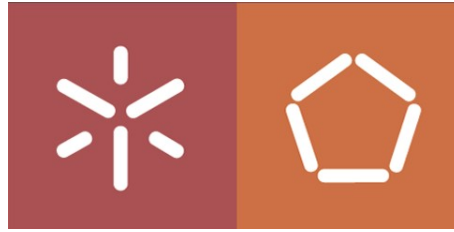


UNIVERSIDADE DO MINHO
DEPARTAMENTO DE INFORMÁTICA



MEI - MESTRADO EM ENGENHARIA INFORMÁTICA

CRİPTOGRAFIA E SEGURANÇA DA INFORMAÇÃO

Engenharia de Segurança

Grupo 13

PD2 - Projeto de Desenvolvimento 2

Hugo Marques PG47848

José Santos A84288

Nuno Mata PG44420

Junho
2023

Conteúdo

1	Introdução	2
1.1	Contextualização	2
1.2	Objetivos e Resultados Esperados	2
1.3	Estrutura do Documento	3
2	DSS Demo WebApp	4
2.1	Digital Signature Service - DSS	4
2.2	DSS Demo WebApp	4
2.3	Parâmetros das Assinaturas Digitais	5
3	Funcionalidades Desenvolvidas	7
3.1	Autenticação inicial	7
3.2	Área de Utilizador	8
3.3	Sign a document	8
3.4	Sign a digest	9
4	Métodos de Desenvolvimento de Software Seguro	11
4.1	Software Assurance Maturity Model (SAMM)	11
4.2	PIA	11
4.3	Buffer Overflow	11
4.4	Hash da senha de um utilizador	11
4.5	Vulnerabilidade de inteiros	11
4.6	Validação de input	12
4.7	Spring Security	12
5	Instalação e Utilização da Aplicação	14
5.1	Requisitos	14
5.2	Instalação da WebApp	14
5.2.1	Windows	14
6	Conclusão	15
7	Anexo I	17
7.1	Pergunta 2.1	17
8	Anexo II	19
8.1	Pergunta 1.1	19

1 Introdução

1.1 Contextualização

A crescente utilização de assinaturas eletrónicas reflete a digitalização de processos em diversos setores, desde o privado até o governamental. Sendo ferramentas valiosas que permitem a verificação da autenticidade de documentos de maneira rápida e eficiente, sem a necessidade de um processo físico. Visto isto, a União Europeia, com o regulamento *eIDAS* (*Electronic Identification And Trust Services*), implementado em 2016, fornece um quadro legal para garantir a segurança e a legalidade das transações eletrónicas. Este regulamento abrange não apenas as assinaturas eletrónicas, mas também a identificação eletrónica e os serviços de confiança, harmonizando as práticas em toda a UE. Este regulamento, *eIDAS*, facilitou também a ampla adoção das assinaturas eletrónicas ao estabelecer normas claras e transparentes para a sua utilização, o que também aumentou a confiança do público nessas tecnologias.

Este documento, realizado no âmbito da Unidade Curricular de Engenharia de Segurança do perfil de Criptografia e Segurança da Informação, tem o objetivo de descrever os desafios encontrados como também os passos realizados ao longo do desenvolvimento do Projeto de desenvolvimento 2 (PD2). Serve, assim, este documento como relatório final para este projeto de desenvolvimento, que teve o foco na utilização de ferramentas disponibilizadas no âmbito do *Digital Signature Services (DSS)*. A DSS é uma biblioteca de *software open-source* disponibilizada pela União Europeia para a criação e validação de assinaturas eletrónicas, em linha com o Regulamento *eIDAS* e standards relacionados.

Como ponto de partida para o desenvolvimento deste projeto referimos a DSS Demo Web App, que tem várias funcionalidades de assinatura e autenticação digital, às quais iremos adicionar e integrar novas características.

1.2 Objetivos e Resultados Esperados

Como descrito em cima, o objetivo principal deste projeto estará relacionado com a integração de novas funcionalidades no *DSS Demo WebApp*. Estas funcionalidades pretendem tornar a aplicação mais completa num ponto de vista de opções para assinaturas digitais. Deste modo, as funcionalidades pretendidas para serem implementadas na aplicação web são as seguintes:

- Transpor as alterações realizadas anteriormente na versão antiga da *DSS Demo WebApp* para a versão mais recente para que continue a suportar a utilização com:
 1. Cartão de Cidadão;
 2. Chave Móvel Digital;
 3. A fonte de *timestamp* do Cartão de Cidadão, de modo a não se utilizar a *dummy timestamp source* que é utilizada nas várias opções da *Demo WebApp* que utilizam *timestamp*.
- Adição de uma interface de autenticação inicial (para fazer Login com utilizador e password);
- Adição de uma área de utilizador, que pode ser acedida apenas depois da autenticação inicial, onde o utilizador poderá definir qual o número de telemóvel que utiliza para a Chave Móvel Digital, sendo que este número deve ser guardado na Base de Dados;
- Sempre que o utilizador efetue uma operação que utilize a Chave Móvel Digital, deve ser automaticamente usado o número que foi guardado na Base de Dados para o respetivo utilizador;
- Por último, o problema particular para o nosso grupo refere-se à possibilidade de assinar com chaves privadas (e respetivos certificados na hierarquia até à Entidade de Certificação raiz) em ficheiro (formato DER), nas opções de assinatura, em específico na funcionalidade de *Sign a digest*.

Com a implementação destes pontos seremos capazes de dotar a aplicação web de ter suporte para a utilização da Chave Móvel Digital. A CMD, que é um dos padrões de assinatura e autenticação digital adotadas pelo governo de Portugal, mesmo sendo uma assinatura digital tem o mesmo valor legal que uma assinatura manuscrita e é

reconhecida na União Europeia. É ainda de mencionar que, Portugal como membro da UE, para além deste padrão referido, segue as normas de *AdES (Advanced Electronic Signatures)* referidas pela *ETSI (European Telecommunications Standards Institute)* e aceita ainda outros padrões de assinatura e autenticação digital.

Em seguida, para além das funcionalidades descritas, e de forma a melhorar a segurança do software também será necessário assegurar o uso de metodologias de software seguro. Aqui, destacamos o processo de desenvolvimento de software criado pela *Microsoft*, o *Microsoft Security Development Lifecycle (SDL)*, que consiste numa série de práticas recomendadas e ferramentas que se concentram em segurança e privacidade, e são incorporadas em todas as fases do processo de desenvolvimento do software. Neste ponto vai ainda ser tido em consideração o modelo de maturidade OWASP Software Assurance Maturity Model (SAMM) desenvolvido pela Open Web Application Security Project (OWASP) onde serão tidos em conta os standards de verificação de segurança de aplicações.

