aplicação cliente recorre também a **tokens JWT (JSON Web Tokens)** para gerir o estado da autenticação localmente.

2.1.2 Servidor

A aplicação servidor foi implementada recorrendo ao **FastAPI**.

O servidor é responsável por criar **endpoints REST** para todas as operações do sistema, sendo que os utilizadores comunicam com o servidor enviando pedidos HTTPS para os respetivos endpoints, que são traduzidos nas operações correspondentes na logica de negocio.

Todos os dados recebidos dos utilizadores são validados através de modelos Pydantic, que rejeitam entradas malformadas e convertem tipos de forma segura, prevenindo injeções e outros ataques.

São utilizados **tokens JWT (JSON Web Tokens)** para gerir o estado da autenticação dos vários utilizadores.

2.1.3 Implementação do Bell-LaPadula (BLP)

A classe BLP representa o núcleo do sistema, implementando todas as regras do modelo.

As regras implementadas incluem o controlo de acesso multi-dimensional, como por exemplo as etiquetas air, sea e land utilizadas em representações de sistemas militares.

Também foi implementada a validação de permissões baseada nas regras **Propriedade de segurança simples** (“No Read Up”), na **Propriedade de segurança estrela** * (“No Write Down”), e a **propriedade discricionária de segurança.**

Foi implementada uma Gestão de utilizadores e objetos seguros com utilização de um Sistema de auditoria e logs detalhados.

Para a alteração de labels segurança foram utilizadas Operações de expurgo.

2.2 Vantagens da Arquitetura

- **Separação de Responsabilidades:** cada componente tem responsabilidades bem definidas, facilitando manutenção e evolução do sistema.
- **Escalabilidade:** a arquitetura permite expansão independente de cada camada conforme necessário.
- **Segurança:** múltiplas camadas de segurança desde a comunicação até ao armazenamento de dados.
- **Auditoria:** o sistema de logs permite rastreabilidade total das operações.
- **Flexibilidade:** a API REST (implementado com FastApi) permite integração com diferentes tipos de clientes no futuro.

3 Servidor

3.1 Visão Geral do Servidor

O servidor implementa uma API REST utilizando FastAPI, a aplicação do servidor funciona como intermediário entre os clientes e a lógica de negócio BLP, oferecendo endpoints seguros para todas as operações do sistema.

3.2 Arquitetura e Dependências

O servidor utiliza as seguintes principais tecnologias:
- **FastAPI:** Framework web para construção de APIs com validação automática de dados
- **Pydantic:** Para definição e validação de modelos de dados, rejeitando automaticamente inputs maliciosos e prevenindo ataques de injeção e erros de tipo.
- **JWT (PyJWT):** Para autenticação baseada em tokens.
- **Uvicorn:** Servidor ASGI para executar a aplicação
- **SSL/TLS:** Comunicações seguras através de certificados personalizados

3.3 Modelos de Dados

O servidor define vários modelos Pydantic para estruturar e validar os dados recebidos:

- **RegisterRequest:** Contém informações para registo de utilizadores (nome, palavra-passe, níveis de segurança e estatuto de confiança)
- **User:** Modelo básico para autenticação com credenciais de utilizador
- **Object:** Define a estrutura de objetos do sistema (nome, conteúdo e classificações de segurança)
- **ExpurgateRequest:** Especifica parâmetros para operações de expurgo (objeto original, novo nome e conteúdo filtrado)
- **ClearanceRequest:** Para alteração de níveis de segurança de utilizadores

3.4 Sistema de Autenticação

3.4.1 Geração de Tokens JWT

O endpoint de login (/login) implementa autenticação baseada em tokens:
- Valida as credenciais através da classe BLP
- Gera tokens JWT com expiração de 30 minutos
- Inclui o nome de utilizador no payload do token
- Utiliza chave secreta para assinatura (definida para fins de teste)

3.4.2 Validação de Tokens

A função get_current_user() funciona como dependência para endpoints protegidos:
- Extrai o token do cabeçalho Authorization
- Valida o esquema Bearer
- Descodifica e verifica a assinatura JWT
- Devolve o nome de utilizador para uso nos endpoints
- Gera exceções HTTP 401 para tokens inválidos ou em falta

3.5 Endpoints de API

3.5.1 Gestão de Utilizadores

POST /register: Permite registo de novos utilizadores, delegando a validação na classe BLP e retornando erro 400 se o utilizador já existir.

