UNIVERSIDADE DE LISBOA

FACULDADE DE CIÊNCIAS

DEPARTAMENTO DE INFORMÁTICA

Privacidade no Ciclo de Vida do Desenvolvimento Seguro

Manuel Maria Marques Pinto da Costa e Oliveira

Mestrado em Engenharia Informática
Especialização em Engenharia de Software

Trabalho de Projeto orientado por:
Professor Doutor José Manuel da Silva Cecílio

2019

Agradecimentos

Em primeiro lugar quero agradecer ao meu Pai e à minha Mãe por tudo o que fizeram por mim. Todo o amor que me deram, e a educação que me proporcionaram, fez de mim a pessoa que sou hoje. Era impossível ter chegado onde cheguei sem o apoio incondicional que sempre tive da vossa parte. Obrigado Pai e obrigado Mãe, nunca poderei descrever por palavras o quão grato me sinto.

Em segundo lugar quero agradecer à minha irmã Mafalda. Talvez não saibas, mas considero-te um exemplo a seguir enquanto irmã e enquanto pessoa. Tenho a certeza que virás muitas mais vezes a deixar-me estupefacto com os teus feitos, a tua energia, e as lições que se aprendem de uma irmã mais nova. Obrigado Mafalda pela inspiração que tens sido.

Quero também agradecer à minha namorada por todo o apoio, carinho e motivação que me tem dado. A tua companhia ao longo de tantas horas foi o que me permitiu chegar aqui. Estiveste sempre disposta a apoiar-me e dar-me força, mesmo quando eu próprio duvidei que era possível. Obrigado Inês por estares sempre ao meu lado.

Quero agradecer a todos os meus amigos com quem cresci em Sintra. Já não vos vejo a todos com a mesma frequência que gostaria, mas como poderia isso ser possível? Entre estudos, trabalhos, filhos e tudo o mais, só mesmo amizades muito fortes podem persistir. Que as nossas amizades persistam por muitos mais anos, como persistiram até agora. A ti Manuel Rafael um especial obrigado, por seres o amigo que és e pela ajuda que me deste nesta etapa. Obrigado malta.

Quero agradecer também a todos os meus amigos que fiz na faculdade, aos mais velhos, aos mais novos, e em especial aos do meu ano. Só mesmo unidos como nós se passa por tanto e se chega ao fim com um canudo. Obrigado a todos vocês por me ajudarem a crescer tanto nos últimos anos.

Quero ainda agradecer aos meus orientadores José Cecílio e Sérgio Sá que tornaram esta tese possível. Obrigado por toda a paciência que tiveram e ajuda que me deram.

Por fim quero agradecer a toda a equipa de cyber da EY que me acompanhou ao longo deste último ano. A vossa boa disposição e companheirismo foram sem dúvida essenciais para o alcançar deste objetivo. Um especial obrigado ao Filipe Firmo e ao Paulo Rosado por terem sido também meus “orientadores”.

Aos meus Pais, à minha irmã, e à minha namorada.

Resumo

Nos dias de hoje, as aplicações de software desempenham um papel crítico para as organizações. À medida que as empresas dependem cada vez mais da integração destas aplicações no desempenho do seu negócio, levantam-se diversas questões relativas à segurança da informação. Empresas que nunca se preocuparam com a segurança das suas aplicações começam agora a ter de considerar os riscos que estas podem representar, uma vez que demonstrar a capacidade de proteger os dados digitais, tanto proprietários como de clientes, é cada vez mais uma necessidade para o negócio ao mesmo tempo que garante vantagem competitiva.

A maior parte dos processos de negócio de hoje em dia são suportados por aplicações de software, no entanto, os investimentos que têm vindo a ser feitos assentam mais na infraestrutura, deixando a segurança ao nível aplicacional para segundo plano. Em Portugal, uma grande parte das empresas estão ainda atrasadas no que diz respeito ao desenvolvimento de software seguro e devem mudar a forma como endereçam o ciclo de vida de desenvolvimento, nomeadamente ao nível de processos, que devem agora estar alinhados com o Regulamento Geral de Proteção de Dados, e ao nível de controlos técnicos, descritos em diversas normas de segurança e também agora na Resolução de Conselho de Ministros nº41/2018 obrigatória para os serviços públicos a partir de Outubro de 2019.

Esta tese propõe uma *framework* para endereçar o problema supracitado, na medida em que fornece diretrizes para que as organizações alterem a forma como desenvolvem software, integrando a segurança nas suas aplicações, em todas as fases do seu ciclo de vida. A partir do trabalho desenvolvido, é possível que uma organização adapte os seus processos, métodos e ferramentas, para que os conceitos *Security by Design* e *Privacy by Design* passem a fazer parte as metodologias de desenvolvimento, e desta forma as organizações consigam obter um maior nível de segurança nas suas aplicações. A *framework* proposta consiste num processo de desenvolvimento de um sistema de software, e uma metodologia de segurança a ser integrada nos processos de desenvolvimento. O resultado final do trabalho pode ser visto como uma solução particular de uma metodologia de segurança, com o seu próprio conjunto de funcionalidades benéficas. A metodologia proposta fornece diretrizes sobre quais as medidas de segurança que devem ser consideradas, sem que sejam sobrecarregados os processos de desenvolvimento utilizados. Em complemento, a metodologia proposta promove a utilização de repositórios de informação de segurança, relacionando soluções, ameaças, vulnerabilidades e políticas de segurança e encorajando a incorporação das melhores práticas nos processos de desenvolvimento.

Palavras-chave: software, segurança, privacidade

Abstract

Nowadays, software applications play a critical role for organizations. Since companies increasingly rely on the integration of these applications to assure the performance of their business, more and more information security issues are being raised. Companies that have never cared about the security of their applications are now beginning to consider the risks they may pose, since demonstrating the ability to protect digital data, both proprietary and customer, is increasingly business need while ensuring competitive advantage.

While it is known that security measures need to be taken to protect applications, it is not always so obvious how to implement them. The truth is that most business processes today are supported by software applications. However, the investments that have been made are based more on the infrastructure, leaving security at the application level to the background. In Portugal, many companies are still behind in software development and should change the way they address the software development lifecycle, particularly at the process level, which should now be in line with the General Data Protection Regulation, and at the level of technical controls, described in various safety standards and also now in Council of Ministers Resolution No. 41/2018, mandatory for public services from October 2019.

This thesis proposes a framework to address the above problem, It provides guidelines for organizations to change the way they develop software by integrating security into their applications at all stages of their life cycle. From the work developed, it is possible for an organization to adapt its processes, methods and tools, so that the concepts Security by Design and Privacy by Design become part of the development methodologies, and in that way organizations can achieve a higher level of security in their applications. The proposed framework consists of a development approach for software systems, and a security methodology to be integrated into the development processes. The final result of the work can be seen as a particular solution of a security methodology, with its own set of beneficial features. The proposed methodology provides guidelines on which security measures should be considered, without overburdening the development processes used. It promotes the use of security information repositories, relating solutions, threats, vulnerabilities and security policies and encouraging the incorporation of best practices in development processes.

Keywords: software, safety, privacy

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Acrónimos

BSIMM – Building Security In Maturity Model

CLASP – Comprehensive, Lightweight Application Security Process

DLP – Data Loss Prevention

DPIA – Data Privacy Impact Assessment

DPO – Data Privacy Officer

FDE – Full Disk Encryption

FPE – Format Preserving Encryption

FSE – File System Encryption

HTTPS – Hyper Text Transfer Protocol Secure

IDS/IPS – Intrusion Detection System/Intrusion Prevention System

LAN – Local Area Network

NSA – National Security Agency

OECD – Organization for Economic Co-operation and Development

OpenSAMM – Open Software Assurance Maturity Model

OWASP – Open Web Application Security Project

PET – Privacy-enhancing technologies

PMD - Privacy MethoDology (metodologia proposta)

RBAC – Role Based Access Control

RCM – Resolução de Conselho de Ministros

RGPD – Regulamento Geral de Proteção de Dados

SAFECode – Software Assurance Forum for Excellence in Code

SDL – Secure Development Lifecycle

SDLC – Software Development Lifecycle

SGSI – Sistema de Gestão de Segurança de Informação

SIEM – Security Information Event Management

SPPF – Security Process Pattern Framework

SSH – Secure Shell

SSO – Single-Sign-On

TLS – Transport Layer Security

VPN – Virtual Private Network

WAN – Wide Area Network

Capítulo 1

Introdução

As normas ISO/IEC 27001:2013 [13] e ISO/IEC 27002:2013 [14] pertencem à família ISO/IEC 27000:2013 que fornecem recomendações relativas às melhores práticas para a gestão da segurança da informação dentro do contexto de um Sistema de Gestão de Segurança de Informação (SGSI).

A norma ISO/IEC 27001:2013 especifica os requisitos para estabelecer, implementar, manter e melhorar de forma contínua o sistema de gestão para a segurança da informação dentro do contexto da organização. Para além disso, inclui os requisitos para a avaliação e tratamento dos riscos da organização relativos à segurança da informação. Os requisitos impostos pela norma ISO/IEC 27001:2013 são genéricos e pretendem ser aplicáveis a qualquer organização. O SGSI é descrito como parte do sistema de gestão global de uma organização, baseado numa abordagem de risco para o negócio que estabelece, implementa, opera, monitoriza, mantém e melhora a segurança da informação. O SGSI inclui estrutura organizacional, políticas, atividades de planeamento, responsabilidades, práticas, procedimentos, processos e recursos. Esta é a principal norma que uma organização deve utilizar como base para obter a certificação empresarial em gestão da segurança da informação, e é considerada a única norma internacional auditável que define os requisitos para um SGSI.

A ISO/IEC 27002:2013 fornece diretrizes para as normas e práticas de gestão da segurança da informação, incluindo a seleção, implementação e gestão de controlos que têm em conta o ambiente de risco da organização. Esta norma foi desenhada para organizações que pretendem selecionar controlos relativos ao processo de implementação de um SGSI baseado na ISO/IEC 27002:2013, implementar controlos comumente aceites para a segurança da informação e desenvolver as suas próprias diretrizes para a gestão da segurança da informação. É recomendável que a norma seja utilizada em conjunto com a ISO/IEC 27001:2013, mas pode ser também consultada de forma independente com a finalidade de adoção das boas práticas. A norma está dividida em diferentes cláusulas, cada uma contendo controlos de segurança relacionados. Existe um total de 14 cláusulas, sendo que muitos dos controlos poderiam ser colocados em cláusulas diferentes, mas para evitar duplicados ou conflitos, os controlos foram atribuídos a apenas uma cláusula e, nalguns casos, referenciados entre diferentes cláusulas.

Para uma organização que apresente um nível de maturidade relativo à segurança da informação pouco desenvolvido, a adoção destas normas e consequente implementação dos controlos nelas descritos representa uma grande transformação a nível organizacional. No contexto do tema da presente tese, a cláusula mais relevante é a 14, denominada “Aquisição, desenvolvimento e manutenção de sistemas”. Esta cláusula está dividida em três objetivos de controlo, que por sua vez contêm cada um deles diversos controlos. Para além disso, para cada controlo são dadas diretrizes para a sua implementação. Os objetivos de controlo e controlos estão descritos no **Anexo A**.

Com a aplicação dos controlos descritos nesta cláusula da norma, qualquer organização que desenvolva ou adquira software, poderá alcançar um maior nível de segurança nas suas aplicações. Todos estes controlos descritos fazem parte do Ciclo de Desenvolvimento de Software Seguro, sendo, portanto, essenciais para as organizações que desejem garantir a integridade, confidencialidade e disponibilidade dos seus sistemas e informação.

1.1 Motivação

Consideremos um operador postal multisserviços com uma presença muito forte em Portugal. Os seus sistemas de informação são determinantes para a sua atividade, eficiência e competitividade e a própria informação é um dos seus ativos mais valiosos. Atualmente está a passar por uma fase de grande transformação em que deseja aumentar o seu nível de maturidade relativo à segurança da informação. É importante referir que contêm centenas de aplicações, tanto para clientes internos (colaboradores da empresa) como externos (público no geral), incluindo canais massivos expostos na Internet para milhares de utilizadores. Muitas destas aplicações têm já anos de existência e como tal funcionam sobre tecnologias *legacy*. Muitas das aplicações tratam dados pessoais dos seus utilizadores aumentando a necessidade de garantir que estão seguras. Atualmente estão a passar por um processo de transformação em que desejam implementar os controlos da ISO/IEC 27002:2013. Como tal o desenvolvimento de software, que está dividido em 4 áreas distintas e é na sua maioria desenvolvido por colaboradores externos, é também um dos domínios a sofrer transformações.

A organização, consciente da importância do tema da segurança da informação para a organização e para os seus clientes, solicitou o apoio da EY para realizar a iniciativa "Segurança da Informação" proveniente do Programa RGPD (Regulamento Geral de Proteção de Dados). Esta iniciativa caracteriza-se pela definição da política de segurança da informação para a organização e de um conjunto de normativos associados.

De forma a desenvolver uma Política de Segurança, adaptada à realidade da organização e que responda às suas necessidades, a EY estabeleceu desde início uma abordagem proactiva, colaborativa e de parceria com a mesma para levar a cabo, numa primeira fase, uma análise do estado atual da segurança da informação.

Numa primeira fase, foram apresentadas as principais conclusões e recomendações resultantes da análise realizada, através da identificação de desvios face às boas práticas de segurança da informação definidas pela norma ISO/IEC 27001:2013. De forma a obter uma avaliação do estado de maturidade da organização relativamente à segurança da informação, foram realizadas 17 entrevistas aos colaboradores chave, com atuação nas áreas associadas aos domínios da norma ISO/IEC 27002:2013, tendo sindo também analisada a documentação existente e partilhada com a EY. A informação recolhida foi analisada e tratada de modo a obter uma avaliação mesurável da maturidade da organização, a sua situação atual e um conjunto de oportunidades de melhoria.

A utilização da norma ISO/IEC 27002:2013 como suporte para a realização desta análise, com o objetivo de identificar eventuais desvios face às boas praticas recomendadas pela mesma, teve em conta não só a sua ampla utilização, grau de reconhecimento e aceitação por parte da indústria, mas também o facto de na organização existir já uma empresa certificada ISO/IEC 27001:2013. O resultado desta análise servirá como como ponto de partida para o desenvolvimento da política de segurança da organização, garantindo assim um correto entendimento da realidade da mesma e o alinhamento com as práticas mais maduras já existentes em empresas dentro da organização.

Relativamente ao domínio 14, "Aquisição, desenvolvimento e manutenção de sistemas", foram tiradas algumas conclusões relativas ao estado atual da segurança da informação da organização, sendo elas:

- De um modo geral, não existem preocupações relativas à segurança da informação na aquisição e desenvolvimento de aplicações, pelo que não são identificados e analisados os requisitos associados (ex. segurança dos dados, acessos, privilégios);
- Apesar de já existirem preocupações de segurança derivadas do RGPD, ainda não estão definidas e implementadas metodologias e políticas de "*Privacy by Default and by Design*";
- Apesar da existência de boas práticas em algumas áreas (ex. utilização de protocolos seguros, certificados, mecanismos de autenticação), não está definida uma política de desenvolvimento seguro e transversal a todas as áreas de desenvolvimento;
- Existem alguns mecanismos de segurança implementados para proteção dos serviços expostos em redes públicas, no entanto, carecem de soluções mais robustas, processos de gestão e normas que definam os controlos a implementar;
- Não existe recolha, análise e tratamento automático de eventos de segurança nem alarmística associada, pelo que a proteção contra ataques e intrusões é reativa;
- Existem soluções *legacy* expostas ao exterior que não respondem às necessidades mínimas de segurança;
- Os testes de aceitação não contemplam preocupações de segurança e testes de segurança são apenas realizados *ad-hoc* e quando as aplicações já se encontram em produção;
- Foram identificadas algumas ineficiências relativamente à gestão e controlo de acessos aos repositórios de código e aos vários ambientes, derivado das diferentes equipas envolvidas;
- São utilizados dados reais em ambiente de teste e qualidade e, ainda que exista segregação entre ambientes, os controlos de segurança praticados nestes ambientes são inferiores aos de produção;
- Grande percentagem do desenvolvimento é realizado por entidades terceiras, no entanto, não existe controlo sobre o cumprimento dos requisitos de segurança por parte dos fornecedores.