Por fim, de modo a demonstrar a conformidade com o RGPD (Regulamento Geral de Proteção de Dados) será utilizada a ferramenta de análise de impacto da proteção de dados *Privacy Impact Assessment (PIA)*, que será capaz de gerir riscos de privacidade e garantir a manipulação responsável de dados pessoais.

1.3 Estrutura do Documento

Com o desenvolvimento do relatório vão ser explicados os passos previamente descritos, também como a explicação das técnicas e métodos usados que auxiliaram o desenvolvimento da aplicação. Desta forma, o documento estará dividido em 5 capítulos:

O primeiro capítulo serve para fazer uma contextualização do problema e também uma descrição dos objetivos que se esperam ser alcançados no final do projeto.

No segundo capítulo vai ser descrita a aplicação que serve de base para a integração dos objetivos descritos anteriormente, a *DSS Demo WebApp*. Aqui serão descritas as funcionalidades mais importantes que podem ser encontradas na versão "base" da aplicação.

De seguida, o terceiro capítulo concentra-se na explicação das funcionalidades adicionadas à versão "base" do *DSS Demo WebApp* descrito no capítulo 2, para se compreender que funcionalidades foram integradas na aplicação de forma a resolver o problema proposto no primeiro capítulo.

O quarto capítulo fará referência às técnicas e métodos de Desenvolvimento de Software Seguro que auxiliaram e foram aplicadas ao longo do desenvolvimento deste projeto.

O quinto capítulo servirá, inicialmente, para explicar o processo de instalação como também de utilização da aplicação onde serão apresentados alguns exemplos práticos da utilização das funcionalidades adicionadas no capítulo 3.

Por fim, o sexto capítulo tem o objetivo de resumir o trabalho desenvolvido, tal como salientar alguns pontos positivos e negativos encontrados ao longo do desenvolvimento e por fim apresentar as conclusões do trabalho.

2 DSS Demo WebApp

Neste capítulo serão apresentadas as funcionalidades da *DSS Demo WebApp* onde são também explicados alguns dos conceitos mais relevantes para o projeto desenvolvido. Para isso é feita uma análise da Framework DSS, que fornece um conjunto de bibliotecas e ferramentas que facilitam a implementação de funcionalidades de assinatura digital. E por fim é apresentada a WebApp, já que é a base do nosso projeto e demonstra a utilização da framework DSS para a implementação destas funcionalidades de assinatura digital.

2.1 Digital Signature Service - DSS

O DSS consiste numa biblioteca de software *open-source*, disponibilizada pela União Europeia, que visa fornecer a implementação de padrões para a criação e validação de assinaturas eletrónicas de acordo com a legislação europeia e com o Regulamento *eIDAS* em particular. [1]

A nível da *framework DSS* podemos identificar as seguintes principais funcionalidades:

1. O suporte a vários formatos de assinatura digital, incluindo *XAdES* (*XML Advanced Electronic Signatures*), *CAdES* (*CMS Advanced Electronic Signatures*), *PAdES* (*PDF Advanced Electronic Signatures*) e *ASiC* (*Associated Signature Containers*), que são padrões reconhecidos pelo regulamento *eIDAS* da UE, que estabelece padrões para a autenticação eletrónica e as transações eletrónicas na União Europeia;
2. Geração de Assinaturas, podendo gerar assinaturas digitais em qualquer um dos formatos suportados. Isto inclui a criação da estrutura de dados apropriada, a geração do hash do documento e a aplicação da assinatura digital;
3. Validação de Assinaturas, já que é capaz de validar assinaturas digitais, verificando se a assinatura foi corretamente aplicada e se o certificado associado é válido.
4. *Timestamps*, suportando a aplicação de *timestamps* às assinaturas digitais, o que pode ser importante para demonstrar a validade de um documento;
5. Suporte para Múltiplas Assinaturas, demonstrado com o uso dos *containers ASiC-S ou ASiC-E*, pois é capaz de lidar com documentos que foram assinados por várias partes, uma funcionalidade importante para documentos contratuais e outros documentos legais que requerem múltiplas assinaturas;
6. Para além das assinaturas digitais, a DSS também suporta a cifra de documentos, permitindo que os dados sejam protegidos contra acesso não autorizado.

Sendo esta *framework open source*, pode ainda ser utilizada, modificada e distribuída livremente, o que é o caso para o desenvolvimento deste projeto, não para o desenvolvimento de uma nova aplicação, mas para a integração de novas funcionalidades.

2.2 DSS Demo WebApp

A *DSS Demo WebApp* representa um exemplo prático da utilização da *framework DSS* no desenvolvimento de uma aplicação. Neste caso, tem como propósito mostrar as características desta *framework*, e, por isso as suas principais funcionalidades vão de acordo com a implementação dos pontos descritos em cima. Assim, das várias funcionalidades da *WebApp* podemos destacar as seguintes mais relevantes para o nosso projeto:

- Assinar documentos nos diferentes formatos suportados (*XAdES*, *CAdES*, *PAdES*, *JAdES* and *ASiC-S/ASiC-E*);
- Assinatura de *digests* pois permite criar uma assinatura digital para o *digest* (ou hash) de um documento. Desta forma a função hash garante a integridade do documento, enquanto a assinatura digital valida a autenticidade;
- Extensão de assinaturas, referindo-se à capacidade de adicionar informações adicionais a uma assinatura digital depois de ter sido criada;

- *Timestamping* de documentos, que é o processo de manter de forma segura o registo do momento em que uma determinada ação ocorreu, por exemplo na criação de um documento ou quando é realizada uma alteração de um documento;
- Verificar a validade de certificados digitais, assinaturas digitais e *timestamps*.

2.3 Parâmetros das Assinaturas Digitais

A DSS Demo Web App é ainda uma plataforma que oferece várias opções para personalizar a criação e validação de assinaturas digitais. É possível selecionar entre vários formatos de assinatura, incluindo XAdES, CAdES, PAdES e ASiC, cada um com suas próprias variações e níveis de garantia de segurança. A escolha de um destes formatos de assinatura vai restringir a seleção dos diferentes parâmetros que vêm a seguir, no entanto é possível fazer várias combinações diferentes na criação de assinaturas variando a escolha dos parâmetros.