POST /login: Autentica utilizadores e retorna tokens JWT válidos por 30 minutos, com erro 401 para credenciais inválidas.

3.5.2 Operações com Objetos

POST /create_object: Endpoint protegido que permite criar objetos através da validação BLP, retornando erro 403 se não houver permissões.
GET /read_object/{obj_name}: Permite leitura de objetos específicos, validando permissões através do modelo BLP e retornando o conteúdo ou erro 403.
POST /expurgate_object: Funcionalidade para expurgo de objetos, criando versões filtradas com classificações reduzidas.

3.5.3 Funcionalidades de Listagem

GET /list_readable_objects: Retorna lista de objetos que o utilizador autenticado pode ler, baseado nas suas autorizações.
GET /list_expurgatable_objects: Fornece lista de objetos que o utilizador pode submeter a expurgo.

3.5.4 Administração

POST /set_clearance: Permite alterar níveis de segurança de outros utilizadores, restrito a utilizadores com privilégios administrativos.
GET /get_logs: Endpoint para consulta de logs do sistema, disponível apenas para utilizadores com nível de segurança máximo (5).

3.6 Tratamento de Erros

O servidor implementa tratamento estruturado de erros utilizando exceções HTTP:
- 400 Bad Request: Para dados inválidos ou utilizadores já existentes
- 401 Unauthorized: Para problemas de autenticação (credenciais inválidas, tokens em falta ou expirados)
- 403 Forbidden: Para violações de controlo de acesso (permissões insuficientes)

3.7 Integração com a Lógica BLP

O servidor funciona como camada de apresentação, delegando toda a lógica de controlo de acesso a uma instância da classe BLP:
- Todas as operações são validadas pela lógica BLP
- O servidor traduz as respostas da classe BLP em códigos HTTP apropriados
- Mantém separação clara entre apresentação e lógica de negócio

3.8 Configuração e Execução segura

O servidor está configurado para execução segura:
- HTTPS obrigatório através de certificados
- Porta 8000 para comunicações locais
- Certificados: Utiliza “cert.pem” e “key.pem”

O servidor representa uma implementação robusta e segura de uma API REST para controlo de acesso multi-nível, fornecendo uma interface web padronizada para interação com o modelo Bell-LaPadula.

4 Cliente

4.1 Configuração e Estado

A aplicação cliente-side mantém várias variáveis globais para gerir o estado da sessão:
- TOKEN: Armazena o token de autenticação JWT recebido após login bem-sucedido
- BASE_URL: Define o endereço do servidor (https://localhost:8000)
- CA: "Path" para o certificado para validação das ligações HTTPS

O cliente utiliza a biblioteca requests para comunicação HTTP e mantém o estado de autenticação durante toda a sessão

4.2 Funcionalidades Disponíveis no Cliente

4.2.1 Autenticação

- Login: O utilizador introduz as suas credenciais (nome de utilizador e palavra-passe). A aplicação

cliente envia estes dados ao servidor e, se a autenticação for bem-sucedida, recebe e armazena um token JWT que será utilizado em todas as operações subsequentes.
- Registo: Permite criar uma nova conta no sistema. Durante o registo, o utilizador define não apenas

as credenciais básicas, mas também o nível de acesso e os domínios de acesso, e especifica se é um utilizador confiável. Após registo bem-sucedido, a aplicação cliente executa automaticamente o login.

4.2.2 Gestão de Objetos

- Criação de Objetos: O utilizador pode criar novos objetos especificando um nome, conteúdo textual e níveis de classificação de segurança nas três dimensões. A aplicação cliente envia esta informação ao servidor com o token de autorização.
- Leitura de Objetos: São disponibilizadas duas abordagens para leitura:
  - Leitura direta através do nome do objeto
  - Listagem de objetos disponíveis para leitura, permitindo ao utilizador selecionar de uma lista

numerada

4.2.3 Operações de Expurgo

Expurgo de Objetos: É permitido ao utilizador selecionar objetos que pode expurgar de uma lista de objetos disponíveis. Durante o processo de expurgo, o utilizador especifica:
- O objeto original a expurgar
- Um novo nome para a versão expurgada
- O conteúdo filtrado (sem informação sensível)
- Um novo nível de Segurança

4.2.4 Funcionalidades Administrativas

- Definição de Níveis de Segurança: Utilizadores com privilégios adequados podem alterar os níveis de segurança de outros utilizadores, especificando o nome de utilizador alvo e o nível a atribuir.
- Visualização de Logs: Permite consultar registos de atividade do sistema, apresentando informações como timestamp, utilizador, ação realizada e detalhes da operação.