A partir das observações efetuadas, foi possível identificar um conjunto de oportunidades de melhorias:

- Definir uma política de desenvolvimento seguro, e garantir a divulgação e formação de todos os visados no desenvolvimento aplicacional;
- Integrar a segurança da informação em todo as fases do ciclo de desenvolvimento, de forma a garantir que são endereçadas as preocupações e os requisitos relativos à segurança das aplicações e da informação;
- Incluir testes de segurança e código no processo de desenvolvimento aplicacional, preferencialmente antes da entrada em produção;
- Integrar os acessos aos repositórios de código e os vários ambientes numa solução de gestão de identidades e acessos (IAM – Identity and Access Management);
- Implementar um mecanismo de anonimização dos dados usados em ambientes de desenvolvimento, testes e qualidade;
- Avaliar a implementação de uma solução *enterprise* de *Web Application Firewall* para proteção dos serviços web expostos para o exterior.

Este é um exemplo de uma organização que teria muito benefícios em implementar um Ciclo de Desenvolvimento Seguro (SDL - Secure Development Lifecycle). No entanto, é preciso referir que o SDL é um processo a ser aplicado ao Ciclo de Vida de Desenvolvimento de Software (SDLC – Software Development Lifecycle), isto significa que são aplicadas as partes do SDL que fazem sentido para um

ambiente de desenvolvimento e produtos específicos. Quando não existe a experiência de alterar o SDLC, ou até quando não existe um SDLC de todo, é aconselhável que seja procurada ajuda profissional. Se não existir um SDLC poderá ser complicado utilizar um SDL, uma vez que a inexistência de um SDLC poderá por si só causar problemas, com ou sem a incorporação de um SDL.

De acrescentar ainda que ultimamente têm-se verificado diversos ataques bem-sucedidos [57, 58], resultando na divulgação de informação confidencial de grandes empresas e seus clientes, o que está a contribuir para um aumento de consciência de que ataques às aplicações podem ter grandes consequências. Para além disso, uma grande maioria dos especialistas reconhece que a eficácia dos atuais programas de segurança precisa de ser melhorada, de forma a diminuir os riscos da ocorrência de um ataque.

Recentemente, têm surgido alguns regulamentos, como é o caso do Regulamento Geral de Proteção de Dados (RGPD) [86], que dão extrema importância a conceitos como segurança por design, e passam a exigir às empresas que cumpram com requisitos mínimos de segurança, nomeadamente no que diz respeito ao tratamento de dados pessoais.

Em Portugal a realidade é que muitas das empresas que desenvolvem software não estão ainda preparadas para desenvolver software seguro, e com a entrada do RGPD e da Resolução de Conselho de Ministros nº41/2018 [87] têm agora pela frente o desafio de garantir que as suas aplicações estão seguras, ou arriscam-se a ter de pagar elevadas coimas.

1.2 Objetivos

O objetivo deste trabalho passa por desenvolver uma *framework* capaz de ajudar as empresas que desenvolvem software a aumentar o nível de segurança das suas aplicações. Todas as empresas que necessitem de adquirir, desenvolver ou manter uma aplicação têm interesse em garantir que a segurança da mesma não será comprometida. Quanto maior for a criticidade da informação que é tratada, maiores serão as consequências no caso de existir um ataque bem-sucedido, e as empresas podem ter de lidar com questões como perda de imagem ou até coimas elevadas no caso de não garantirem a segurança da informação.

A *framework* a desenvolver irá ajudar as empresas a aumentarem os níveis de segurança das suas aplicações, funcionando como um guia para a organização, relativamente a processos e ferramentas a utilizar. O foco da *framework* será garantir que as organizações estão a cumprir com as boas práticas para o desenvolvimento de software, assim como diversas normas de segurança. Para tal, será necessário começar por identificar e caracterizar o Ciclo de Vida de Desenvolvimento Seguro (principais aspetos de desenho, codificação e implementação, instalação) para a partir de aí ser possível a criação de um guia de implementação e utilização de software seguro para as organizações, considerando os diversos aspetos do RGPD e outras normativas.

1.3 Contribuições

O resultado mais evidente do trabalho desenvolvido e descrito ao longo do presente documento, para além de todo o trabalho de pesquisa realizado ao longo do processo, consiste essencialmente em duas contribuições que se complementam:

- uma arquitetura para o desenvolvimento de uma solução segura (identificando os componentes de segurança e as relações entre eles);

- uma metodologia de segurança para o desenvolvimento de software (considerando os processos de segurança que devem acompanhar o desenvolvimento e os artefactos que suportam tais processos).

As contribuições apresentadas permitirão às organizações construir e manter software mais seguro, servindo como ponto de partida para a reformulação de processos, métodos e soluções, de modo a endereçar as atuais preocupações existentes relativas à segurança da informação.

1.4 Enquadramento Institucional

A realização deste trabalho enquadra-se no âmbito da disciplina Dissertação/Projeto de Engenharia Informática (DPEI) da Faculdade de Ciências da Universidade de Lisboa (FCUL), sendo esta disciplina um requisito para a conclusão do Mestrado em Engenharia Informática (MEI). O trabalho foi desenvolvido numa instituição de acolhimento externa, neste caso uma empresa privada, EY.

1.5 Estrutura do documento

O presente documento encontra-se organizado da seguinte forma:

- O capítulo **"Introdução"** apresenta a motivação, objetivos, contribuições, contexto e organização do documento.
- O capítulo **"Desenvolvimento Seguro: Conceitos e Definições"** apresenta o Ciclo de Desenvolvimento de Software, as principais ameaças identificadas para os dias de hoje, as diferentes abordagens utilizadas para o Ciclo de Desenvolvimento Seguro (SDL) e o conceito de *Privacy by Design*.
- O capítulo **"Análise de Requisitos"** apresenta os requisitos de privacidade, os padrões e estratégias utilizados para o *design* de privacidade e o estado da arte, de um ponto de vista da privacidade, dos componentes essenciais num sistema seguro.
- O capítulo **"Metodologia de Segurança e Privacidade (proposta)"** apresenta a solução proposta em termos de arquitetura de alto nível e metodologia de segurança.
- O capítulo **"Avaliação"** apresenta uma avaliação da solução proposta.
- O capítulo **"Conclusões e Trabalho Futuro"** apresenta as conclusões e trabalho futuro.

Capítulo 2

Desenvolvimento Seguro: Conceitos e Definições

Neste capítulo são introduzidos os principais conceitos e definições que resultaram do trabalho de pesquisa realizado numa fase inicial. São introduzidos os conceitos relativos ao Ciclo de Vida do Desenvolvimento de Software e as principais ameaças à segurança do mesmo. Posteriormente são apresentadas algumas das soluções existentes no que diz respeito a desenvolvimento seguro. Por fim, é apresentado o conceito de *Privacy by Design* assim como os requisitos impostos pelo Regulamento Geral de Proteção de Dados.

2.1 Ciclo de Vida do Desenvolvimento de Software

A primeira fase do Ciclo de Vida de Desenvolvimento de Software passa por definir quais os requisitos para o software a ser desenvolvido. Apesar de existirem diversas abordagens para a especificação dos requisitos de software existem algumas partes que são consideradas essenciais e recomendadas para definir uma especificação com qualidade. Uma especificação completa deverá incluir os seguintes requisitos [1]:

- **Funcionais –** quais as funcionalidades a serem desenvolvidas
- **Interfaces externas –** de que forma é que o software interage com todos os componentes que o envolvem, desde pessoas até outros elementos de software
- **Desempenho –** qual o tempo de resposta, disponibilidade ou tempo de recuperação para cada funcionalidade do software
- **Atributos –** considerações relativas a portabilidade, exatidão, manutenção, segurança, etc.
- **Restrições do desenho na implementação –** normas requeridas, linguagem de implementação, políticas para a integridade da base de dados, limite de recursos, ambiente operacional, etc.

O tema da segurança é abordado como um dos atributos do software, sendo que este irá influenciar de forma severa o desenho da aplicação. Tendo em conta os requisitos impostos pela segurança poderá ser necessário, por exemplo, manter certas funcionalidades em módulos separados, permitir a comunicação apenas entre certas partes do programa, verificar a integridade dos dados para variáveis críticas, utilizar certas técnicas criptográficas, manter um histórico ou registos específicos. Deste modo, é necessário endereçar a segurança desde a fase inicial do desenvolvimento de software [1].

A engenharia de software tem sofrido grandes alterações nos últimos anos. O crescimento da Internet, assim como conceitos como Big Data, Software as a Service, computação na nuvem ou sistemas virtuais têm dado origem a sistemas cada vez mais complexos. Deste modo, o ciclo de vida de desenvolvimento de software tem sofrido bastantes alterações, uma vez que existem cada vez mais metodologias diferentes de desenvolver software e assentam cada vez mais em métodos contínuos e concorrentes, como o Agile ou DevOps [2].

No que diz respeito ao desenvolvimento de software, é possível dividir o desenvolvimento em vários processos ou atividades, cada uma delas com um propósito e resultados próprios. Em todas as fases do projeto é necessário endereçar as questões de segurança como uma das prioridades.

Numa primeira fase existem os processos contratuais, que definem as atividades necessárias para estabelecer um acordo entre duas partes interessas. Nesta fase, devem ser definidos e analisados os requisitos do sistema que, entre outros, deverão endereçar as preocupações ao nível da segurança. Dada a importância da segurança, deverão ser criados e documentados planos em separado que enderecem as preocupações de segurança. É já neste plano inicial que devem ser documentadas as regras sobre quem é que deve, ou não, aceder à informação relativa a cada projeto. Para além disso, a configuração da infraestrutura deverá ser planeada, e documentada, para endereçar as necessidades não só funcionais, como de desempenho, disponibilidade e segurança [3].

Software em produção está constantemente sob ataques. Desde à décadas que os atacantes exploram as vulnerabilidades existentes e a tendência é para o número de ataques continuar a aumentar. Firewalls, sistemas de deteção de intrusões e antivírus não são suficientes para resolver este problema. Só com um esforço conjunto, de toda a comunidade que desenvolve software, para desenvolver aplicações mais robustas e resilientes, poderá desencorajar os atacantes e aumentar a confiança dos utilizadores. É crucial que em cada fase do desenvolvimento de software sejam incluídas análises de segurança, defesas e contramedidas que resultem no lançamento de software mais seguro. Desde o levantamento de requisitos, até ao desenho e implementação, a segurança deve ser incluída no Ciclo de Desenvolvimento de Software, de forma a garantir aos utilizadores o melhor e mais seguro software possível [4].

2.2 Principais ameaças – OWASP Top 10

A OWASP (Open Web Application Security Project) é uma organização internacional sem fins lucrativos dedicada à segurança ao nível das aplicações web. Entre todas as ferramentas, documentos, fóruns, etc., disponibilizados pela OWASP, um dos seus trabalhos mais conhecidos é o "OWASP Top 10" que pretende evidenciar os 10 riscos mais críticos para aplicações *web*. Este é um trabalho desenvolvido por especialistas de todas as partes do mundo e recomendado a qualquer organização que tencione minimizar os riscos de segurança [5] e [6]. Os riscos descritos no relatório "OWASP Top 10 2017" podem ser consultados no **Anexo B**.

2.3 Ciclo de Desenvolvimento Seguro

De um modo geral, o software é desenhado para cumprir com uma série de funcionalidades e só depois, e em alguns casos, são acrescentadas medidas de segurança. Com o passar dos anos, e com o aumento do número de ameaças, tem havido uma mudança na forma de desenvolver software, e a segurança começa a ser introduzida nas aplicações desde o seu desenho. Em 2004, algumas organizações começaram a introduzir processos no Ciclo de Desenvolvimento de Software, o que deu origem a novas metodologias maioritariamente conhecidas por Ciclos de Desenvolvimento Seguros (SDLs). A primeira organização a fazê-lo foi a Microsoft, ao lançar o seu SDL que foi inicialmente implementado internamente. Desde então, surgiram processos como Microsoft SDL-Agile, Software Assurance Maturity Model (SAMM), Building Security in Maturity Model (BSMM), Secure Software Development Lifecycle (SSDL) ou Comprehensive Light-weight Application Security Process (CLASP). Todos estes processos fornecem guias para empresas e indivíduos que desenvolvem software introduzirem segurança no desenho das suas aplicações, e não apenas quando estas já se encontram em produção [7].

A empresa Errata conduziu um estudo com o propósito de perceber quais as organizações que não estavam a implementar a segurança no desenvolvimento de software, e o porquê de não o fazerem, visto que poderiam poupar dinheiro ao diminuir a necessidade de resposta a incidentes ou evitar danos á sua

imagem. Metade dos inquiridos responderam que a segurança era sempre uma preocupação no desenvolvimento de software, e apenas cinco dos participantes, representando 10,9% do total, responderam que a adoção de um SDL era desnecessária. No entanto, apenas 14 dos inquiridos, 30,4% do total, responderam que utilizam um processo SDL formal. De todos os inquiridos, onze, representando 23,9%, responderam que a implementação de um SDL consumia demasiado tempo; quatro, representando 8,7%, responderam que eram desnecessários; dois, representando 4,3%, responderam que eram demasiado caros; e sete, representando 15,2%, responderam que consumia demasiados recursos. Nove, representando 19,6%, responderam que não tinham conhecimento da existência de SDLs [7].

A verdade é que muitas vezes os clientes não procuram software mais seguro. Se for dada a escolha entre terem o software disponível hoje, com mais funcionalidades e menor custo, ou mais tarde, com maior custo e menos funcionalidades, mas mais segurança, os clientes vão optar pela primeira opção. No entanto, a descoberta de vulnerabilidades no software, nas fases iniciais do desenvolvimento, permitiria que a maior parte das vulnerabilidades facilmente nesta corrigida. Existem diversas abordagens SDL diferentes que permitem alcançar este objetivo [7].

2.3.1 Diferentes abordagens SDL

Microsoft SDL e SDL-Agile

Estes dois processos da Microsoft, disponibilizados de forma gratuita, adicionam passos específicos, relacionados com a segurança, ao processo de desenvolvimento, testes e lançamento de software [7]. Um dos passos mais importantes é a utilização de modelagem de ameaças para identificar as vulnerabilidades de uma aplicação e determinar a melhor forma de endereçar essas vulnerabilidades. A abordagem utiliza também análises de código estáticas e dinâmicas para encontrar possíveis problemas. Testes de *fuzzing* ajudam a encontrar problemas ao introduzir dados inválidos, inesperados ou aleatórios como *input*, para verificar se o programa entra em estado erróneo. A metodologia SDL-Agile foi pensada para o desenvolvimento ágil.

O Ciclo de Desenvolvimento Seguro (SDL) é um processo de desenvolvimento de software, criado pela Microsoft, com o objetivo de ajudar a construir software mais seguro e endereçar os requisitos de segurança necessários mantendo custos de desenvolvimento reduzidos. O processo consiste em acrescentar uma série de atividades focadas na segurança e respetivos entregáveis para cada uma das fases do desenvolvimento de software. Quando comparado com software que não esteve sujeito ao SDL, o software desenvolvido com este processo apresenta uma percentagem significativamente reduzida de vulnerabilidades de segurança descobertas. Um dos grandes desafios do desenvolvimento de software é criar software mais seguro que necessite de menos atualizações e represente um menor custo em termos de gestão de segurança [8].

Adotar o SDL significa alterar os processos existentes para o desenvolvimento de software ao integrar medidas que levam a uma melhoria em termos de segurança. O objetivo destas melhorias no processo é reduzir a quantidade e severidade das vulnerabilidades de segurança no software utilizado por clientes. A um alto nível, o processo considerado aceite na generalidade da indústria passa pelas fases de levantamento de requisitos, desenho, implementação, verificação e lançamento [8].