Em primeiro lugar é possível escolher entre diferentes opções de *packaging*, que se refere à forma como a assinatura estará associada com o documento assinado, que podemos dividir em duas principais categorias:

- ***Enveloped***: A assinatura é incorporada com os próprios dados/ficheiro original, significando que os dados levam juntamente a assinatura.
- ***Detached***: A assinatura é armazenada separadamente dos dados/ficheiro original. Os dados permanecem inalterados e a assinatura é uma entidade separada que faz referência aos dados.

Uma assinatura pode ainda ser *enveloping* caso os dados assinados sejam incluídos como um sub-elemento da assinatura, i.e, os dados são encapsulados ou incluídos como parte integrante do objeto de assinatura. Por outro lado, pode ser *internally detached*, que embora seja utilizada muito raramente, são incluídos num ficheiro separado, tanto os dados assinados como os dados da assinatura, semelhante ao *detached*.

Depois, é possível escolher entre diferentes formatos de *containers* incluindo o **ASiC-S**, o **ASiC-E**, caso o *packaging* seja *detached* de forma a armazenar a assinatura, o documento e outras informações como *timestamps* e certificados. Estes permitem o encapsulamento do documento original e a sua assinatura digital num único ficheiro ou em ficheiros separados.

Importante ainda referir que para cada um dos formatos de assinatura existem normas publicadas pelo *ETSI* que descrevem as especificações e os requisitos para esses formatos, chamadas *baseline profiles* e que procuram garantir a interoperabilidade e a confiabilidade das assinaturas digitais. Existem quatro baseline profiles para os formatos CAdES, XAdES e PAdES, nomeadamente:

- ***BASELINE-B (Basic Signature)***: refere-se a uma assinatura básica que atende aos requisitos mínimos de conformidade. É uma assinatura que pode ser validada desde que o certificado de assinatura seja válido, ou seja, não tenha sido revogado ou expirado.
- ***BASELINE-T (Signature with Time)***: assinatura que inclui todos os requisitos do *BASELINE-B* e também exige que a assinatura digital seja associada a uma marca de tempo confiável, usada para comprovar que a assinatura foi criada num determinado momento, garantindo evidências de integridade e autenticidade ao longo do tempo.
- ***BASELINE-LT (Signature with Long-Term Validation Material)***: assinatura que inclui todos os requisitos do *BASELINE-T* e estabelece a necessidade de inclusão de material de validação a longo prazo na assinatura digital. Material esse que inclui informações ou referências que permitem verificar a assinatura num futuro distante, garantindo a validade da assinatura a longo prazo.
- ***BASELINE-LTA (Signature providing Long Term Availability and Integrity of Validation Material)***: assinatura que abrange todos os requisitos dos perfis anteriores (*BASELINE-T* e *BASELINE-LT*) e exige que a assinatura digital forneça garantia de disponibilidade e integridade dos materiais de validação de longo prazo, ou seja, os materiais de validação associados devem estar acessíveis e não tenham sido alterados ao longo do tempo

Existe ainda a opção de selecionar o **algoritmo de digest** (SHA1, SHA256, SHA384 e SHA512), que vai ser usado para gerar o hash do documento que for assinado digitalmente, este hash é então assinado com a chave privada para criar a assinatura digital. A assinatura e o hash podem então ser usados para verificar a autenticidade e a integridade do documento: qualquer alteração no documento após a assinatura resultaria em um hash diferente quando o documento é verificado, sinalizando que o documento foi alterado.

Em seguida, existem duas opções que podem ou não ser selecionadas são relativamente ao ***Allow expired certificate***, num contexto de que normalmente, um certificado digital tem uma data de validade após a qual não é mais considerado confiável. Porém, pode acontecer de ser necessário trabalhar com certificados que ultrapassaram a sua data de validade. Por exemplo, pode haver a necessidade de verificar uma assinatura digital que foi criada com um certificado que, desde então, expirou. Nestes casos podemos ativar esta opção, no entanto devemos ser cuidadosos já que confiar em certificados expirados pode trazer falhas de segurança.

Por fim, a opção de ***Add a content timestamp***. Esta opção permite criar um *timestamp* para a assinatura/-documento com o objetivo de provar que determinados dados existiam num certo ponto no tempo, isto é realizado recorrendo a um serviço de *timestamping* (TSA, Time Stamping Authority), que fornece um *timestamp* que comprova a existência dos dados num determinado momento. Um documento que possua um *timestamp* pode então ser verificado posteriormente para confirmar que existia e não foi alterado desde a hora e a data fornecidas pelo *timestamp*.

Para concluir, a seleção/utilização dos parâmetros que acabamos de descrever vão depender do objetivo pretendido para a assinatura. Isto porque existem parâmetros ideais para diferentes casos de uso das assinaturas, por exemplo aquelas que apenas necessitem de ser validadas a curto prazo devem escolher um nível (*baseline*) diferente das que sejam pretendidas para longo prazo.

3 Funcionalidades Desenvolvidas

Nesta fase do documento, o objetivo passa por demonstrar quais e de que maneira foram implementadas as funcionalidades propostas pelo professor para o projeto em questão. Antes de implementar as funcionalidades propriamente ditas que vão de em conta com os requisitos do projeto, primeiramente foi necessário transpor as alterações efetuadas para a versão **5.11.1** da DSS Demo WebApp e que se encontram descritas no capítulo 1. Inicialmente foi comparado cada ficheiro de código individualmente, da versão **5.8.2** correspondente à versão alterada (pelos colegas do ano passado) com a versão **5.8.2** original, de modo a garantir quais ficheiros de código haviam sido modificados. Após isso passamos à fase de transpor todas essas alterações para a versão **5.11.1** da WebApp. Após concluída esta etapa, pudemos finalmente passar para a implementação das funcionalidades pedidas que vão ser descritas de seguida.

3.1 Autenticação inicial

Nesta primeira etapa de desenvolvimento, foi necessário, para além de acrescentar a interface em si, intercetar os pedidos de forma a que o utilizador, sempre que queira efetuar uma operação que envolva a utilização da Chave Móvel Digital, tenha de estar autenticado para aceder às páginas oferecidas pelo serviço. Desta forma, procedemos à implementação da base de dados que fará o armazenamento de dados dos nossos utilizadores, nomeadamente credenciais de *Login* e número de telemóvel associado. Para este propósito, foi utilizado o *Spring Security*, juntamente com uma base de dados *NoSQL*, nomeadamente, o *MongoDB*.