4.3 Interface de Utilizador

A aplicação cliente apresenta um menu principal baseado em consola com as seguintes características:
- Limpeza de ecrã entre operações para melhor legibilidade
- Indicação de estado, mostrando se o utilizador está autenticado

- Menu numerado com oito opções principais
- Validação de entrada com mensagens de erro informativas
- Pausas controladas ("Press Enter to continue") para permitir ao utilizador ler as mensagens

4.4 Fluxo de Funcionamento

- Inicialização: A aplicação inicia sem autenticação
- Autenticação: O utilizador deve fazer login ou registar-se
- Operações: Com autenticação válida, as funcionalidades ficam disponíveis
- Autorização: Cada operação inclui o token JWT no cabeçalho HTTP
- Tratamento de Respostas: a aplicação processa as respostas do servidor e apresenta feedback ao utilizador

4.5 Tratamento de Erros

O cliente implementa tratamento robusto de erros:
- Exceções de Rede: Captura e apresenta erros de comunicação
- Códigos de Estado HTTP: Interpreta respostas de erro do servidor
- Validação Local: Verifica se o utilizador está autenticado antes de operações que o requerem
- Mensagens Informativas: Fornece feedback claro sobre sucessos e falhas

4.6 Segurança na Comunicação

- HTTPS Obrigatório: Todas as comunicações utilizam TLS
- Validação de Certificados: Utiliza certificado para validação
- Tokens JWT: Implementa autenticação baseada em tokens
- Headers de Autorização: Inclui tokens em todas as operações autenticadas

O cliente funciona como uma interface completa e intuitiva para interação com o sistema BLP, abstraindo a complexidade da comunicação HTTP e fornecendo uma experiência de utilizador estruturada e segura.

A classe BLP constitui uma implementação completa do modelo de segurança Bell-LaPadula, fornecendo um sistema de controlo de acesso obrigatório baseado em níveis hierárquicos de segurança e domínios de informação. Esta classe encapsula toda a lógica necessária para gerir utilizadores, objetos e as respectivas permissões de acesso, garantindo o cumprimento rigoroso das regras fundamentais do modelo BLP.

4.7 Arquitectura e Componentes Principais

A nossa implementação baseia-se numa arquitetura de dupla base de dados SQLite, separando dados operacionais de registos de auditoria. Integra um sistema de criptografia simétrica (Fernet) para proteger o conteúdo armazenado e implementa autenticação segura através de hash bcrypt. São definidos níveis de segurança de 1 a 5 e três domínios predefinidos (Air, Land, Sea), criando um ambiente controlado para demonstração e teste do modelo.

4.8 Modelo de Segurança Implementado

A classe BLP implementa as regras centrais do modelo Bell-LaPadula através de validações rigorosas em todas as operações de acesso. A regra "No Read Up" é aplicada consistentemente, impedindo que utilizadores acedam a informação classificada acima do seu nível de autorização. O controlo de escrita garante que os utilizadores só podem criar objetos com o seu nível exato de segurança, enquanto o

sistema de domínios adiciona uma camada adicional de controlo de acesso baseada em categorias de informação.

4.9 Gestão de Utilizadores e Hierarquia de Privilégios

É estabelecida uma hierarquia clara de utilizadores, distinguindo entre contas normais e utilizadores "de confiança" que podem realizar operações administrativas. Os novos utilizadores são automaticamente atribuídos com permissões mínimas, enquanto a classe permite a elevação controlada de privilégios através de utilizadores autorizados. Esta abordagem garante que as alterações de segurança são realizadas apenas por entidades apropriadas.