O aumento das preocupações a nível de segurança levou à criação de um conjunto de princípios para a construção de software mais seguro. Estes princípios são referidos pela Microsoft como SD^3+C – *Secure by Design*, *Secure by Default*, *Secure in Deployment*, e *Communications* [8].

Cada um dos elementos do SD^3+C impõe requisitos no processo de desenvolvimento sendo que os dois primeiros elementos, *Secure by design* e *Secure by default* têm o maior impacto. *Secure by design* implica que existam processos para prevenir a introdução de vulnerabilidades à partida enquanto que *Secure by default* implica que a superfície exposta do software seja minimizada [8].

O SDL tem implicações em cada uma das fases do desenvolvimento de software [8] para além de que acrescenta uma nova fase no início, a fase de treino, e outra no final, a fase de resposta. Em cada uma das fases são introduzidas atividades com o objetivo de aumentar o nível de segurança do produto final. As atividades, para cada uma das fases, são descritas no **Anexo C**.

A Microsoft alterou o seu processo de desenvolvimento de software, implementando o SDL, em resposta à crescente pressão feita pelos seus clientes que exigiam software mais seguro, indo ao encontro às suas necessidades e preservando a credibilidade da empresa. A necessidade de endereçar a segurança cresce à medida que as ligações existentes entre sistemas por todo o mundo aumentam, o que significa um ambiente cada vez mais propício a ataques e consequentemente mais riscos para os utilizadores de software. Os ataques passaram de alterar o código de websites por diversão para o que é hoje considerado cibercrime, em que o atacante poderá ter acesso a informação privada ou sensível ou utilizar um sistema para realizar ataques aos seus utilizadores, como é o caso dos esquemas de *phishing* ou extorsão [9].

OpenSAMM

A metodologia OpenSAMM [83] foi desenhada para ajudar as organizações a formular e implementar estratégias para o desenvolvimento seguro, adaptadas aos riscos específicos enfrentados pela organização. Esta abordagem, disponível gratuitamente, está dividida em quatro práticas de desenvolvimento de software, que por sua vez estão divididas em três níveis diferentes de sofisticação. As organizações podem determinar quais as práticas e níveis de sofisticação são mais apropriados para a sua realidade. As práticas tidas em conta são avaliação de ameaças, práticas de arquitetura segura, verificação e testes de software e gestão de vulnerabilidades [7].

O OpenSAMM é baseado em quatro funções de negócio essenciais envolvidas no desenvolvimento de software, cada uma delas com um conjunto de três práticas de segurança (**Figura 2.1**). Estas práticas de segurança são atividades associadas a cada uma das funções essenciais que ajudam a garantir a segurança de cada uma delas. Cada prática de segurança tem três níveis de maturidade associados (ou objetivos) com objetivos, atividades, resultados, métricas, custos, e níveis bem definidos [10].

Este modelo pode ser utilizado como referência para avaliar um programa de segurança e criar uma pontuação que mostra a sua evolução. Esta habilidade de atribuir um valor específico ao nível de maturidade de uma organização, permitindo acompanhar a sua evolução ao longo do tempo, é uma das

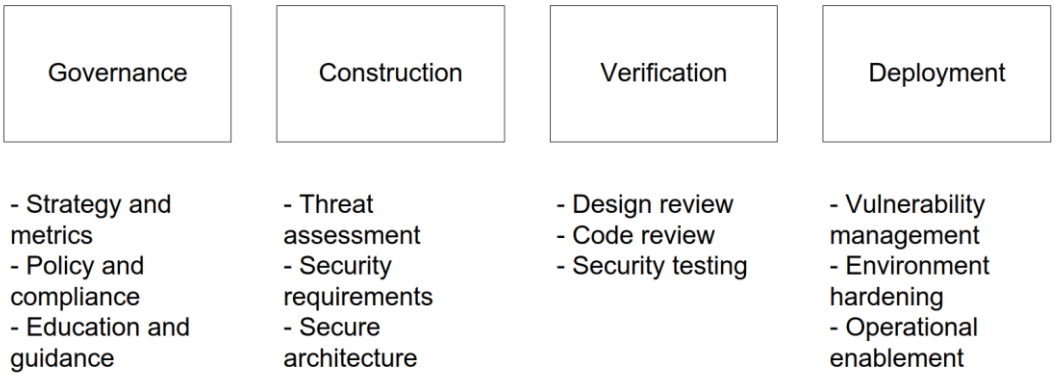

Figura 2.1 *– OpenSAMM. Obtido a partir de [83].*

maiores vantagens do modelo. A avaliação pode ser feita para cada uma das práticas de segurança ao atribuir valores a cada resposta, mas uma avaliação mais detalhada pode ser feita com algum esforço adicional de auditoria. Como exemplo, um *template* utilizado para entrevistas de avaliação de maturidade pode ser obtido da Internet. Este modelo está também preparado para facilitar a implementação de um programa de garantia de qualidade de software, ao fornecer um conjunto de passos que podem ser ajustados às necessidades de cada organização [10].

BSIMM

Building Security In Maturity Model (BSIMM) [84] é um modelo, disponibilizado de forma gratuita, que está organizado à semelhança do OpenSAMM uma vez que deriva de uma versão *beta* do mesmo. O modelo está dividido em quatro áreas de desenvolvimento de software, cada uma com três práticas de segurança, introduzindo passos como treino relativo a segurança, modelagem de ameaças, *design* de segurança, análise e garantia de qualidade, testes de segurança, integração com medidas existentes, e gestão de configurações e vulnerabilidades [7].

É uma abordagem prática, baseada em evidências empíricas e observação de dados de nove empresas distintas, nas quais se incluem serviços financeiros, vendedores de software independentes e empresas de tecnologia. Ao contrário de outras metodologias para o desenvolvimento seguro, o BSIMM não contém uma compilação teórica das melhores práticas, mas sim uma coleção de práticas reais utilizadas por empresas. As nove organizações consideradas seguem diferentes abordagens para o desenvolvimento seguro e as melhores práticas são derivadas da sua experiência. Deste modo, independentemente da metodologia utilizada, a grande maioria das melhores práticas teóricas propostas por outras metodologias estão presentes no BSIMM [10].

A *framework* do modelo BSIMM é chamada Software Security Framework e consiste em doze práticas normalizadas cada uma com diferentes atividades associadas (incluindo objetivos e atividades) e agrupadas em quatro domínios (**Figura 2.2**) [10].

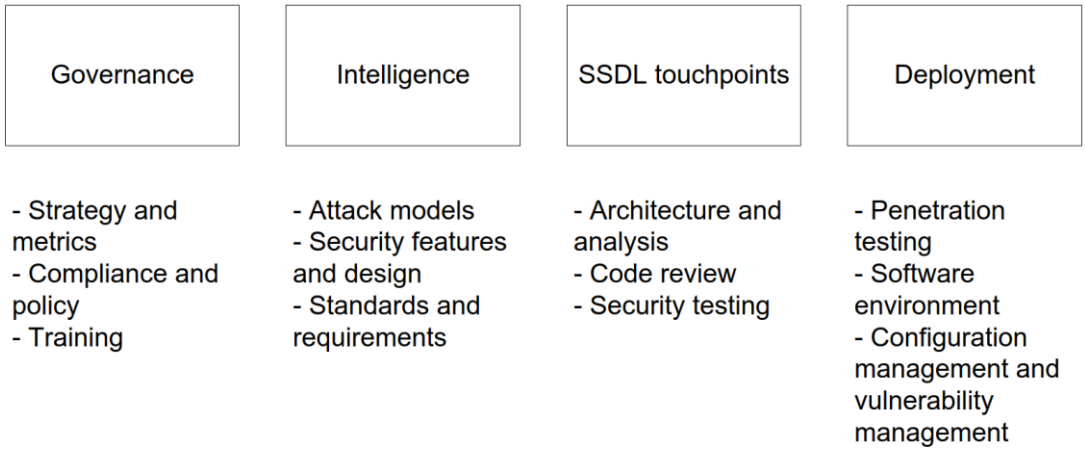

Figura 2.2 *– BSIMM. Obtido a partir de [84].*

À semelhança do OpenSAMM, o BSIMM considera três níveis de maturidade, com níveis de exigências de segurança incrementais, cada um deles com um conjunto de atividades associado a cada prática, de um total de 110 atividades. Este modelo pode ser utilizado como referência para diferentes organizações ajudando-as a priorizar as alterações de acordo com a pontuação obtida para cada uma das doze práticas. Tal como o OpenSAMM, a possibilidade de avaliar o estado de maturidade de cada uma das práticas de segurança é uma das vantagens do modelo. O BSIMM é, portanto, o resultado do estudo

subjacente que mostra as práticas que são de facto utilizadas para obter um processo de desenvolvimento de software mais seguro [10].

Este modelo é mais simples de aplicar do que o OpenSAMM mas carece de alguma informação, como por exemplo diretrizes sobre como medir e classificar as atividades de forma a obter um valor comparável que pode ser utilizado entre organizações [10].

Securosis' SDL

Consiste numa série de publicações em blogs relativas a desenvolvimento ágil. Por norma, o desenvolvimento ágil não endereça a segurança, e move-se demasiado rápido entre os ciclos para que tal processo seja fácil. Esta metodologia lida com questões como treinos e testes e dá ênfase à priorização de questões de segurança ajustando os processos ágeis. Por exemplo, testes de segurança poderiam ter lugar em períodos maiores do que apenas duas ou quatro semanas [7].

Securosis começou por ser uma publicação sobre um programa de segurança para o desenvolvimento de aplicações *web*. A publicação propõe um ciclo de desenvolvimento seguro focado na segurança de aplicações *web*. É uma abordagem prática, de baixo custo e apresenta o tipo de ferramentas que devem ser utilizadas em cada uma das três fases do ciclo proposto (**Figura 2.3**) [10].

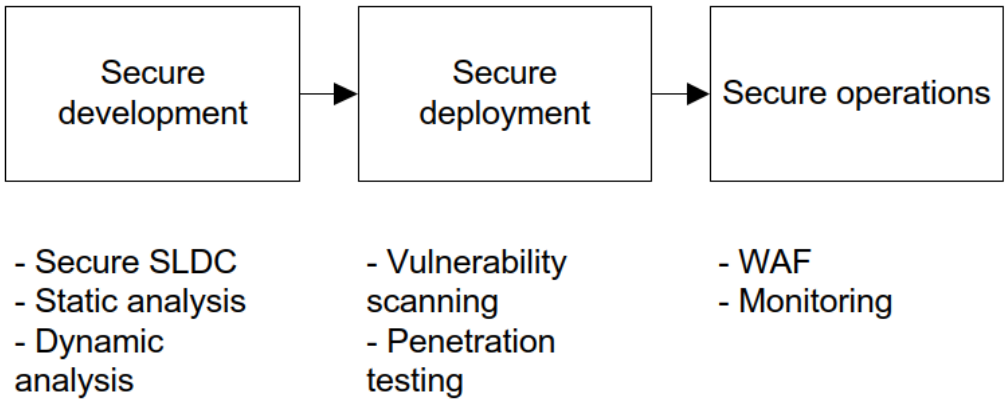

Figura 2.3 *– Securosis' SDL. Obtido a partir de [10].*

A descrição de cada uma das fases é a seguinte:

- **Secure Development** - Esta fase foca-se nos procedimentos iniciais necessários para desenvolver a aplicação como levantamento de requisitos, desenho, implementação e garantia de qualidade. Ferramentas de análise estáticas e dinâmicas são utilizadas para automatizar uma parte desta fase.
- **Secure Deployment** – É nesta fase que o código produzido está de acordo com as especificações e está pronto para uma avaliação de vulnerabilidades (encontrar *bugs* de segurança) e testes de penetração (classificar e explorar *bugs* de segurança).
- **Secure Operations** – Quando a aplicação é implantada devem ser utilizadas ferramentas preventivas para monitorizar a operação. O tipo de ferramentas utilizadas consiste em Web Application Firewalls e ferramentas de monitorização de atividade na aplicação e bases de dados.

CLASP

É uma metodologia lançada pela OWASP em 2006. Esta abordagem endereça questões de segurança do software desde as fases iniciais, críticas, do desenvolvimento de software. Utiliza sete melhores práticas incluindo programas para sensibilização sobre segurança para programadores, arquitetos, gestores de projeto, e até executivos. Adicionalmente, esta abordagem implica a definição dos requisitos de segurança e uma avaliação da aplicação para potenciais pontos fracos, através de métodos como modelagem de ameaças [7].

O modelo CLASP consiste numa série de componentes com as melhores práticas de segurança formalizadas que cobre todo o ciclo de vida do software, e não apenas o seu desenvolvimento, para que as preocupações relativas à segurança possam ser adotadas desde as fases iniciais do SDL utilizado pela organização. Este conjunto de 24 atividades relacionadas com segurança, que pode ser facilmente integrado no SDL utilizado pela organização, permite endereçar as vulnerabilidades de segurança de forma sistemática. Existem ferramentas que permitem automatizar o processo sempre que possível. É uma abordagem que depende muito da organização da equipa envolvida no projeto em diferentes funções, cada uma com um conjunto bem definido de atividades e responsabilidades que devem endereçar. Isto é feito em contraste com outros SDLs onde estas atividades são parte do processo de desenvolvimento do SDL [10].

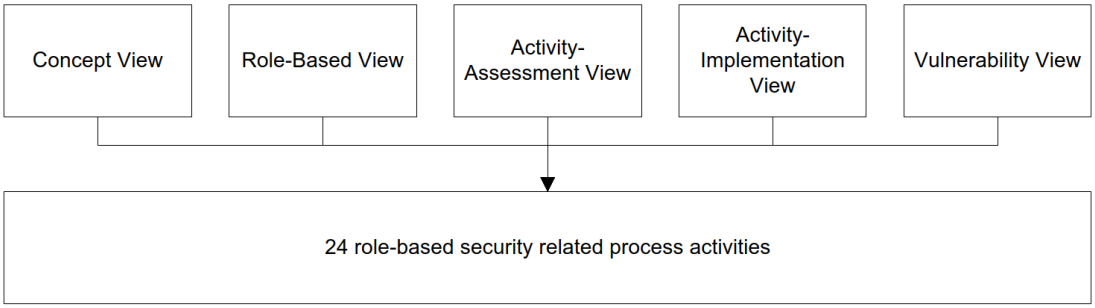

Figura 2.4 *– OWASP CLASP. Obtido a partir de [10].*

O CLASP está organizado em 5 perspetivas de alto nível do processo CLASP [10] chamadas CLASP Views (**Figura 2.4**), cuja descrição é a seguinte:

- **Concepts view** – define que os serviços básicos de segurança que devem ser satisfeitos para cada recurso são: autorização, confidencialidade, autenticação, disponibilidade, *accountability* e não repúdio. Isto é alcançado seguindo sete melhores práticas de segurança:
    - Realizar programas de sensibilização organizacionais;
    - Realizar avaliações às aplicações;
    - Capturar requisitos de segurança;
    - Implementar práticas de desenvolvimento seguro;
    - Desenhar processos de remediação de vulnerabilidades;
    - Definir e monitorizar métricas;
    - Publicar diretrizes de segurança operacional.
- **Role-Based View** – mostra de que forma é que a equipa de projeto deve endereçar as questões de segurança dependendo de responsabilidades específicas para cada função (gestores de projeto, auditores de segurança, programadores, arquitetos e outros). As funções de *designer*, arquiteto e gestor de projeto são aquelas que necessitam treino específico de segurança. Os programadores apenas precisam de programar bem, sem *bugs*, seguindo as políticas, normas e diretrizes da própria organização.