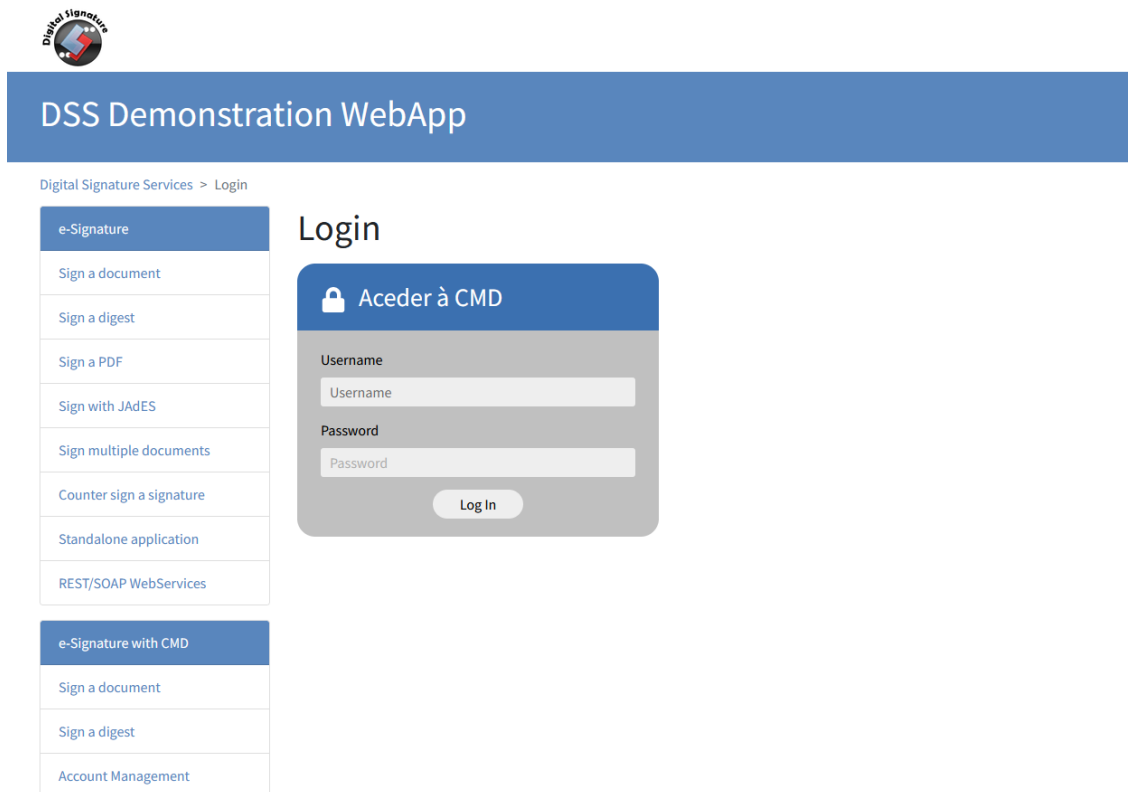


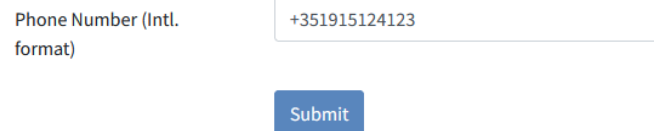
Figura 1: Formulário de *Login*

Resumidamente, se o utilizador não tiver sessão iniciada, pode efetuar qualquer operação que não necessite da CMD, mas assim que tente aceder a qualquer página de *e-Signature with CMD*, é redirecionado para uma página simples de *Login*, onde insere o seu *username* e *password*. No capítulo 4 são abordadas as técnicas de segurança utilizadas, no contexto do desenvolvimento de software seguro, como a validação de input.

3.2 Área de Utilizador

Para implementar este ponto já teríamos de ter a base de dados implementada de forma a que houvesse a possibilidade de guardar o número de telemóvel inserido pelo utilizador, assim como o apresentar quando este acede à sua área de utilizador, ou às operações que o necessitem, que serão todas as desenvolvidas no âmbito da *e-Signature with CMD*. Neste âmbito criamos uma página simples de gestão da conta onde é possível alterar o número de telemóvel associado à conta com sessão iniciada, e atualizar esse valor na base de dados *MongoDB*. A figura 2 ilustra a página de gestão da conta onde é possível fazer a alteração do número de telemóvel, o que for submetido nesta página será o número que recebe a mensagem com o código com o OTP (*one-time password*) necessário para confirmar todas as operações realizadas com a CMD.

Account Management



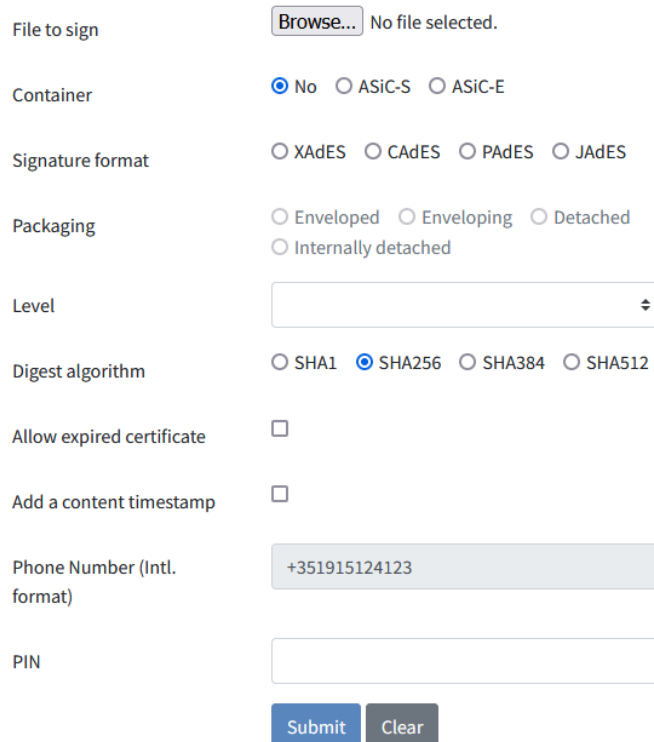
Phone Number (Intl. format)

Figura 2: Página de Alteração do Número CMD

3.3 Sign a document

Na opção desenvolvida de assinatura de documentos no contexto de *e-Signature with CMD*, foi adicionada a possibilidade de assinar documentos usando a CMD especificada na área de utilizador descrita anteriormente, desde que o PIN associado à CMD do número de telemóvel apresentado seja inserido corretamente. Esta página está ilustrada pela figura 3.