4.10 Controlo de Objetos e Informação

Toda a informação gerida pelo BLP é tratada como objetos classificados, cada um com o seu próprio nível de segurança e conjunto de domínios. Assim é assegurado que o acesso a estes objetos obedece estritamente às regras do modelo BLP, incluindo funcionalidades especializadas como a expurgação de documentos, que permite criar versões filtradas de informação classificada para divulgação a níveis inferiores de autorização.

4.11 Sistema de Auditoria

A implementação do BLP incorpora um sistema abrangente de auditoria que regista todas as operações realizadas no sistema. Este mecanismo de logging é essencial para ambientes que requerem conformidade regulamentar e permite rastrear todas as atividades dos utilizadores. O acesso aos registos de auditoria é restrito aos administradores de mais alto nível, mantendo a integridade do sistema de monitorização.

5 Bell-LaPadula

A classe BLP constitui uma implementação completa do modelo de segurança Bell-LaPadula, fornecendo um sistema de controlo de acesso obrigatório baseado em níveis hierárquicos de segurança e domínios de informação. Esta classe encapsula toda a lógica necessária para gerir utilizadores, objetos e as respectivas permissões de acesso, garantindo o cumprimento rigoroso das regras fundamentais do modelo BLP.

5.1 Arquitectura e Componentes Principais

A nossa implementação baseia-se numa arquitetura de dupla base de dados SQLite, separando dados operacionais de registos de auditoria. Integra um sistema de criptografia simétrica (Fernet) para proteger o conteúdo armazenado e implementa autenticação segura através de hash bcrypt. São definidos níveis de segurança de 1 a 5 e três domínios predefinidos (Air, Land, Sea), criando um ambiente controlado para demonstração e teste do modelo.

5.2 Modelo de Segurança Implementado

A classe BLP implementa as regras centrais do modelo Bell-LaPadula através de validações rigorosas em todas as operações de acesso. A regra "No Read Up" é aplicada consistentemente, impedindo que utilizadores acedam a informação classificada acima do seu nível de autorização. O controlo de escrita garante que os utilizadores só podem criar objetos com o seu nível exato de segurança, enquanto o sistema de domínios adiciona uma camada adicional de controlo de acesso baseada em categorias de informação.

5.3 Gestão de Utilizadores e Hierarquia de Privilégios

É estabelecida uma hierarquia clara de utilizadores, distinguindo entre contas normais e utilizadores "de confiança" que podem realizar operações administrativas. Os novos utilizadores são automaticamente atribuídos com permissões mínimas, enquanto a classe permite a elevação controlada de privilégios através de utilizadores autorizados. Esta abordagem garante que as alterações de segurança são realizadas apenas por entidades apropriadas.

5.4 Controlo de Objetos e Informação

Toda a informação gerida pelo BLP é tratada como objetos classificados, cada um com o seu próprio nível de segurança e conjunto de domínios. Assim é assegurado que o acesso a estes objetos obedece estritamente às regras do modelo BLP, incluindo funcionalidades especializadas como a expurgação de documentos, que permite criar versões filtradas de informação classificada para divulgação a níveis inferiores de autorização.

5.5 Sistema de Auditoria

A implementação do BLP incorpora um sistema abrangente de auditoria que regista todas as operações realizadas no sistema. Este mecanismo de logging é essencial para ambientes que requerem conformidade regulamentar e permite rastrear todas as atividades dos utilizadores. O acesso aos registos de auditoria é restrito aos administradores de mais alto nível, mantendo a integridade do sistema de monitorização.

6 Conclusão

Neste trabalho foi implementado o modelo Bell-LaPadula, sendo que foram adicionadas varias features novas que permitiram superar as suas limitações originais. Foram adicionados mecanismos de gestão dinâmica de utilizadores, domínios de informação e expurgação controlada de objetos.

A utilização de tecnologias como FastAPI, Pydantic e JWT no servidor permitiu implementar todo o sistema de forma eficiente e segura, permitindo também uma grande facilidade de manter o sistema e expandir o mesmo. A arquitetura cliente-servidor de três camadas assegurou separação clara de responsabilidades e escalabilidade.

O sistema de auditoria permite rastrear todas as operações, garantindo ainda mais proteção e segurança a todo o sistema.

Em suma, a solução proposta mantém os princípios centrais de confidencialidade e mínimo privilégio do modelo BLP, oferecendo simultaneamente a flexibilidade necessária para cenários modernos de segurança informática.