- **Activity-Assessment View** – mapeia as diferentes funções com as atividades de segurança (são 24 atividades no total) a implementar. Essas atividades são: programa de sensibilização em segurança, monitorizar métricas de segurança, especificar ambiente operacional, identificar a política de segurança global, identificar recursos e limites de confiança, identificar funções do utilizador e capacidade dos recursos, documentar requisitos de segurança relevantes, detalhar casos de má utilização, identificar a superfície de ataque, aplicar princípios de segurança no desenho, pesquisar e avaliar a postura de segurança das soluções tecnológicas, anotar o desenho com propriedades de segurança, especificar configurações de segurança de bases de dados, realizar análises de segurança dos requisitos do sistema e desenho (*threat modeling*), integrar análises de segurança no processo de gestão de código fonte, implementar interfaces para os contratos, implementar e elaborar políticas para os recursos e tecnologias de segurança, endereçar questões de segurança reportadas, realizar revisões de segurança ao código fonte, identificar, implementar e realizar testes de segurança, verificar atributos de segurança dos recursos, realizar assinaturas ao código, construir um guia de segurança operacional e gerir o processo de divulgação de questões de segurança.
- **Activity-Implementation View** – Detalha cada uma das 24 atividades baseadas em função
- **Vulnerability View** – detalha as consequências, tipo de problemas, tempos de exposição e técnicas para evitar e mitigar vulnerabilidades de segurança. Considera 104 tipos de vulnerabilidades, as suas categorias, tempos de exposição, consequências, plataformas afetadas, recursos, análise de risco e períodos para evitar ou mitigar vulnerabilidades.

Software Security Touchpoints

O modelo Software Security Touchpoints, que foi proposto pela empresa Cigital em 2004, é um conjunto de sete melhores práticas (*touchpoints*) que podem ser aplicadas por uma organização ao seu atual processo SDL (**Figura 2.5**), independentemente da natureza do mesmo (cascata, espiral, etc.) [10].

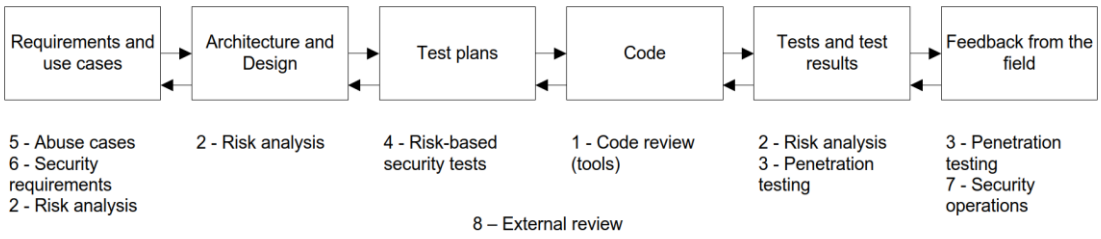

Figura 2.5 *– Software Security Touchpoints. Obtido a partir de [10].*

Relativamente aos *touchpoints*, são considerados dois níveis de bugs de software, ao nível do código fonte e ao nível da arquitetura, como tal os dois *touchpoints* considerados mais importantes são a análise ao código fonte e a análise à arquitetura [10]. Os *touchpoints*, apresentados por ordem de eficácia, são os seguintes:

- **Revisão de código** – utilizando ferramentas de análise estática;
- **Análise de risco** – baseada em padrões de ataque e modelos de ameaça;
- **Testes de intrusão** – utilizando uma abordagem *black-box* que deve ter em consideração a arquitetura do sistema;
- **Testes de segurança baseados em risco** – com requisitos rastreáveis;
- **Casos de abuso** – descrevendo o comportamento do sistema sob ataque;
- **Requisitos de segurança** – os requisitos de segurança devem ser definidos;

- **Segurança das operações** – monitorizar para falhas de segurança durante o funcionamento dos sistemas.

Software Assurance Forum for Excellence in Code

À semelhança do BSIMM, o SAFECode é um consórcio liderado pela indústria formado em 2007, constituído pelas empresas EMC, Juniper Networks, Microsoft, Nokia, SAP e Symantec. O seu objetivo é aumentar os níveis de confiança nos produtos e serviços de informação e comunicação. Este consórcio divulgou algumas publicações focadas em métodos e práticas de desenvolvimento seguro, uma visão geral sobre as melhores práticas da indústria à semelhança do BSIMM e um conjunto de princípios para a formação em desenvolvimento de software seguro [10].

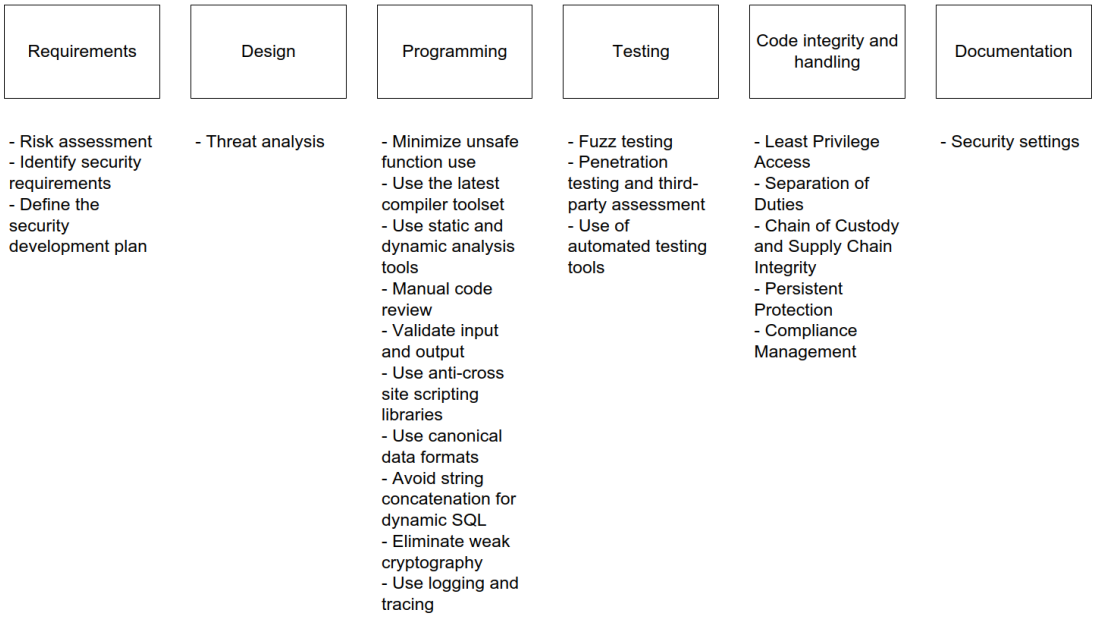

Figura 2.6 *– SAFECode. Obtido a partir de [10].*

O guia SAFECode contém uma lista das melhores práticas associadas ao desenvolvimento seguro, atualmente utilizadas pela indústria (**Figura 2.6**). A lista das melhores práticas [10] foca-se nos seguintes aspetos:

- **Requisitos** – incluindo formação em desenvolvimento seguro e testes;
- **Desenho** – incluindo análise de ameaças antes de *commits* de código;
- **Programação** – incluindo análises de código estáticas e dinâmicas, revisão; manual de código e validação de *inputs* e *outputs*;
- **Testes** – consistindo em testes de *fuzzing*, testes de intrusão e avaliações externas;
- **Integridade e gestão do código** – focando-se em aspetos como o princípio do privilégio mínimo, separação de funções e proteção persistente;
- **Documentação** – definindo as melhores práticas e como configurar o software para um maior nível de segurança.

O modelo SAFECode também salienta a necessidade de os líderes promoverem o uso destas melhores práticas, criando assim uma maior sensibilidade para a segurança em todos aqueles envolvidos no processo de desenvolvimento de software [10].

2.3.2 Comparação entre abordagens

A **Tabela 2.1** apresenta uma visão geral sobre algumas das diferentes abordagens para o Ciclo de Desenvolvimento de Software Seguro descritas neste capítulo [10].

Tabela 2.1 *– Comparação entre diferentes abordagens para o ciclo de desenvolvimento seguro. Obtido a partir de [10].*

	Nome	Ano	Pontos principais
SDL	**CLASP**	2006	24 melhores práticas de segurança que cobrem todo o SDL.
	Microsoft SDL	2004	SDL completamente integrado muito baseado em análise de risco e num conjunto de atividades para cada uma das fases. Adequado para o desenvolvimento de Sistemas Operativos e grandes empresas que desenvolvem software.
	Touchpoints	2004	7 melhores práticas que podem ser aplicadas a um ciclo de desenvolvimento de software existente.
Iniciativas de segurança	**OpenSAMM**	2009	Modelo de maturidade mensurável com 3 níveis diferentes e 12 práticas de segurança. Pode ser utilizado para *benchmarking.*
	BSIMM	2009	12 melhores práticas com 3 níveis de maturidade atualmente utilizadas pela indústria. Pode ser utilizado para *benchmarking.*
	SAFECode	2008	Visão geral das melhores práticas utilizadas pela indústria dando ênfase à importância da liderança.
	Securosis	2009	SDL específico para aplicações web focado na automatização através de ferramentas.

É possível observar que uma das restrições em aplicar um SDL é o seu custo consideravelmente elevado, tendo em conta a necessidade de alterar processos, treinar as pessoas, e em último caso atrasar a entrega do produto final. Algumas das abordagens podem ser utilizadas em projetos de larga escala, enquanto outras são mais adequadas a organizações mais pequenas [10].

Para além das metodologias descritas anteriormente, existem muitas outras lançadas desde então, como a Information Security Evaluation Methodology, lançada pela Agência de Segurança Nacional dos Estados Unidos da América, e que inclui informação sobre desenvolvimento de software seguro para a NSA (National Security Agency) e outras agências governamentais. Em todas elas, o maior desafio em desenvolver software seguro, prende-se com os custos e complexidade necessários para tal. Por outro lado, com o aumento do conhecimento e da sensibilização, assim como a integração da segurança em ferramentas de desenvolvimento conhecidas, muitas das dificuldades de implementar um SDL vêm-se reduzidas. A maior parte dos executivos quer ver uma justificação para os custos tendo em conta os benefícios e não existem evidências de que uma metodologia seja melhor que outra. Com o aumento da exigência por parte dos clientes, no que diz respeito a incidentes como fuga de informação, começa a ser mais fácil justificar o porquê de implementar um SDL [7].

O processo Microsoft SDL pode ser aplicado a projetos de grande dimensão, como desenvolvimento de sistemas operativos e aplicações muito grandes. Para além disso, a Microsoft fornece um conjunto de ferramentas para ajudar no processo, com a condição de serem utilizadas em ambiente Windows. O modelo CLASP pode ser adotado para projetos mais pequenos e que envolvem menos recursos. Pode ser utilizado como um processo SDL de raiz ou ser aplicado a um processo existente com um número reduzido de passos. O modelo Touchpoints pode ser integrado num processo SDL existente na organização, acrescentando segurança aos projetos de desenvolvimento de software. Este modelo apresenta atividades que devem ser aplicadas aos vários artefactos criados durante o desenvolvimento [10].

As restantes iniciativas focam-se em fornecer um conjunto de melhores práticas que estão a ser utilizadas por organizações referência na indústria de desenvolvimento de software, práticas estas que já demonstraram fornecem um bom balanço entre o seu custo e benefício. OpenSAMM e BSIMM também permitem realização de *benchmarks* entre a maturidade atual e a adoção das práticas propostas. Enquanto que o SAFECode chama à atenção para a necessidade de existir uma liderança que promova a segurança, alterando a mentalidade das equipas para que desenvolvam software mais seguro, o Securosis foca-se nas ferramenteas existentes que ajudam a automatizar uma grande parte dos processos necessários para implementar a segurança no desenvolvimento de software [10].

Apesar das diferentes visões apresentadas por cada uma das abordagens, todas elas contribuem para o desenvolvimento de aplicações mais seguras, e têm mais em comum entre si do que o contrário. É possível observar que entre as diferentes abordagens existem alguns pontos comuns [10]:

- Treino em segurança;
- Revisão de arquitetura;
- Revisão de código fonte;
- Testes de intrusão;
- Documentação e políticas de segurança.

As aplicações, especialmente aquelas expostas na Internet, são um alvo constante para atacantes e nunca estarão seguras, uma vez que novas vulnerabilidades e técnicas para as explorar estão sempre a ser descobertas. Até as aplicações desenvolvidas num ambiente SDL poderão tornar-se vulneráveis num período próximo ao seu lançamento. Esta é uma das razões pela qual o processo não termina com o lançamento da aplicação e deve continuar durante a sua manutenção sendo a manutenção após o lançamento, e durante todo o ciclo de vida da aplicação, um requisito obrigatório [10].

Todos os modelos defendem que a segurança das aplicações deve ser aplicada desde as fases iniciais do desenvolvimento, desde a definição de requisitos, arquitetura, desenho, código, testes, validação e manutenção. Esta abordagem para o desenvolvimento acaba por ter um menor custo a longo prazo e já provou dar resultados na indústria [10].

2.3.3 Análise de literatura

Uma vez que os vendedores de sistemas operativos desde há muito que investem na segurança dos seus produtos, os ataques têm vindo a mudar o seu foco para níveis mais altos da camada aplicacional, como por exemplo bases de dados ou software de antivírus. Isto verifica-se uma vez que a probabilidade de conseguir um ataque bem-sucedido é maior e o esforço para o realizar é de alguma forma recompensado, uma vez que um ataque a um sistema operativo não fornece diretamente dados valiosos para o atacante, ao contrário de bases de dados, ferramentas de gestão de relação com clientes (CRM - Customer Relationship Management) ou até um sistema de cuidados de saúde [9].

A implicação dos ataques às aplicações em vez de sistemas operativos não pode ser subestimada. A maioria dos fabricantes produz aplicações específicas para o negócio ou aplicações para os utilizadores finais, e não sistemas operativos. Para além disso, a maioria dos administradores experientes na área da segurança focaram-se em garantir a segurança dos sistemas operativos, ao introduzir defesas ao nível da rede, como por exemplo *firewalls*. Muitos utilizadores e vendedores de software viam a segurança como um problema do sistema operativo ou do perímetro da rede, mas essa realidade é agora considerada bastante limitada. Resumindo, se um software for acessível por utilizadores potencialmente maliciosos, dentro ou fora de um perímetro protegido por *firewalls*, então essa aplicação irá sofrer um ataque [9].

A decisão de construir software mais seguro nem sempre é tão linear quanto ir de encontro às necessidades dos clientes. Investimentos na segurança podem ser difíceis uma vez que o retorno desse investimento (ROI) é difícil de demonstrar. A justificação de que poderá existir um eventual ataque nem sempre é suficiente para justificar investimentos. Questões relacionadas com a privacidade, por outro lado, são mais fáceis de compreender, uma vez que é fácil compreender o que significa a divulgação de informação pessoal, sensível ou confidencial de forma não autorizada, e qual o impacto que poderá ter para uma organização. Esta divulgação não autorizada poderá levar, em alguns países, a implicações em termos legais ou regulamentares. Resumindo, a privacidade pode ser uma justificação para a implementação de medidas de segurança protegendo as aplicações de possíveis ataques. Segurança e privacidade são conceitos distintos, mas a existência de mecanismos eficazes de segurança é um pré-requisito para proteger a privacidade dos dados existentes nas aplicações [9].

Outro fator a considerar são os níveis mínimos a garantir para um determinado serviço (SLAs). Uma aplicação com tempos de resposta elevados até mesmo em baixo não vai de encontro às necessidades dos seus clientes. Da mesma forma que privacidade e segurança são conceitos distintos, mas interligados, o mesmo acontece para segurança e fiabilidade. Por exemplo, qualquer medida de segurança tomada para proteger de uma negação de serviço (DoS) é também uma funcionalidade de fiabilidade [9].

Em ultima análise, as questões relativas a segurança, privacidade e fiabilidade podem ser vistas como uma questão de qualidade do software. Algumas destas questões podem sobrepor-se, como por exemplo quando existe uma negação de serviço ou quando uma mensagem de erro, após a falha de uma aplicação, revela informação confidencial na sua mensagem de erro. Para além desta sobreposição, é importante ter ainda em conta que questões de segurança, privacidade e fiabilidade podem não estar diretamente relacionadas com a qualidade, como por exemplo se um utilizador divulgar voluntariamente informação confidencial, possivelmente através de um esquema de *phishing*, ou até se uma pessoa desligar acidentalmente o fornecimento de energia de um determinado servidor [9].