Sign a document with CMD



File to sign No file selected.

Container ☒ No ☐ ASiC-S ☐ ASiC-E

Signature format ☐ XAdES ☐ CAdES ☐ PAdES ☐ JAdES

Packaging ☐ Enveloped ☐ Enveloping ☐ Detached ☐ Internally detached

Level

Digest algorithm ☐ SHA1 ☒ SHA256 ☐ SHA384 ☐ SHA512

Allow expired certificate ☐

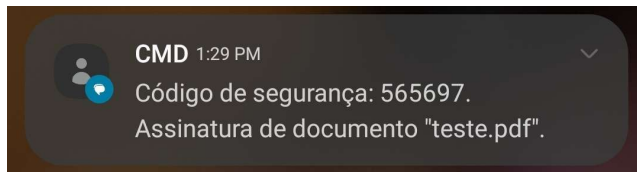
Add a content timestamp ☐

Phone Number (Intl. format)

PIN

Figura 3: Assinatura de Documentos com CMD

Após o formulário de assinatura de documentos com a CMD seja submetido corretamente, tanto nos parâmetros da assinatura como no PIN da CMD, passamos para o ecrã onde vamos inserir o OTP que vai ser recebido por mensagem para o número de telemóvel em questão. Podemos ver em baixo, na figura 4 uma demonstração deste processo.



(a) Mensagem com o OTP

Sign a document with CMD

OTP

(b) Página para Inserir o OTP

Figura 4: Processo de Envio e Confirmação do OTP (*one-time password*)

3.4 Sign a digest

Nesta última etapa, aplicamos o serviço da Chave Móvel Digital à classe exemplo do *sign a digest* da aplicação demo da DSS, da mesma forma que fizemos para a classe *sign a document* falada anteriormente. Para isso, tivemos de adaptar a página exemplo, mais uma vez, para suportar a nossa assinatura com CMD, e também lidar com o pedido de OTP da CMD, obtendo assim a nossa assinatura para assinar o digest de um certo ficheiro. O processo de envio e confirmação do OTP é realizado da mesma forma que o descrito no ponto anterior. A seguir, na figura 10, está representada a página *Sign a digest* no âmbito da coluna *e-Signature with CMD* desenvolvida, também representada na figura 6 com todas a funcionalidades desenvolvidas mencionadas neste capítulo.

Sign a digest with CMD

Signature format	<input type="radio"/> XAdES <input type="radio"/> CAdES <input type="radio"/> JAdES
Digest algorithm	<input type="radio"/> SHA1 <input checked="" type="radio"/> SHA256 <input type="radio"/> SHA384 <input type="radio"/> SHA512
Digest to sign (Base64)	<input type="text"/>
	<div>Drag a file here to compute digest</div>
Level	<input type="text" value=""/>
Allow expired certificate	<input type="checkbox"/>
Add a content timestamp	<input type="checkbox"/>
Phone Number (Intl. format)	<input type="text" value="+351915124123"/>
PIN	<input type="text"/>
	<input type="button" value="Submit"/> <input type="button" value="Clear"/>

Figura 5: *Sign a Digest* com a CMD

e-Signature with CMD
Sign a document
Sign a digest
Account Management

Figura 6: Coluna *e-Signature with CMD*, com as funcionalidades adicionadas

4 Métodos de Desenvolvimento de Software Seguro

Com o objetivo de mitigar riscos de segurança e proteger o software contra ameaças e para garantir e assegurar a integridade, confidencialidade e disponibilidade dos recursos de informação que o suportam, existem diversas práticas e abordagens que são utilizadas durante o processo de desenvolvimento de software. Essas técnicas envolvem considerar a segurança em todas as etapas do ciclo de vida do desenvolvimento de software, desde a concepção até à manutenção contínua do produto. O projeto em questão teve em contas as metodologias descritas a seguir.

4.1 Software Assurance Maturity Model (SAMM)

O **OWASP SAMM** consiste num modelo de referência que procura introduzir segurança no *SDLC* e que se baseia em **15** práticas de segurança agrupadas em 5 funções de negócios: **Governance, Design, Implementation, Verification** e **Operations**. O principal objetivo do modelo é ajudar a identificar as capacidades do grupo de trabalho e ajudar no desenvolvimento, através do Modelo de Maturidade (SAMM). O **OWASP SAMM** é organizado em três níveis de maturidade (1, 2 e 3), cada um com objetivos específicos e atividades recomendadas. À medida que uma organização avança nos níveis de maturidade, demonstra uma melhoria gradual na sua capacidade de garantir a segurança do software. Este modelo SAMM foi preenchido no âmbito das fichas de trabalho da disciplina e as questões respondidas estão colocadas no **7**.

4.2 PIA

Novamente, no âmbito das fichas de trabalho da disciplina, foi pedido que se utilizasse no projeto, uma ferramenta para realizar uma análise de impacto da proteção de dados (*PIA - Privacy Impact Assessment*), que ajuda a identificar e avaliar os riscos e impactos relacionados à privacidade de dados num determinado projeto, sistema ou organização. Para além de se ter analisado como é que este projeto se enquadra nos nove critérios que permitem avaliar se o processamento de dados pessoais resulta num risco elevado, recorreu-se à ferramenta **DPIA** para gerar o PIA para o projeto. Os resultados obtidos encontram-se no **8**.

4.3 Buffer Overflow

Tendo entendido que *buffer overflow* é considerada uma vulnerabilidade grave de segurança, visto que, quando explorada corretamente, pode permitir que um invasor execute código arbitrário no sistema, tivemos isso em atenção quando abordamos este projeto.

Felizmente, visto se tratar de uma aplicação escrita na linguagem Java, não se encontrar vulnerável a problemas de *buffer overflow*. Isto porque, a linguagem Java, a gestão da memória é realizada automaticamente pelo *"garbage collection"*, reduzindo a possibilidade de problemas de *buffer overflow*. Ao contrário das linguagens de baixo nível, no Java, os *buffers* são geridos pelo ambiente de execução *Java Virtual Machine (JVM)*, que garante a segurança e integridade dos objetos e *buffers* de memória, evitando "estouros". Deste modo, o programa encontra-se alheio a este tipo de problemas.