Deste modo, pode-se concluir que a segurança não pode ser endereçada de forma independente, mas sim como a interseção de conceitos como privacidade, fiabilidade e qualidade, e assim sendo, os esforços necessários para endereçar a segurança são mais evidentes, justificando os investimentos na melhoria do ciclo de vida de desenvolvimento de software [9].

Sistemas que dependem de software são uma realidade cada vez mais comum. À medida que estes sistemas vão estando cada vez mais interligados, os danos de um ataque bem-sucedido são cada vez mais pesados. Para além disso, um estudo recente realizado pelo U.S. Department of Homeland Security mostra que mais de 90% dos ataques são possíveis não por falhas de algoritmos criptográficos, redes ou hardware, mas sim por vulnerabilidades existentes ao nível das aplicações. Este problema apenas poderá ser resolvido com a adoção generalizada de um Ciclo de Desenvolvimento Seguro [11].

A causa mais comum para os sistemas que dependem de software serem tão inseguros é porque estes foram desenhados por humanos. É sabido que os humanos podem falhar e cometer erros, e para além disso o processo que é utilizado para desenvolver software não permite evita-los, uma vez que muitas vezes falha em identificar o problema desde o início, ou dá prioridade ao lançamento da aplicação no mercado em vez da segurança. Deste modo pode-se concluir que para resolver o problema da segurança no software não é suficiente a existência de uma nova ferramenta de desenvolvimento. Deste modo, levanta-se a questão sobre o que é possível fazer para resolver o problema. Primeiro, é preciso investigar e melhor compreender onde e como surgem os erros críticos no ciclo de desenvolvimento de software. Até à data, as primeiras observações não identificam bem o problema e como tal é necessário realizar mais investigação para avaliar o impacto positivo, e possíveis efeitos negativos, das estratégias de mitigação, e avaliar a aplicabilidade das diversas estratégias a diferentes metodologias, como por exemplo cascata, modelo V, SCRUM, DevOps, etc. Com base nessas descobertas, será possível então aos engenheiros de software criar ferramentas que ajudem a evitar os erros de segurança mais comuns [11].

2.4 Privacy by Design

Pagamentos online, IDs digitais, camaras de vigilância ou utilização de smartphones para colocar informação relativa à localização ou atividades dos seus utilizadores online, são algumas das circunstâncias dos dias de hoje que levantam preocupações relativas à privacidade. Um próximo passo para garantir a privacidade de todos é a adoção do conceito *Privacy by Design* [12].

Este conceito traduz-se na implementação de vários princípios de privacidade diretamente na especificação de sistemas tecnológicos, de forma a que as regras relativas à privacidade passem a fazer parte das operações e gestão do tratamento de dados. Esta abordagem pretende salvaguardar a privacidade dos utilizadores sem diminuir a eficácia e capacidade de cada tecnologia [12].

O conceito *Privacy by Design*, ao ser integrado no RGPD, deixa de ser um conceito teórico e passa a ser uma obrigação legal que todas as entidades que processam informação têm de endereçar. De acordo com o regulamento, deverão ser tomadas medidas para garantir que os princípios de proteção de dados são implementados no desenho de aplicações ou sistemas. Mecanismos como pseudonimização, minimização dos dados e certificação são apenas alguns exemplos da aplicação do conceito *Privacy by Design* explícitos no regulamento [12].

A implementação do conceito terá um grande impacto nas organizações uma vez que estas terão provavelmente que usar novas tecnologias, desenvolver novos produtos, serviços ou sistemas, de forma a garantir que os princípios de *Privacy by Design* são respeitados e os dados pessoais processados estão seguros. As organizações terão de adotar novas políticas e processos internos que permitam avaliar numa base regular o cumprimento das obrigações relativas à privacidade. Para concluir, a implementação do conceito *Privacy by Design* é uma necessidade imediata para os setores onde a privacidade dos dados, e a proteção dos dados pessoais, é uma preocupação emergente [12].

A norma internacional ISO/IEC 29100:2011 [15] fornece uma *framework* de alto nível para a proteção de informação pessoalmente identificável no que diz respeito aos sistemas de informação. Esta *framework* ajuda as organizações a definir os seus requisitos de proteção de privacidade através da:

- Especificação de uma terminologia comum de privacidade;
- Definição dos intervenientes e as suas funções no tratamento de dados pessoais;
- Descrição dos requisitos de proteção de privacidade;
- Referenciação dos princípios de privacidade conhecidos.

2.4.1 OWASP Top 10 Privacy Risks

O projeto OWASP Top 10 Privacy Risks [16] foi lançado em setembro de 2014 com o objetivo de evidenciar quais os principais riscos a enfrentar no que diz respeito à proteção da privacidade dos dados. Identificar e entender quais são os riscos existes é o primeiro passo para a implementação do conceito *Privacy by Design*, pois só assim é possível implementar as medidas necessárias para melhorar a privacidade. Os riscos identificados abrangem aspetos tecnológicos e organizacionais. Não se restringem apenas os aspetos legais, mas também considera os riscos do quotidiano associados à utilização dos sistemas. Os principais riscos de privacidade identificados pela OWASP são os seguintes:

- **P1 Vulnerabilidades em aplicações Web –** Uma vulnerabilidade é um problema fundamental em qualquer sistema que processe ou armazene dados dos seus utilizadores. A falha em desenhar e implementar adequadamente uma aplicação, detetar um problema ou aplicar imediatamente uma correção (*patch*) irá provavelmente resultar numa violação de privacidade.

- **P2 Fuga de dados no lado do Operador –** Consiste em prevenir a fuga de qualquer informação que contenha ou esteja relacionada com os dados dos utilizadores, ou os próprios dados em si, para qualquer parte não autorizada, resultando numa perda de confidencialidade dos dados. Uma fuga de informação pode ocorrer de forma intencional, derivada de intenções maliciosas, ou de forma acidental, por exemplo quando existe uma gestão de controlo de acessos insuficiente, armazenamento inseguro, duplicação dos dados ou falta de sensibilidade para o tema.

- **P3 Resposta insuficiente a fugas de dados –** Não informar os indivíduos afetados sobre uma possível violação ou fuga de dados, resultante de eventos intencionais ou não intencionais. Falha em remediar a situação eliminando a sua causa ou não tomando ações para limitar a fuga.

- **P4 Exclusão insuficiente de dados pessoais –** Falha na eliminação efetiva ou temporária dos dados pessoais após o término do propósito especificado ou mediante solicitação.

- **P5 Políticas, Termos e Condições não transparentes –** Não fornecer informações suficientes para descrever como os dados são tratados, como são recolhidos, armazenados, processados e excluídos. Falha ao tornar estas informações facilmente acessíveis e compreensíveis para as partes interessadas.

- **P6 Recolha de dados não necessários para o propósito original –** Recolher dados descritivos, demográficos ou outro tipo de dados relacionados com o indivíduo que não sejam necessários para o propósito do sistema. Também se aplica a qualquer tipo de dados cujo titular não tenha fornecido o seu consentimento.

- **P7 Partilha de dados com terceiros –** Fornecer dados de um indivíduo a qualquer terceiro sem primeiro obter o consentimento do titular. Esta partilha pode ocorrer devido a uma compensação monetária pela partilha dos dados, devido ao uso inapropriado de recursos de terceiros (por exemplo, mapas) ou até mesmo devido a erros.

- **P8 Dados pessoais desatualizados –** Consiste na utilização de dados desatualizados, incorretos ou falsos, possivelmente derivado de uma falha em atualizar ou corrigir os dados.

- **P9 Falta de expiração de sessão ou expiração inexistente –** Falha em efetivamente forçar o término de sessão o que poderá resultar na recolha de dados adicionais sem o consentimento ou consciencialização do indivíduo.

- **P10 Transferência de dados insegura –** A falha em garantir a transferência de dados em canais cifrados e seguros poderá levar à fuga de dados. Falha em implementar mecanismos que reduzam as hipóteses de fuga de dados.

2.4.2 Sete princípios fundamentais de privacidade

Privacy by Design é um conceito baseado em incorporar proactivamente conceitos de privacidade no desenho e operação de sistemas de informação, infraestruturas de rede e processos de negócio. O conceito *Privacy by Design* assenta em sete princípios fundamentais de privacidade:

- **Proativo não reativo - preventivo não corretivo –** Antecipar, identificar e prevenir eventos invasivos antes que eles aconteçam, o que significa agir antes dos acontecimentos e não depois.
- **Utilizar a privacidade como a configuração padrão –** Garantir que os dados pessoais são protegidos automaticamente em todos os sistemas de informação ou processos de negócios, sem que nenhuma ação adicional seja exigida por parte do indivíduo para proteger a sua privacidade, e sem que o consentimento para a partilha de dados seja assumido.
- **Incorporar privacidade no design –** As medidas de privacidade não devem ser um complemento, mas componentes totalmente integrados do sistema.
- **Manter todas as funcionalidades (soma positiva, não soma zero) –** O conceito *Privacy by Design* emprega uma abordagem "*win-win*" para todos os objetivos legítimos de *design* do sistema, ou seja, tanto a privacidade como a segurança são importantes, e não é necessário abdicar ou prejudicar funcionalidades para alcançar as mesmas.
- **Garantir a segurança de ponta a ponta –** A segurança do ciclo de vida de dados significa que todos os dados devem ser retidos com segurança enquanto necessários e destruídos quando não forem necessários, ou seja, recolher e manter apenas os dados estritamente necessários, assim como destruir os dados quando não forem relevantes ou não existir licitude para os manter.
- **Manter a visibilidade e a transparência –** Assegurar aos *stakeholders* que os processos de negócio e as tecnologias são operadas de acordo com os objetivos definidos de forma visível, transparente, aberta, documentada e que estão sujeitos a verificações independentes.
- **Respeitar a privacidade do utilizador (centrada no utilizador) –** Manter os sistemas centrados nos seus utilizadores, isto significa oferecer aos utilizadores opções de privacidade granulares, padrões de privacidade maximizados, avisos detalhados de informações de privacidade, opções fáceis de usar e notificação clara de alterações.

2.4.3 OECD Privacy Framework

A *framework* de privacidade da OECD (Organisation for Economic Co-operation and Development) assenta em 8 princípios:

- **Princípio de limitação da recolha –** Devem existir limites para a recolha de dados pessoais e quaisquer dados recolhidos devem ser obtidos através de meios legais e justos e, quando apropriado, com o conhecimento e consentimento explícito do titular dos dados. Quando apropriado uma vez que podem existir casos em que o consentimento (explícito) não é necessário. Por exemplo, ao recolher os dados de um cartão de crédito para efetuar um pagamento, ou recolher a morada para envio de uma encomenda.
- **Princípio de qualidade dos dados** – Os dados pessoais devem ser relevantes para o propósito que foram recolhidos e nessa mesma medida devem ser exatos, completos e permanecer atualizados.

- **Princípio de definição da finalidade –** O propósito para a recolha de dados deve ser especificado antes da recolha e a sua utilização deve ser limitada ao cumprimento do objetivo inicial ou qualquer outro objetivo que não seja incompatível com o propósito inicial desde que o mesmo seja especificado no caso de existir alguma alteração.
- **Princípio de limitação de utilização –** Os dados pessoais não devem ser divulgados, tornados disponíveis ou de qualquer forma utilizados para qualquer propósito que não o estipulado exceto se com o consentimento do titular ou por força da lei.
- **Princípio da salvaguarda de segurança –** Os dados pessoais devem ser protegidos com as medidas de segurança adequadas contra riscos de perda ou acesso, destruição, utilização, modificação ou divulgação não autorizada.
- **Princípio de abertura –** Deve existir uma política geral sobre a utilização dos dados pessoais, especificando os principais propósitos do seu tratamento assim como a identidade e meios de contacto do responsável pelo tratamento.
- **Princípio de participação do indivíduo –** Os indivíduos devem ter o direito a:
    - Obter do responsável pelo tratamento a confirmação de quais os dados pessoais que lhe dizem respeito e que são ou não tratados;
    - Que lhe sejam comunicados os dados relacionados:
        - Dentro de um prazo razoável;
        - Por um preço, se existir, que não seja excessivo;
        - Por meios razoáveis;
        - De um modo prontamente compreensível;
    - Obter explicações caso o pedido seja rejeitado e sobre como contestar a rejeição;
    - Contestar quaisquer dados que lhe sejam relacionados, e caso a contestação seja aceite, ter os seus dados apagados ou retificados.
- **Princípio de responsabilização –** O responsável pelo tratamento terá de fornecer evidências da conformidade para com os princípios descritos anteriormente.

2.4.4 RGPD

Antes do RGPD

A Diretiva de Proteção de Dados, introduzida em 1995, estabeleceu os elementos básicos para a proteção de dados dos cidadãos da União Europeia. A Lei de 1995 não estava preparada para acompanhar a explosão que existiu no que diz respeito à recolha e armazenamento de dados dos dias de hoje. Apesar de ter sido estendida para considerar os dados existentes na Internet, não foi desenhada para cumprir o efeito. O mundo mudou desde 1995, e para endereçar estas questões foi necessário criar mais leis para proteger os dados nesta era digital.

Depois do RGPD

O novo Regulamento Geral de Proteção de Dados (RGPD) foi desenhado para proteger os dados nos dias de hoje e no futuro. Os seus objetivos incluem o fornecimento de um conjunto de regras robusto para proteger os direitos fundamentais e liberdades dos indivíduos da União Europeia, permitir o movimento livre de dados pessoais dentro da União Europeia e harmonizar a legislação relativa à proteção de dados em todos os estados membros.

O RGPD aplica-se a todas as organizações que estão estabelecidas dentro da União Europeia e que tratem dados pessoais e a todas as organizações estabelecidas fora da União Europeia no caso de tratarem

dados de indivíduos da mesma e lhes oferecerem produtos, serviços ou monitorizarem o seu comportamento.

A partir do RGPD, é possível extrair um conjunto de princípios com implicações no tratamento de dados pessoais. De seguida, são apresentadas as transcrições mais relevantes do próprio regulamento seguidas de uma descrição das mesmas e um conjunto de medidas técnicas que podem ser utilizadas para alcançar o seu objetivo.

Licitude, lealdade e transparência

- *"Objeto de um tratamento lícito, leal e transparente em relação ao titular dos dados*" RGPD Art.5.1(a)
- O tratamento de dados pessoais, independentemente de quão pequeno seja, tem de assentar numa base legal sólida. Os responsáveis pelo tratamento de dados pessoais precisam fornecer uma visão clara sobre como funciona o tratamento e as consequências sobre os dados antes de recolher e tratar os mesmos.
- **Medidas técnicas**: medidas de transparência incorporadas, medidas jurídicas incorporadas, serviços de não repúdio.

Limitação das finalidades

- *"Recolhidos para finalidades determinadas, explícitas e legítimas e não podendo ser tratados posteriormente de uma forma incompatível com essas finalidades; o tratamento posterior para fins de arquivo de interesse público, ou para fins de investigação científica ou histórica ou para fins estatísticos, não é considerado incompatível com as finalidades iniciais, em conformidade com o artigo 89º, nº 1"* RGPD Art.5.1(b)
- Limitação de propósito requer que os responsáveis pelo tratamento de dados definam as razões para recolher e tratar os dados. Os responsáveis pelo tratamento de dados precisam garantir que os dados são tratados de acordo com a finalidade original. Os dados não devem ser tratados quando o objetivo for alterado sem uma base legal sólida. Além disso, a rastreabilidade dos dados em todo o seu ciclo de vida é necessária para saber quando o tratamento de dados não estiver de acordo com a finalidade original.
- **Medidas técnicas**: inventário de dados incorporado, relatórios e notificações.

Minimização dos dados

- *"Adequados, pertinentes e limitados ao que é necessário relativamente às finalidades para as quais são tratados"* RGPD Art.5.1(c)
- A minimização de dados envolve a redução da quantidade de dados recolhidos ao estritamente necessário para que a interação com o titular dos dados seja satisfeita.
- **Medidas técnicas**: Armazenamento centralizado, pseudonimização dos dados, remoção de meta-dados não utilizados, *proxies* intermediários.

Exatidão

- *"Exatos e atualizados sempre que necessário; devem ser adotadas todas as medidas adequadas para que os dados inexatos, tendo em conta as finalidades para que são tratados, sejam apagados ou retificados sem demora"* RGPD Art.5.1(d)
- Exatidão consiste em tomar as medidas necessárias para garantir a precisão dos dados obtidos assim como verificar a fonte dos mesmos. Nos sistemas de informação, a exatidão de um registo também pode ser medida pela capacidade de qualquer indivíduo entender corretamente o que o registo diz.
- **Medidas técnicas**: validação de *inputs*, tratamento de disputas de dados, limpeza dos dados.

Limitação da conservação

- *"Conservados de uma forma que permita a identificação dos titulares dos dados apenas durante o período necessário para as finalidades para as quais são tratados; os dados pessoais podem ser conservados durante períodos mais longos, desde que sejam tratados exclusivamente para fins de arquivo de interesse público, ou para fins de investigação científica ou histórica ou para fins estatísticos, em conformidade com o artigo 89º, nº 1, sujeitos à aplicação das medidas técnicas e organizativas adequadas exigidas pelo presente regulamento, a fim de salvaguardar os direitos e liberdades do titular dos dados"* RGPD Art.5.1(e)
- O princípio de limitação da conservação concentra-se em manter os dados pessoais apenas para o período em que os dados atendem a sua finalidade. Os responsáveis pelo tratamento de dados têm total responsabilidade para manter o controlo dos dados e removê-los quando não já não estiverem a ser tratados para a sua finalidade original.
- **Medidas técnicas**: rastreabilidade, limpeza automatizada, relatórios.

Segurança (Integridade e Confidencialidade)

- *"Tratados de uma forma que garanta a sua segurança, incluindo a proteção contra o seu tratamento não autorizado ou ilícito e contra a sua perda, destruição ou danificação acidental, adotando as medidas técnicas ou organizativas adequadas"* RGPD Art.5.1(f)
- Devem ser implementadas as medidas de segurança adequadas à proteção dos dados pessoais. Integridade e confidencialidade são conceitos fundamentais de segurança da informação. Proteger a privacidade do titular dos dados, mantendo a sua integridade, é manter a exatidão e consistência dos dados armazenados. Por outro lado, a confidencialidade dos dados é mantida protegendo a informação de divulgação e acesso não autorizado. As medidas para garantir este princípio devem ser implementadas e operadas durante todo o ciclo de vida dos dados.
- **Medidas técnicas**: criptografia *end-to-end, v*alidação dos dados, autenticação, autorização.

Responsabilidade

- *"O responsável pelo tratamento é responsável pelo cumprimento do disposto no nº 1 e tem de poder comprová-lo"* RGPD Art.5.2
- Os responsáveis pelo tratamento de dados devem ser responsáveis e demonstrar conformidade com as disposições dos regulamentos. A demonstração pode ser realizada de várias maneiras, desde o não processamento de dados pessoais de forma ilegal, até a implementação dos princípios de privacidade nos sistemas de informação.
- Medidas técnicas: autenticação e autorização, registos de auditoria à prova de violações, monitorização, prevenção de perda de dados (Data Loss Prevention).

Direitos dos indivíduos

Com a introdução do RGPD, é também introduzido um conjunto de direitos dos titulares dos dados que devem ser garantidos pelos responsáveis pelo tratamento. De seguida são apresentados esses direitos, assim como uma descrição dos mesmos:

- **O direito a ser informado –** Os titulares dos dados têm o direito a obter informações legíveis sobre a forma como são tratados os seus dados pessoais. Esta informação tem de ser apresentada de forma clara e num formato que não dificulte a compreensão, como por exemplo extensos documentos com Termos e Condições.
- **O direito de acesso –** Os titulares dos dados podem submeter um pedido de acesso e obter uma cópia de todos os seus dados pessoais. Os responsáveis pelo tratamento devem responder no espaço de um mês. A informação a ser fornecida deverá incluir:
    - O propósito do tratamento de dados;
    - O tipo de dados a ser tratado;

    - As entidades com quem os dados poderão ser partilhados;
    - Períodos de retenção dos dados;
    - As fontes dos dados, no caso de estes não terem sido recolhidos diretamente do titular;
    - A existência de alguma decisão tomada de forma automática, incluindo a lógica envolvida – por exemplo as questões utilizadas para a criação de um perfil.
    - A informação fornecida deverá igualmente incluir o direito ao apagamento, retificação, portabilidade e restrição de tratamento, assim como a indicação da autoridade controladora a ser contactada em caso de queixa.
- **O direito à restrição do tratamento –** Os titulares dos dados têm o direito de restringir o tratamento dos seus dados, o que significa que os responsáveis pelo tratamento continuam a armazenar os dados mas os mesmos são marcados como restritos e não podem de outra forma ser tratados sem o consentimento adicional do titular.
- **O direito à retificação –** Os titulares dos dados têm o direito de corrigir quaisquer dados pessoais que estejam incorretos. Quando notificados, os responsáveis pelo tratamento devem proceder às correções sem demora injustificada.
- **O direito à remoção (direito a ser esquecido) –** Os titulares dos dados têm o direito de, em certas circunstâncias, solicitar a remoção dos seus dados. Isto significa que os titulares dos dados podem exigir aos responsáveis pelo tratamento dos mesmos a eliminação de todos dados que lhes dizem respeito, incluindo a publicação de qualquer informação e eventualmente impedir que os dados sejam tratados por terceiros.
- **O direito à portabilidade –** Os titulares dos dados têm o direito de receber uma cópia dos seus dados pessoais num formato legível por máquina e de os transferir para qualquer outra entidade. Por exemplo, se um individuo decidir mudar de banco tem o direito de pedir a portabilidade dos seus dados de um banco para o outro, e o primeiro banco tem a obrigação de fornecer os dados num formato eletrónico.

Requisitos gerais

Para além dos requisitos impostos pelos princípios gerais de proteção de dados, e pelos direitos dos indivíduos, devem ainda ser considerados outros requisitos impostos pelo RGPD descritos de seguida:

- **Data Protection Impact Assesment (DPIA) –** Os responsáveis pelo tratamento de dados necessitam realizar uma avaliação de impacto de privacidade quando uma nova atividade de processamento possa resultar num risco para os titulares dos dados. Esta avaliação permite aos responsáveis pelo tratamento identificar e mitigar os riscos que poderiam de outra forma não ser identificados.
- **Consentimento explícito –** De acordo com o novo regulamento, as empresas devem solicitar e receber o consentimento para recolher, utilizar e mover dados pessoais. Este pedido deve ser feito de forma clara, inteligível e de fácil acesso. As pessoas devem ser capazes de retirar o seu consentimento com a mesma facilidade com que são capazes de fornecê-lo.
- **Notificações obrigatórias de violação de dados –** No caso de existir uma violação dos dados, os responsáveis pelo seu tratamento devem notificar os utilizadores, bem como as autoridades, dentro de 72 horas após tomar conhecimento da violação. Esta notificação é de extrema importância uma vez que as violações dos dados podem resultar em “riscos para os direitos e liberdades dos indivíduos".
- **Encarregados da Proteção dos Dados (Data Protection Officers – DPO) –** Ao abrigo do RGPD, os requisitos de manutenção de registos internos e a nomeação de encarregados de proteção de dados (DPOs, funcionários encarregues de gerir a proteção de dados) são

obrigatórios para grandes empresas. Os DPOs são contratados pelo seu conhecimento especializado em leis e práticas de proteção de dados.

- ***Privacy by design*** **–** Embora este não seja um novo conceito, ao abrigo do RGPD, a privacidade por *design* torna-se uma exigência legal. Isto significa que a privacidade e a proteção de dados precisaram ser consideradas criticamente desde o início e durante todo o ciclo de vida de um projeto. De acordo com o Artigo 23º do RGPD, os responsáveis pelo tratamento dos dados devem apenas manter e processar dados que sejam absolutamente necessários para que um projeto seja concluído. Além disso, o acesso aos dados deve ser limitado apenas ao pessoal encarregado do seu tratamento.

Capítulo 3

Análise de Requisitos

Neste capítulo são apresentados os requisitos identificados como sendo necessários para sistemas que preservam a privacidade dos seus utilizadores.

A proteção da privacidade tem como objetivo fornecer propriedades abstratas, ou seja, independentes do contexto, aos sistemas de informação [17]. Os três conceitos, confidencialidade, integridade e disponibilidade, são comummente aceites como sendo os conceitos fundamentais na proteção de sistemas de informação. Estes conceitos permanecem estáveis à décadas e servem como base para inúmeras metodologias de segurança. Para além disso, têm sido alvo de propostas para diversas extensões e refinamentos.

Como complemento destes três pilares da segurança, foram propostos em 2009 [18] três objetivos específicos para lidar com questões de privacidade, nomeadamente desacoplamento, transparência e interveniência. Estes conceitos foram posteriormente refinados, em [19, 20], e incorporados num modelo proposto para ser utilizado a nível Europeu [21, 22], e são descritos de seguida:

- **Desacoplamento –** garante que os dados relevantes relacionados com privacidade não podem ser cruzados entre diferentes domínios, o que significa que os diferentes processos devem ser operados de forma a que os dados tratados não possam, de forma alguma, ser relacionados com qualquer outro conjunto de dados fora do domínio. O desacoplamento está relacionado com os princípios de limitação das finalidades e minimização dos dados (princípios descritos na **secção 2.4.4**). Os mecanismos que permitem alcançar ou suportam o desacoplamento compreendem a minimização dos dados, separação de contextos (separação física, criptografia, utilização de identificadores diferentes, controlo de acessos), anonimização e pseudonimização, e remoção antecipada dos dados.
- **Transparência –** garante que todo o tratamento de dados pessoais, incluindo os aspetos legais, técnicos e organizacionais desse mesmo tratamento, é compreendido a qualquer momento pelos titulares. Esta informação tem de estar disponível antes, durante e após o tratamento ocorrer. A quantidade de informação fornecida, e o modo como deve ser comunicada, irá depender do seu destinatário (por exemplo o responsável pelo tratamento, o utilizador, um auditor interno ou a autoridade controladora). A transparência está relacionada com o princípio de licitude, lealdade e transparência e é um pré-requisito para o princípio da responsabilização (princípios descritos na **secção 2.4.4**). Os mecanismos que permitem alcançar ou suportam a transparência compreendem *logging* e reporte, documentação compreensível, responsabilidades, código fonte, políticas de privacidade, notificações e restante informação e comunicação aos titulares dos dados.
- **Interveniência –** garante que é possível existir uma intervenção no que diz respeito a todo o tratamento de dados pessoais, ou planeamento do mesmo, em particular pelos titulares dos dados. A interveniência está relacionada com os princípios relativos aos direitos dos indivíduos, por exemplo o direito à retificação e remoção, o direito de retirar consentimento (o consentimento não só deve ser explicito como pode ser retirado) ou o direito à restrição do tratamento (direitos dos indivíduos descritos na **secção 2.4.4**). Além disso, a interveniência é importante para outras partes interessadas, por exemplo os responsáveis pelo tratamento dos

dados, na medida em que permite que estes influenciem ou terminem o tratamento dos dados em qualquer momento. Os mecanismos que permitem alcançar ou suportam a interveniência compreendem processos estabelecidos para terminar ou influenciar o tratamento de forma total ou parcial, alterar manualmente uma decisão automatizada, políticas *break the glass*, pontos únicos de contacto para os pedidos de intervenção por parte dos titulares, funcionalidades para os utilizadores alterarem as suas definições (por exemplo, alterar a configuração de perfil), ou desativar um sistema automático ou de monitorização por um determinado período de tempo.

Alcançar os objetivos de proteção necessários significa encontrar o equilíbrio entre os requisitos derivados dos 6 conceitos fundamentais (integridade, confidencialidade, disponibilidade, desacoplamento, transparência e interveniência) relativos aos dados e aos processos técnicos e organizacionais de tratamento. Considerações quanto a licitude, lealdade e responsabilização fornecem orientação para tomar decisões de *design* e aplicar as proteções apropriadas.

3.1 Desenho de uma solução segura

A arquitetura de software envolve um conjunto de decisões significativas sobre a forma como se organiza determinado sistema de software. Incluí a seleção dos componentes que constituem a solução, as interfaces pelas quais um sistema é composto a partir desses componentes, o comportamento esperado quando os diferentes componentes colaboram, a composição destes elementos estruturais e comportamentais num subsistema maior, e o estilo arquitetural que guia a organização dos diversos elementos [17].

Existem diversas metodologias para o desenvolvimento de software, e para garantir a incorporação do conceito de *Privacy by Design* durante as diversas fases é necessário considerar diferentes aspetos. Na fase de levantamento de requisitos é necessário definir as estratégias de *Privacy by Design*, durante a fase de desenho é útil recorrer-se a padrões de *design* conhecidos, e finalmente, durante a implementação são aplicadas tecnologias concretas que permitem uma melhoria da privacidade.

3.1.1 Padrões e estratégias de design

A utilização de padrões de *design* é útil para tomar decisões relativas à forma como deve ser organizado um sistema de software [17]. Um padrão de *design* "fornece um esquema para o refinamento dos subsistemas ou componentes de um sistema de software, ou as relações existentes entre si. Descreve uma estrutura comum e recorrente entre componentes comunicantes que resolve um problema de *design* geral dentro de um contexto particular" [23].

Os padrões de *design* permitem a quem desenha um sistema resolver um problema ao dividir o mesmo em sub-problemas, mais pequenos e de fácil gestão, sem preocupação com os detalhes da implementação. Ao mesmo tempo, o padrão descreve claramente as consequências da subdivisão proposta, permitindo determinar se a aplicação do padrão permite alcançar o objetivo geral. A descrição de um padrão de *design* [24] tipicamente contém os seguintes elementos: o seu nome, propósito, contexto (a situação em que se aplica), implementação (a sua estrutura, componentes e as relações entre si), e as suas consequências (os seus resultados, efeitos laterais e contrapartidas).

Muitos dos padrões de *design* existentes são bastante específicos, e deste modo não podem ser aplicados diretamente na fase de levantamento de requisitos. No entanto, existem padrões de *design* que representam um maior nível de abstração, chamados padrões de arquitetura de software [17]. Estes padrões expressam uma organização ou esquema fundamental para os sistemas de software, fornecendo

um conjunto de subsistemas predefinidos que especificam as suas responsabilidades, regras e orientações para a organização das relações entre sistemas de software.

O Model-View-Controller é considerado um padrão de arquitetura de software, no entanto, a distinção entre padrão de arquitetura e de *design* nem sempre é linear. A realidade é que se pode olhar para o desenho de um sistema a diversos níveis de abstração, e os padrões de *design* concretos são demasiado específicos para serem aplicados em certos casos. Existem princípios mais gerais que orientam a arquitetura dos sistemas, considerando diferentes níveis de abstração são considerados estratégias de *design*. [25].

De um modo geral, uma estratégia de *design* com foco na privacidade é uma estratégia que atinga um determinado nível de proteção da privacidade. Apesar de limitarem as possibilidades, as estratégias de *design* não impõem necessariamente uma estrutura específica para o sistema. Deste modo, as estratégias são também aplicáveis durante a fase de levantamento de requisitos. De notar que a noção de estratégia de *design* de privacidade não deve ser confundida com os princípios de privacidade, descritos na **secção 2.4.4**.

3.1.2 Estratégias de design de privacidade

De seguida são descritas 8 estratégias de *design* de privacidade [25], derivadas dos princípios legais da legislação de proteção de dados. No que diz respeito à natureza das diferentes estratégias, é possível distinguir entre estratégias orientadas aos dados e estratégias orientadas aos processos.