4.4 Hash da senha de um utilizador

O armazenamento seguro de senhas dos utilizadores é essencial para proteger as informações confidenciais. Em vez de armazenar senhas em texto simples (plaintext), é recomendado aplicar um algoritmo de hash seguro às senhas. O processo de *hashing* transforma a senha original em uma sequência de caracteres ilegível, tornando difícil ou impossível reverter o processo. Além disso, para aumentar ainda mais a segurança, é recomendado o uso de técnicas como *salting*, em que um valor único é adicionado à senha antes de aplicar o hash. O Spring Security, por exemplo, oferece suporte a algoritmos de *hashing* e *salting* para proteger as senhas dos utilizadores. No nosso caso usamos *bcrypt* para guardar as nossas passwords na base de dados.

4.5 Vulnerabilidade de inteiros

Foi também abordado nas aulas da disciplina, os problemas relacionados com a vulnerabilidade de inteiros, que podem levar a comportamentos inesperados, corrupção de memória e potenciais falhas de segurança. Estas

vulnerabilidades de inteiros podem ser exploradas por atacantes para causar "crash" software, *Denial of service*, distribuição indesejada de informações ou execução de código arbitrário. Este problema não afeta diretamente este projeto, visto que não estamos a guardar qualquer variável como inteiro. O número de telemóvel, o OPT e o Pin da CMD são guardados numa *String*.

4.6 Validação de input

A validação de input é um aspeto crucial no desenvolvimento de software seguro. Ao implementarmos a validação de input, é possível garantir que os dados fornecidos pelos usuários atendam aos requisitos esperados. Isso ajuda a prevenir ataques como injeção de SQL, cross-site scripting (XSS) e outros tipos de exploração de vulnerabilidades relacionadas à entrada de dados. Ao utilizar bibliotecas de validação, como a *Bean Validation Framework*, é possível aplicar restrições, como tamanho máximo, formato ou obrigatoriedade de um certo padrão, dos dados de entrada, proporcionando uma camada adicional de segurança.

No âmbito deste projeto, asseguramos isto em diferentes operações de input. No caso da autenticação, os campos são validados, isto é, garantimos que a *password* tenha entre 8 a 64 carateres, pelo menos um dígito numérico, pelo menos uma letra minúscula, uma maiúscula e um caractere especial. Se a senha não atender aos requisitos, haverá uma mensagem de erro.

Existe também validação no número de telemóvel inserido, que deve respeitar o código identificado Português +351 e deverá ter 9 algarismos, não podendo ter letras e o tamanho do input deve ser treze no total.

Em relação ao PIN da Chave Móvel Digital, este deverá ter entre 4 a 8 algarismos e o OTP recebido por SMS, deverá ter apenas algarismos e apenas seis. De seguida estão evidenciadas as expressões regulares definidas para garantir as validações anteriormente referidas:

```
//password
@NotNull
@Pattern(regexp = "^(?=.*[0-9])(?=.*[a-z])(?=.*[A-Z])(?=.*[!@#&()-[{}:;'/?/*~$^+=<>]).{8,64}$" , message = "{error.auth.password.wrongInput}")
private String password;

//Numero de telemóvel
@NotNull
@Pattern(regexp = "(\\+351) *9[0-9]{8}", message = "{error.cmd.userId.wrongInput}")
private String userId;

//Pin CMD
@NotNull
@Pattern(regexp = "[0-9]{4,8}", message = "{error.cmd.userPin.wrongInput}")
private String userPin;

//Código OTP
@NotNull
@Pattern(regexp = "[0-9]{6}", message = "{error.cmd.userOtp.wrongInput}")
private String userOtp;
```

Figura 7: Validação de input

4.7 Spring Security

O Spring Security é uma framework amplamente utilizada para implementar recursos de autenticação e autorização em aplicações Java. Esta fornece uma série de recursos de segurança, como proteção de URLs, controlo de acesso baseado em roles, autenticação, gestão de sessões e muito mais. O Spring Security é altamente configurável e pode ser integrado facilmente a diferentes tecnologias e estruturas, como ao Spring MVC. Ao utilizar o Spring Security corretamente, é possível garantir a proteção adequada dos recursos da aplicação e fornecer uma experiência segura

para o utilizador final. Além disso, o spring security já estava a ser usado pela aplicação demo da DSS, tornando assim ainda mais fácil de implementar um sistema de Login com recurso a uma base de dados. O Spring Security além de ser uma framework amplamente utilizada, constantemente testada e atualizada, é uma framework madura e, portanto segura. No nosso caso além de tratar do sistema de autenticação, trata de proteger os urls que necessitam de autenticação, criar a sessão e as *cookies* para a guardar, impedir *Cross-Site Request Forgery (CSRF)* e ataques *cross-site scripting (XSS)* através do *Content Security Policy (csp)*.

No âmbito do projeto, nós usamos o *Spring Security* para implementar uma interface *UserDetails* que fornece informações sobre um utilizador autenticado. A partir desta interface, implementamos métodos que nos retornam uma coleção de objetos que representam as permissões concedidas ao utilizador e métodos que são usados para verificar se a conta do utilizador está ativa, expirada, bloqueada ou se as credenciais estão expiradas. O Spring Security ajuda também no processo de autenticação de utilizadores, na medida em que verifica se o utilizador existe na base de dados por meio de uma classe chamada *"UserRepository"*.

5 Instalação e Utilização da Aplicação

Após a parte da explicação daquilo que foi implementado no projeto, esta parte do documento é também igualmente essencial para explicar de que forma deve ser usado aquilo que foi implementado. Para tal vamos descrever brevemente como utilizar a aplicação e como instalar a *DSS Demo WebApp*.

5.1 Requisitos

De forma a que a aplicação seja instalada com sucesso, devem-se atender os requisitos propostos pela documentação oficial do DSS [2]:

- Versão do Java igual ou superior a 8 (testado até à versão 19);
- Versão do *Maven* igual ou superior a 3.6;
- Versão do *Tomcat* superior a 8.5 para Java 8 e superior a 9 para Java 9 e superior (aplicação web);
- Memória e disco: ver os requisitos mínimos para a JVM utilizada. Em geral, quando mais memória disponível melhor a performance;
- Sistema operacional: sem requisitos específicos (testamos em Windows)

5.2 Instalação da WebApp

Após o cumprimento dos requisitos mínimos, podemos proceder à instalação da aplicação web, da seguinte forma:

5.2.1 Windows

De maneira a se poder testar e utilizar a DSS Demo Web App em ambiente Windows, com as novas funcionalidades adicionadas, é necessário seguir os seguintes passos:

- Efetuar "git clone" do conteúdo do repositório da aplicação;
- Aceder à diretoria *AP2-PD2/dss-demonstrations* e correr o comando: **mvn clean install**;
- Aceder a *AP2-PD2/dss-demonstrations/dss-demo-bundle/target* e descomprimir a pasta **dss-demo-bundle-5.11-SNAPSHOT**;
- Aceder à pasta descomprimida e executar o ficheiro **Webapp-Startup.bat**;
- Aceder ao endereço: <http://localhost:8080/>.