Estratégias orientadas aos dados

- **Estratégia #1 – MINIMISE –** A estratégia de *design* de privacidade que pode ser considerada a mais básica é MINIMISE, que dita que a quantidade de dados pessoais a serem tratados deverá ser restringida ao mínimo possível. Esta estratégia é descrita de forma extensa em [26]. Ao assegurar que não são recolhidos dados pessoais, ou que não são recolhidos mais dados que aqueles estritamente necessários, o possível impacto de privacidade para um sistema é limitado. Aplicar a estratégia MINIMISE significa que é necessário explicitar que o tratamento de dados pessoais é proporcional ao seu propósito, e que não existem outros meios, menos invasivos, para atingir o mesmo propósito. A decisão de recolher dados pessoais pode ser feita de várias formas, e pode ter lugar nas fases de desenvolvimento ou em tempo de execução. Por exemplo, poderá ser tomada a decisão de não ser recolhida qualquer informação sobre um determinado sujeito, ou recolher apenas um conjunto limitado de atributos.
- **Estratégia #2 – HIDE –** A segunda estratégia de *design* (HIDE) dita que quaisquer dados pessoais, e as relações entre si, devem ser escondidas. O racional que suporta esta estratégia é que ao esconder os dados pessoais, estes não podem ser facilmente utilizados para fins impróprios. A estratégia em si não determina de quem devem, os dados, ser escondidos. Isto dependerá do contexto específico no qual a estratégia é aplicada. Em certos casos, quando a estratégia é usada para esconder informação que emerge da utilização de um sistema, em comunicações por exemplo, a intenção é esconder a informação de qualquer outra parte que não participe na comunicação. Em outros casos, em que a informação é tratada de forma legítima por alguma parte, a intenção é esconder a informação de qualquer outra parte. Esta estratégia corresponde a garantir a confidencialidade, e pretende atingir desacoplamento e inobservância [27].
- **Estratégia #3 – SEPARATE –** A terceira estratégia de *design* (SEPARATE) dita que os dados pessoais devem ser tratados de forma distribuída, ou seja, em compartimentos separados sempre que possível. Ao separar o processamento ou armazenamento de diversas fontes de dados

pessoais que pertencem ao mesmo indivíduo, deixa de ser possível criar perfis completos sobre o mesmo. Para além disso, a separação é um bom meio para atingir a limitação de propósito. Esta estratégia favorece o tratamento em separado em vez de soluções centralizadas. Em particular, dados provenientes de diferentes fontes deverão ser armazenados em diferentes bases de dados, e essas bases de dados não devem ser interligadas. O processamento dos dados deverá ser feito localmente sempre que possível, e o mesmo para o seu armazenamento. As tabelas nas bases de dados deverão ser separadas sempre que possível, e as respetivas linhas das tabelas deverão ser difíceis de interligar, por exemplo, ao remover quaisquer identificadores, ou ao utilizar pseudónimos específicos por tabela.

- **Estratégia #4 – AGGREGATE –** A quarta estratégia de *design* (AGGREGATE) dita que os dados pessoais devem ser tratados no maior nível de agregação possível e com o menor nível de detalhe possível no qual são ainda relevantes. A agregação da informação em grupos de atributos ou grupos de indivíduos permite restringir o nível de detalhe existente nos dados pessoais. Desta forma os dados tornam-se então menos sensíveis, uma vez que os dados são gerais o suficiente, e a informação armazenada é válida para um elevado número de indivíduos, sendo que pouca informação poderá ser atribuída a um determinado individuo, protegendo a sua privacidade.

Estratégias orientadas a processos

- **Estratégia #5 – INFORM –** A estratégia INFORM corresponde a noção importante de transparência, o que significa que os titulares dos dados devem ser devidamente informados quando existir o tratamento de dados pessoais. Sempre que os titulares dos dados utilizam um sistema, estes devem ser informados sobre a informação que é processada, a forma e o propósito do tratamento. Também deverá ser incluída informação sobre os métodos utilizados para a proteção dos dados, ou seja, deverá ser transparente o nível de segurança do sistema. Os titulares dos dados deverão também ser informados sobre terceiros com quem a informação seja partilhada e sobre os seus direitos de acesso e como exercê-los.
- **Estratégia #6 – CONTROL –** A estratégia CONTROL dita que os titulares dos dados deverão ter controlo sobre o tratamento dos seus dados pessoais. Esta estratégia é de facto uma contrapartida importante da estratégia INFORM. Se não existirem os meios adequados de controlar a utilização dos dados pessoais, não existe grande propósito em informar os titulares sobre o facto dos dados serem recolhidos. O contrário também se aplica, sem a informação adequada não existe propósito em ser pedido o consentimento para o tratamento. A legislação existente sobre a proteção de dados garante aos titulares o direito de visualizar, atualizar e requisitar o apagamento dos dados pessoais que lhe dizem respeito. Esta estratégia reforça este facto, e os padrões de *design* associados garantem aos utilizadores as ferramentas para exercer os seus direitos.
- **Estratégia #7 – ENFORCE –** A sétima estratégia (ENFORCE) dita que deverá ser imposta uma política de privacidade compatível com os requisitos legais. Esta estratégia está relacionada com o princípio da responsabilidade. O ENFORCE garante que existe uma política de privacidade, o que representa um passo importante para garantir que um sistema respeita a privacidade dos seus utilizadores durante a sua operação. O nível de privacidade irá, portanto, depender da política imposta, que deverá no mínimo ser compatível com os requisitos legais aplicáveis. Como resultado, a limitação de finalidades irá também ser abrangida por esta estratégia. Em ultima análise, o mais importante é a necessidade de ser de facto imposta a política, o que implica que no mínimo sejam implementados os mecanismos técnicos de proteção apropriados para prevenir violações à política de privacidade.

- **Estratégia #8 – DEMONSTRATE –** A estratégia DEMONSTRATE dita que o responsável pelo tratamento dos dados deverá ser capaz de demonstrar a conformidade para com a respetiva política de privacidade e quaisquer requisitos legais aplicáveis. Esta estratégia suporta o princípio da responsabilidade. A estratégia DEMONSTRATE vai mais longe que a estratégia ENFORCE na medida em que requer que o responsável pelo tratamento dos dados prove que está de facto no controlo, ou seja, demonstre que a política de privacidade está efetivamente implementada nos sistemas de informação. No caso de existirem problemas ou reclamações, o responsável pelo tratamento deverá ser capaz de determinar a dimensão de quaisquer possíveis fugas de dados.

3.1.3 Privacy-enhancing technologies

Quando se fala em *Privacy by Design*, o conceito de Privacy-Enhancing Technologies (PETs) é bastante utilizado [28, 29].

Privacy-Enhancing Technologies é um sistema que aplica medidas para proteção de informação privada ao eliminar ou minimizar dados pessoais e deste modo prevenir acessos desnecessários ou tratamento não autorizado de dados pessoais, sem que seja prejudicada a funcionalidade do sistema de informação. Por princípio, PETs são utilizadas para implementar um determinado padrão de *design* recorrendo a uma tecnologia concreta, como por exemplo anonimização ao nível da aplicação, websites que optam por não rastrear qualquer informação sobre o utilizador ou ferramentas criptográficas que protegem a informação transmitida na Internet.

3.2 Componentes de uma solução segura

3.2.1 Autenticação

Funcionalidades de privacidade dos protocolos de autenticação

A autenticação dos utilizadores é o processo pelo qual os utilizadores demonstram a sua identidade perante um sistema e obtêm acesso a informação confidencial ou ações com um determinado nível de privilégio [17]. Assim que esta ligação é estabelecida, as comunicações podem prosseguir na base de que ambas as partes conhecem as respetivas identidades. Mecanismos de autenticação são a base de segurança dos sistemas e são normalmente o primeiro passo para a utilização de serviços remotos ou garantir o controlo de acessos. Autenticação forte também poderá ser um mecanismo para garantir que apenas um determinado sujeito, ou parte autorizada, tem acesso a informação privada.

Durante o processo de autenticação de um determinado utilizador, e dependendo do modo específico como este é feito, poderão levantar-se algumas questões relativas à privacidade do utilizador [17]. No caso de autenticação que recorra a protocolos de rede, poderão existir entidades maliciosas que, ao observar passivamente a rede, tentem identificar ou rastrear os utilizadores que se autenticam. Poderá também acontecer, um utilizador que se tente autenticar num sistema errado controlado por entidades maliciosas, e como resultado acabar por fornecer a sua identidade, e possivelmente as suas credenciais (ataque conhecido como *phishing*). Por fim poderá também acontecer que, após uma autenticação bem-sucedida, uma entidade maliciosa que observe passivamente a comunicação entre ambas as partes, utilize os metadados da sessão para inferir a identidade das partes comunicantes.

Os protocolos de autenticação considerados estado da arte fornecem proteções contra os ataques identificados. Em particular, previnem terceiros de inferir a identidade das partes que se autenticam,

aplicam controlos relativos a *phishing* e não permitem que sejam inferidas as identidades durante sessões seguras.

Um exemplo de um protocolo considerado estado da arte é o protocolo Just Fast Keying (JFK) [30]. Este protocolo existe em duas variantes, JDKi e JDKr, que protegem a identidade do *initiator* e *responder* respetivamente, através da utilização de certificados e criptografia de chave pública. A autenticidade, integridade e confidencialidade das futuras mensagens é garantida através da utilização de chaves diferentes em cada sessão (*forward secrecy*). Uma ilustração do protocolo em questão é apresentada na **Figura 3.1**.

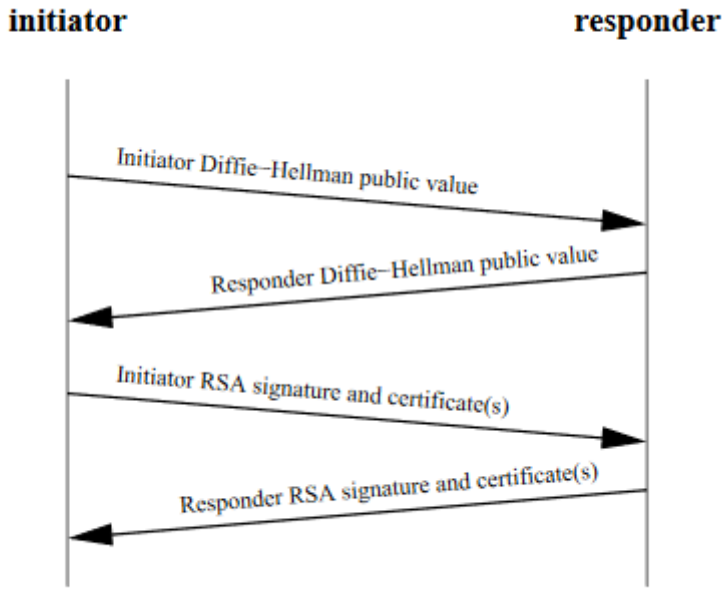

Figura 3.1 *– Ilustração do protocolo Just Fast Keying. Obtido a partir de [30].*

Formas mais fracas de autenticação são comummente utilizadas em serviços web, e baseiam-se em fornecer um *username* e *password* sobre uma ligação cifrada (por exemplo recorrendo ao protoloco TLS). A vantagem da utilização deste mecanismo é que permite recorrer a tecnologias web standard, incluindo HTTPS e formulários web. Também permite esconder a identidade do cliente que se autentica, no entanto, não permite por definição esconder a identidade do servidor de um observador passivo. Para além disso, simplifica a possibilidade de serem realizados ataques de *phishing* e, em alguns casos, pode ainda ser possível reverter a sessão para o protocolo HTTP, não cifrado, divulgando a identidade do utilizador autenticado e os respetivos cookies que podem ser utilizados para personificação. Deste modo, este mecanismo deve ser utilizado com precaução, e todas as interações deverão ser devidamente cifradas.

Privacidade na gestão de identidades federadas e single-sign-on

Sistemas federados de gestão de identidades permitem a diferentes entidades que registam utilizadores a identificação dos mesmos a partir de um único processo de autenticação. Nestes sistemas, um indivíduo pode registar-se como utilizador de um serviço A com uma certa identidade, e autenticar utilizando essa mesma identidade em um serviço terceiro B. De facto, o serviço B poderá permitir a autenticação a partir de um qualquer número de serviços, aceitando as identidades fornecidas pelos mesmos, desde que exista uma relação de confiança entre eles. Uma ilustração de um sistema federado de gestão de identidades é apresentada na **Figura 3.2**.

Este tipo de sistemas permite aos utilizadores acederem a diferentes aplicações, relacionadas, passando uma única vez pelo processo de autenticação, e desta forma mitigando os riscos da utilização de diversas combinações de credenciais, por outro lado levanta algumas questões relativas a privacidade. As implementações iniciais de single-sign-on não permitiam ao utilizador escolher os dados que eram transferidos entre cada aplicação que o utilizador visitava. No contexto de uma única organização, este não seria um problema, no entanto, à medida que serviços como Active Directory Federation Services proliferaram, a informação dos utilizadores começou a sair do controlo das entidades que recolheram os dados. Com a introdução do RGPD, considerar esta questão tornou-se um requisito legal, e métodos como o OpenID Connect tornaram-se mais atrativos.

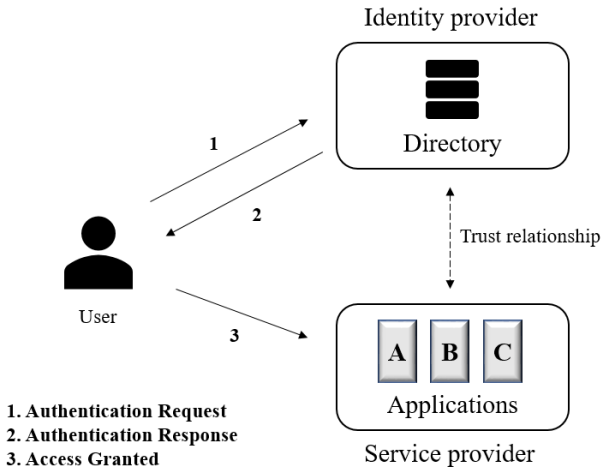

Figura 3.2 *– Ilustração de gestão de identidades federadas e single-sign-on. Obtido a partir de [31].*

No que diz respeito a serviços de *single-sign-on* (SSO), existe um número elevado de serviços fornecidos, com propriedades de privacidade variadas. Um sistema federado de gestão de identidades com *single-sign-on*, com funcionalidades de privacidade avançadas, é desenhado para permitir aos seus utilizadores fornecer apenas um subconjunto de atributos a um fornecedor de serviços após autenticação. Até mesmo o identificador do utilizador poderá não ser fornecido dependendo da necessidade de identificar univocamente os utilizadores. Deste modo é possível incorporar e suportar o princípio de minimização dos dados.

Esta capacidade de permitir a divulgação seletiva de atributos, incluindo anonimidade parcial, é considerada o estado da arte para sistemas federados de gestão de identidades. No entanto, é importante notar que as proteções de privacidade oferecidas por estes protocolos são apenas robustas quanto a proteger os utilizadores dos fornecedores de serviços. Um observador passivo, que intercete a comunicação, poderá ser capaz de inferir quem são os utilizadores assim com os seus atributos. Para além disso, quem fornece a identidade é ciente de todas as sessões de autenticação que o utilizador participe, e consequentemente os serviços a que este acede. Protocolos com entidades, incluindo aquelas que fornecem identidades, que permitam a observação dos detalhes das sessões de autenticação devem ser evitados, uma vez que podem ser utilizados para monitorização abusiva. De um modo geral, e quando possível, devem ser adotadas técnicas que permitam a proteção dos utilizadores de observadores passivos assim como de fornecedores de identidades abusivos.