6 Conclusão

Dada a conclusão deste projeto é importante deixar algumas considerações, primeiro a nível do projeto, mas também relativamente à unidade curricular já que este é o último momento de avaliação e será a última oportunidade que teremos para escrevê-lo.

A nível do projeto desenvolvido, o maior desafio na sua concretização é referente à implementação de novas funcionalidades na DSS Demo WebApp. Isto porque foram tarefas que não só nos exigiam entender o funcionamento desta aplicação e das suas funcionalidades de forma a podermos integrar novas adições, mas também a necessidade de recorrer a frameworks e APIs de terceiros de forma a tornar a aplicação mais completa e segura. Com o desenvolvimento desta aplicação tivemos mais uma vez contacto com novas metodologias e técnicas de software seguro que é sempre um aspeto positivo, já que nos dota de conhecimentos importantes que devemos ter em consideração e levar para os nossos futuros projetos. A nível de melhorias futuras na aplicação desenvolvida, aquela que nos pareceu mais evidente seria a implementação da CMD nas restantes funcionalidades da *DSS WebApp* que o suportassem.

A nível da unidade curricular, consideramos que foi uma experiência enriquecedora no aspeto em que com os vários trabalhos desenvolvido ao longo do semestre, inclusivamente com desafios semanais, foi uma excelente oportunidade de conhecer APIs e código de terceiros, também como as tecnologias e metodologias mais recentes no campo da segurança de software. O facto de haver um contacto com tecnologias tão relevantes e recentes, por exemplo neste projeto com o uso dos padrões de assinatura emitidos pela própria comissão europeia é um aspeto muito positivo, já que sentimos que existe aplicabilidade destes conceitos no mundo real.

Para concluir, a experiência de desenvolvimento deste projeto foi bastante desafiante, mas permitiu-nos adquirir uma compreensão mais profunda e prática dos tópicos abordados na Unidade Curricular, e apesar de algumas dificuldades ao longo do projeto, no final consideramos que concluímos os objetivos e resultados propostos inicialmente.

Referências

- [1] European Commission, “Digital Signature Service - DSS.”
- [2] European Commission, “Digital Signature Service.”

7 Anexo I

SAMM (Software Assurance Maturity Model)

7.1 Pergunta 2.1

Identifique a maturidade de três práticas de segurança (à sua escolha) que utiliza no projeto de desenvolvimento 2 (PD2) da UC de Engenharia de Segurança (Fase Assess do SAMM)

As práticas de segurança escolhidas foram as *Threat Assessment*, *Security Requirements*, e *Security Testing*. As áreas que obtiveram menor pontuação e que precisam portanto de serem melhoradas foram as áreas *Governance* e *Operations*, com pontuações de, 1.29 e 1.31, respetivamente. As áreas *Verification* e *Construction* são as que melhor se adequam ao projeto, visto que as outras se adequam mais a um ambiente empresarial. Os resultados, em formato .xlsx, foram colocados na diretoria do repositório *AP2-PD2*.

Para cada uma das práticas de segurança identificadas na pergunta anterior, estabeleça o objetivo para a mesma (Fase Set the Target do SAMM), i.e., o nível de maturidade pretendido

Analisando o score obtido na *toolbox* do SAMM, o objetivo passa por chegar ao nível 1 de maturidade nas práticas de segurança *Threat Assessment* e *Security Testing* e ao nível 2 na prática *Security Requirements*. Após a conclusão deste objetivo, serão elaboradas novas avaliações destas mesmas práticas de modo a entender se será ou não exequível, com o tempo e os recursos de que disponibilizamos, um aumento destes objetivos (para avançar para os níveis seguintes de maturidade).

Desenvolva o plano para atingir o nível de maturidade pretendido

Para atingir os níveis desejados de maturidade descritos anteriormente, procederíamos da seguinte forma para as diferentes práticas de segurança.

Security Testing

1. Desenvolver um plano de teste abrangente para cobrir os requisitos de segurança identificados;
2. Identificar as ferramentas e técnicas de teste adequadas para os diferentes aspetos de segurança a serem testados, como vulnerabilidades conhecidas, autenticação, autorização, etc;
3. Utilizar ferramentas automatizadas e manuais para identificar e explorar possíveis vulnerabilidades;
4. Executar testes de penetração para verificar a resistência do sistema a ataques.
5. Acompanhar regularmente a maturidade da prática de teste de segurança e fazer ajustes conforme necessário.

Security Requirements

1. Documentar claramente os requisitos de segurança num documento de especificação formal ou documento de requisitos;
2. Garantir que os requisitos sejam consistentes, completos, não ambíguos e verificáveis;
3. Ouvir o feedback de outras atividades de segurança, e implementar esse feedback no desenvolvimento do projeto.

Threat Assessment

1. Realizar uma análise sistemática para identificar e documentar potenciais ameaças;
2. Avalie os riscos associados a cada ameaça identificada e priorizá-los consoante a sua gravidade;
3. Optar por práticas de codificação segura, controlos de acesso, monitorização para reduzir a probabilidade e o impacto das ameaças.

8 Anexo II

RGPD (Regulamento Geral de Proteção de Dados)

8.1 Pergunta 1.1

Os nove critérios aplicados no *Artigo 29* para avaliar se o processamento de dados pessoais resultará num risco elevado, são os seguintes:

1. Avaliação ou pontuação relativos a informações sobre o titular dos dados;
2. Implementação de software de tomada de decisão automatizada com efeito jurídico significativo ou similar, que visa tomar decisões sobre titulares de dados que produzam “efeitos jurídicos relativos à pessoa singular” ou que “afete de forma semelhante significativa a pessoa singular”;
3. Monitorização sistemática, para observar, monitorizar ou controlar titulares de dados, incluindo dados recolhidos através de redes ou “uma monitorização sistemática de uma área acessível ao público”;
4. Dados sensíveis ou dados de natureza altamente pessoal, como por exemplo, informações sobre opiniões políticas de indivíduos, bem como dados pessoais relativos a condenações ou infrações penais;
5. Dados processados a larga escala. O GDPR não define o que constitui “larga escala”, no entanto, existem recomenda alguns fatores a considerar para determinar o que constitui “larga escala”, como por exemplo o número de titulares de dados em causa, o volume de dados a serem processados, o tipo de processamento, duração do processamento, entre outros;
6. Interligação ou combinação de conjuntos de dados provenientes de duas ou mais operações de processamento de dados realizadas para diferentes finalidades e de um modo que irá além das expectativas de um indivíduo;
7. Tratamento de dados relativos a titulares de dados vulneráveis, devido ao desequilíbrio de poder entre os titulares dos dados e o responsável pelo tratamento. Os titulares podem ser crianças, funcionários, ou outros membros da sociedade com necessidades especiais de proteção;
8. Aplicação de novas soluções tecnológicas ou organizacionais, como combinar o uso de impressão digital e reconhecimento facial para melhorar o controlo de acesso físico. Tal tecnologia pode envolver novas formas de recolha e uso de dados, possivelmente com alto risco aos direitos e liberdades dos indivíduos;
9. Quando o tratamento dos dados em si “impede que os titulares dos dados exerçam um direito ou utilizem um serviço ou um contrato”. Isso inclui operações de processamento que visam permitir, modificar ou recusar o acesso dos titulares de dados a um serviço ou a celebração de um contrato.

O nosso projeto cumpre os seguintes critérios:

1. Critério 2 - A assinatura digital com o cartão de cidadão, em termos de legislação, é equivalente à assinatura de um documento com uma assinatura manual, e portanto as implicações legais dessa assinatura fazem cumprir este critério;
2. Critério 8 - A utilização da CMD e servidor de Timestamp do Cartão de Cidadão Português, pode ser entendida como o uso de uma tecnologia “inovadora”, fazendo cumprir este critério;
3. Critério 9 - No nosso projeto, o utilizador possui operações de autenticação para utilizar certas ferramentas do serviço e portanto se as operações de processamento de dados relacionadas ao login e password resultarem em restrições injustificadas ou excessivas aos direitos dos titulares de dados, como o direito de acesso a um serviço, então o critério pode ser aplicável. Se um serviço online restringisse o acesso de um utilizador com base em critérios discriminatórios, isso poderia implicar uma violação dos direitos e liberdade desse utilizador, e um DPIA seria necessário.
4. Critério 4 - Os dados de autenticação (username e password), o número de telemóvel do utilizador e até os documentos que o utilizador deseja assinar, contêm ou podem conter informações pessoais sobre o indivíduo.

5. Critério 3 - As informações que são fornecidas para o utilizador receber um OTP para efetuar um processo de assinatura, envolve o tratamento de dados pessoais e é necessário perceber se essa monitorização trata esses dados em larga escala ou se está relacionada a uma área acessível ao público. Se a aplicação está a monitorizar sistematicamente os utilizadores ou a recolher dados pessoais em grande quantidade, como parte do processo de autenticação e assinatura, pode ser necessário realizar uma DPIA.

Em baixo, encontra-se evidenciada uma visão geral dos riscos associados ao sistema, assim como a gravidade e os potenciais impactos. Os resultados foram obtidos através das respostas efetuadas às questões propostas pela ferramenta *CNIL* (*Commission Nationale de l'Informatique et des Libertés*) para desenvolver o DPIA do projeto.

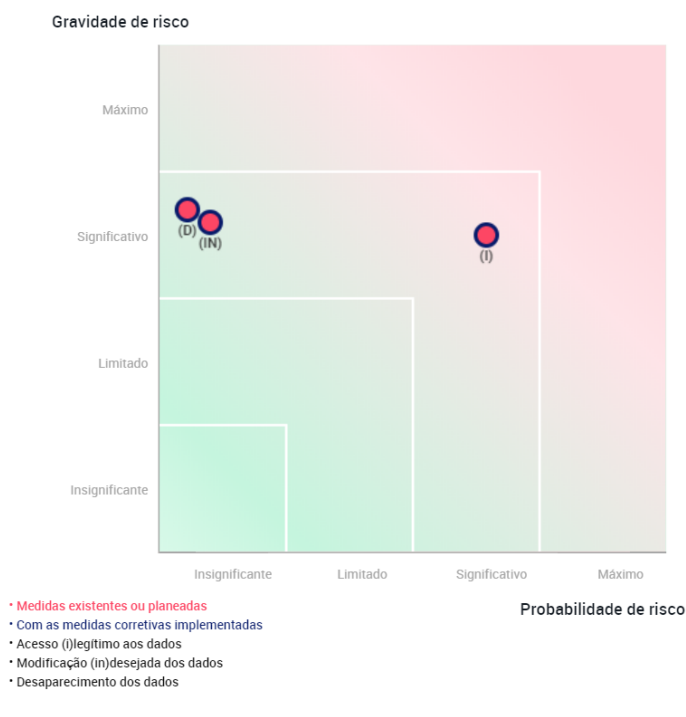


Figura 8: Gravidade de risco

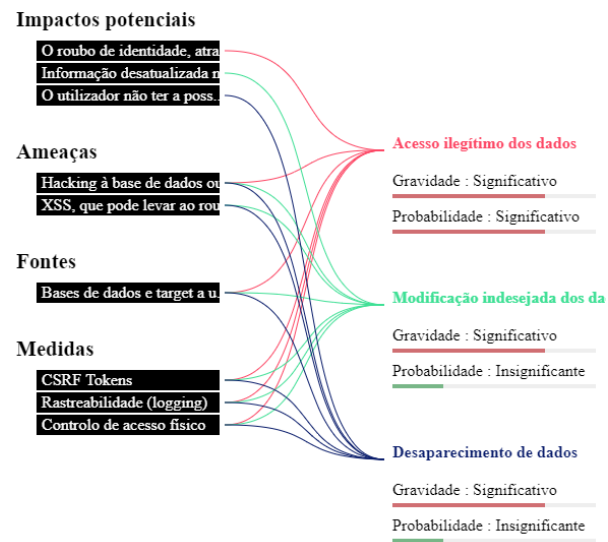


Figura 9: Impactos potenciais



Figura 10: Visão geral dos riscos