3.2.2 Comunicações seguras

A maior parte das ligações de rede físicas fornecem fracas garantias de confidencialidade e privacidade [17]. As redes locais assentam cada vez mais em tecnologias sem fios, e nas redes de longa distância, onde é impossível garantir a segurança contra observadores passivos. Qualquer informação transferida entre utilizadores e serviços ou entre utilizadores, deverá ser cifrada utilizando as técnicas mais recentes para o efeito e garantindo que esta é ininteligível para observadores passivos. Todos os tipos de comunicação dos utilizadores devem ser protegidos, dados pessoais ou *inputs* sensíveis introduzidos pelos utilizadores devem ser cifrados para preservar a sua privacidade, no entanto, qualquer acesso a recursos públicos deve ser obscurecido através de criptografia para prevenir um observador passivo de inferir os padrões de navegação dos utilizadores, definir perfis, utilização de serviços ou extração de identificadores que possam ser utilizados para rastreamento futuro [32].

Criptografia básica entre clientes e serviços

A utilização de canais cifrados fornece um elevado nível de confidencialidade para as comunicações entre clientes e serviços, desde que implementados e configurados de forma correta. Uma grande maioria das tecnologias existentes recorrem ao protocolo Transport Layer Security (cuja versão mais recente é TLS 1.3 [33], apesar de ser aceite a versão 1.2 [34]), assim como o protocolo Secure Shell (SSH) [35]. Estas tecnologias fornecem um canal confidencial, e possivelmente autenticado, entre clientes e serviços.

Tanto o TLS como o SSH utilizam criptografia de chaves públicas que permitem que sejam estabelecidos canais cifrados sem que os clientes e servidores partilhem nenhum segredo anterior. No entanto, o protocolo TLS depende de uma infraestrutura de chaves públicas e de uma autoridade certificadora para garantir a autenticidade do servidor. Deste modo, autoridades comprometidas podem levar ao comprometimento dos canais de comunicação. Para recursos públicos acessíveis a partir de web-browsers é necessário um certificado de chaves públicas válido. O protocolo SSH depende de verificações de chaves manuais, o que poderá trazer dificuldades para utilizadores não técnicos e, como tal, deverá ser utilizado apenas a confirmação de que os utilizadores são capazes de fazer a verificação de forma correta e consistente.

Pode ser usado um elevado número de diferentes tecnologias para a comunicação de forma a proteger os dados dos utilizadores em trânsito na rede. Um exemplo é o IPSec [36] que cria um canal de comunicação seguro entre máquinas ligadas por rede, ou entre redes ligadas por ligações públicas. É recomendado que o tráfico interno a uma organização (LAN – Local Area Network) seja cifrado caso contenha informação dos utilizadores, tal como é o caso na realização de back-ups ou na comunicação entre aplicações e servidores de bases de dados. Ligações através de redes sobre áreas amplas (WAN – Wide Area Network), sejam elas públicas ou privadas, devem ser sempre cifradas uma vez que é muito difícil garantir a segurança física de tais canais. Existe um elevado número de tecnologias Virtual Private Network (VPN) para a proteção de tais ligações [37, 38].

Criptografia básica end-to-end

Serviços como Voice-over-Internet-Protocol (VoIP), correio eletrónico e mensagens instantâneas servem como meio de comunicação entre utilizadores finais. Tais serviços devem cifrar a comunicação de uma forma *end-to-end*, ou seja, a comunicação é cifrada num dos *end-points* e decifrada no outro, fazendo com que o conteúdo de tal comunicação seja ininteligível para qualquer entidade terceira, incluindo o fornecedor de serviço. Enquanto que o fornecedor possa assistir os utilizadores no processo de autenticação, permitindo o estabelecimento de tal canal de comunicação, é preferível que, de uma perspetiva de privacidade, as chaves utilizadas para proteger a confidencialidade e integridade dos dados

nunca estejam disponíveis para os fornecedores de serviços. É por este motivo que se deve recorrer a criptografia end-to-end, ou seja, criptografia assimétrica, também conhecida como criptografia de chave pública.

Este tipo de serviço poderá necessitar de alguma visibilidade sobre o conteúdo das comunicações, seja para efeitos de *routing* ou para acrescentar valor ao serviço. Nestes casos, o menor número de informação possível deverá ser exposto ao fornecedor e, na maior medida possível, qualquer manipulação do conteúdo da comunicação devera ser feita nos dispositivos dos utilizadores finais, minimizando a informação divulgada ao fornecedor.

Várias tecnologias foram propostas, implementadas e normalizadas em vários aspetos para fornecer confidencialidade *end-to-end*. O software Pretty Good Privacy (PGP) [39] e a norma S/MIME [40] podem ser utilizados para proteger a troca de emails num formato *end-to-end*. Os protocolos Off-The-Record (OTR) [41] podem ser utilizados para proteger as conversações com base em mensagens instantâneas, para além disso, existem diversas aplicações móveis que fornecem comunicações cifradas *end-to-end*, como por exemplo Status ou Adamant [42, 43].

Rotação de chaves, *forward secrecy* e resistência à coação

As tecnologias modernas de criptografia fornecem elevadas garantias de confidencialidade, mas apenas tendo em conta que as chaves secretas utilizadas não são comprometidas [17]. Deste modo, as chaves criptográficas tornam-se alvos valiosos para roubo ou extorsão.

Para minimizar o potencial comprometimento das chaves estas devem ser rodadas com frequência, deste modo minimizando a exposição de dados privados em caso de comprometimento. Os sistemas modernos de criptografia, incluindo algumas configurações de TLS, fornecem *forward secrecy*, uma propriedade que garante que são utilizadas novas chaves em cada sessão de comunicação, e que as mesmas são descartadas quando a ligação é terminada.

Anonimidade e pseudonimização nas comunicações

Criptografia *end-to-end* é utilizada para proteção do conteúdo das comunicações, no entanto, existem metadados que ficam expostos a entidades terceiras. Estes metadados podem ser informação sobre a comunicação, como por exemplo os intervenientes, a hora ou volume das mensagens, a duração das sessões, a localização e possivelmente a identidade dos *end-points*.

Do ponto de vista da privacidade, a exposição destes metadados poderá ter um impacto elevado [17]. Por exemplo, descobrir o facto de que um jornalista está a comunicar com alguém de uma organização poderá comprometê-lo como fonte jornalística, mesmo que os detalhes da conversa sejam confidenciais.

Os sistemas que escondem os metadados das comunicações são conhecidos como sistemas de comunicações anónimas [44]. Estes sistemas podem fornecem um conjunto de propriedades de privacidade para além de simples anonimidade. Um sistema pode fornecer anonimidade do remetente no caso de esconder a identidade de quem envia uma mensagem ou inicia uma ligação de rede. Por outro lado, pode fornecer anonimidade do destinatário se for possível estabelecer o contacto do individuo ou serviço sem que se conheça o seu endereço de rede ou físico. A primeira propriedade é útil para que os utilizadores possam aceder a serviços de forma anónima, a segunda propriedade é útil para que se possam correr serviços sem expor metadados sobre o *host* ou operadores.

A anonimidade do remetente e destinatário garante que qualquer informação pessoal é escondida tanto de entidades terceiras como dos respetivos parceiros de comunicação. Outra propriedade é anonimidade de entidades terceiras, que garante que os metadados não são revelados a terceiros,

enquanto que os intervenientes na comunicação conhecem, com elevada probabilidade de certeza, as identidades de cada um. Anonimidade de terceiras partes é o mais próximo dos canais seguros tradicionais, e simplesmente aumenta a proteção na medida em que esconde os metadados.

Variações de anonimidade de terceiros incluem canais de comunicação com base em pseudónimos. Tais sistemas escondem a identidade do remetente e destinatário, mas associam a cada utilizador um pseudónimo estável que pode ser usado para controlar os abusos ou *spam*. É importante notar que ao utilizar um pseudónimo por demasiado tempo, este pode tornar-se tão revelador com a própria identidade, uma vez que os utilizadores realização ações que podem ser ligadas e divulgam uma crescente quantidade de dados pessoais ao longo do tempo. É, portanto, uma boa prática permitir aos utilizadores alterar o seu pseudónimo ou ter controlo sobre um maior número de pseudónimos de cada vez [17].

3.2.3 Armazenamento seguro

Armazenamento seguro refere-se à capacidade de armazenar os dados sem que ninguém consiga aceder-lhes, impedindo o tratamento dos mesmos por qualquer entidade para além daquela que os armazenou (responsável pelo tratamento) ou por alguém que o responsável autorize [17]. O maior desafio para implementar privacidade do armazenamento é a prevenção do acesso aos dados por parte de entidades não autorizadas. Se o armazenamento dos dados for local, então esta prevenção pode ser feita com controlos de acesso físicos ao local, no entanto, se o equipamento onde é feito o armazenamento estiver ligado a uma rede é necessário também prevenir os acessos feitos remotamente. Se o armazenamento for feito na *cloud* medidas de controlo de acessos físico não são possíveis. Assim sendo, são necessárias medidas de prevenção orientadas à tecnologia sob a qual é realizado o armazenamento.

Controlos do sistema operativo

A autenticação e listas de controlo de acessos geridas pelos sistemas operativos são a forma mais comum de garantir algum nível de privacidade no armazenamento [17]. No entanto, se um atacante ganhar acesso físico ao computador, ou for capaz de passar por cima dos controlos do sistema operativo, poderá ter acesso aos dados.

Armazenamento local cifrado

Armazenar os dados num formato cifrado é uma opção direta para garantir a privacidade dos mesmos. As opções são cifra de disco total (FDE) ou cifra ao nível do sistema de ficheiros (FSE). No caso da cifra de disco total, todo o conteúdo do disco é cifrado, incluindo os programas que fazem *boot* ao próprio sistema operativo. Para tal pode ser utilizado software ou hardware de cifra. Em contraste, no caso da cifra ao nível do sistema de ficheiros apenas o conteúdo dos ficheiros é cifrado, mas não os metadados sobre o sistema de ficheiros em si, como a estrutura do diretório, nome ou tamanho dos ficheiros. Este tipo de cifra oferece proteção insuficiente no caso dos metadados em si necessitarem de se manter confidenciais.

Uma propriedade desejável no armazenamento cifrado é *forward secrecy*. Isto significa que caso uma chave de longo termo seja comprometida, as chaves derivadas dessa chave de longo termo, utilizadas para cifrar os dados, não devem ficar também comprometidas. Utilizar um algoritmo não determinístico para derivar chaves criptográficas a partir da chave de longo termo é uma forma de alcançar *forward secrecy*.

Dois exemplos de software que pode ser utilizado para cifra de disco são o Bitlocker [45] e o Symantec Endpoint Encryption [46].

Format-preserving encryption

Criptografia que preserva o formato (FPE) [47] cifra texto em claro, num formato específico, em texto cifrado de formato idêntico. Por exemplo, um número de cartão de crédito é cifrado num outro número válido, uma palavra em inglês é cifrada numa outra palavra em inglês, etc. Uma ilustração de criptografia que preserva o formato é apresentada na **Figura 3.3**.

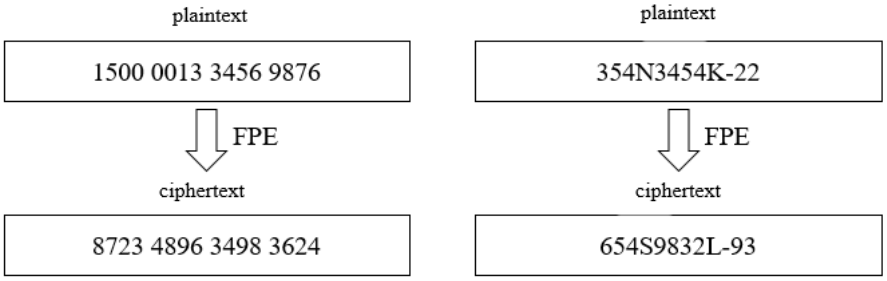

Figura 3.3 *– Ilustração de Format-preserving encryption. Adaptado a partir de [47].*

Uma motivação para este tipo de cifra é integrar a criptografia em aplicações existentes que tenham os seus modelos de dados bem definidos para os valores dos vários atributos nas bases de dados. Conceptualmente, este tipo de cifra pode ser visto como uma permutação aleatória do domínio do texto em claro, uma vez que o domínio do texto cifrado é exatamente o mesmo do texto em claro. No entanto, para domínios alargados, é inviável gerar e lembrar uma permutação realmente aleatória. Deste modo, o problema da FPE é gerar uma permutação pseudoaleatória a partir de uma chave secreta, de tal forma que o tempo de computação necessário para um único valor seja reduzido.

Armazenamento estenográfico

Sistemas de ficheiros estenográficos [48] permitem o armazenamento de informação privada enquanto escondem a existência dessa mesma informação. Este mecanismo fornece aos utilizadores um elevado nível de privacidade uma vez que os dados são estenograficamente incorporados em dados visíveis, tornando-se desta forma ocultos.

Um sistema de ficheiros estenográfico permite a um utilizador aceder a um ficheiro conhecendo o seu nome e *password*, mas um atacante que não possua esta informação não será capaz de conhecer a localização do ficheiro, mesmo que com acesso total a todo o software e hardware. Deste modo, armazenamento estenográfico é especialmente atrativo em ambientes *cloud*, nos quais o software e hardware não estão sob o controlo do utilizador.

As contrapartidas do armazenamento estenográfico são ineficiência de espaço, uma vez que são necessários muitos dados para esconder a informação real, e possível perda de dados devido a colisões de dados ou perda da chave.

TrueCrypt [49] suporta armazenamento estenográfico ao permitir vários volumes de dados escondidos, duas ou mais *passwords* dão acesso a diferentes volumes num mesmo ficheiro, mas apenas um dos volumes contém realmente a informação relevante.

Armazenamento remoto seguro

Armazenamento cifrado ou estenográfico também pode ser utilizado para alcançar a privacidade do armazenamento em configurações remotas, tais como a *cloud* [17]. No entanto, nestes casos, as operações de proteção dos dados têm de ser realizadas localmente, uma vez que as chaves utilizadas devem permanecer sob o controlo do utilizador para que a privacidade do armazenamento seja alcançada.

Pesquisa em dados cifrados

Enquanto que a cifra de dados, local ou remota, é uma boa forma de atingir a privacidade do armazenamento, significa sacrificar funcionalidades em prol da segurança e privacidade [17]. Por exemplo, se um utilizador desejar consultar todos os documentos que contenham uma determinada *keyword*, poderá ser necessário decifrar todos os documentos e então procurar pela *keyword* nos documentos em claro. Se se tratar de um ambiente *cloud*, será necessário acrescentar ainda o tempo de *download* ao tempo de decifra, o que poderá revelar-se uma tarefa demasiado pesada.

A capacidade de pesquisar em dados cifrados é especialmente atrativa em ambientes *cloud*, uma vez que o utilizador pode delegar esta tarefa ao sistema *cloud*, que por norma possui um maior poder computacional e armazena os dados cifrados localmente. Nos casos em que o servidor de armazenamento não for de confiança, e o utilizador desejar cifrar a informação localmente, poderão ser utilizadas técnicas para que o servidor seja capaz de determinar, com uma certa probabilidade, se cada documento possui a *keyword* sem que aprenda algo para além disso.

Privacidade nas bases de dados

O significado de privacidade em bases de dados é altamente dependente do contexto em que este conceito é utilizado [17]. Por norma, refere-se à privacidade dos correspondentes aos quais as entradas na base de dados diz respeito. Poderá também considerar-se a privacidade das próprias bases de dados em si, no contexto de diversas bases de dados possuídas por diversas organizações que colaboram. Por exemplo nos cuidados de saúde, ambos os requisitos acima descritos são necessários, uma vez que é necessário garantir a privacidade dos pacientes ao mesmo tempo que os registos médicos não devem ser transferidos entre um hospital e, por exemplo, uma companhia de seguros. No contexto de bases de dados interativas, que permitam *queries*, e em particular nos motores de busca, uma preocupação é também a privacidade das *queries* submetidas pelos utilizadores, especialmente após o incidente em que foram divulgadas 36 milhões de *queries* feitas por 657000 utilizadores [50].

Deste modo é possível dividir a privacidade das bases de dados em três dimensões:

- **Privacidade dos correspondentes** – prevenir a identificação dos titulares aos quais os registos na base de dados correspondem (por exemplo indivíduos ou organizações). Por norma a privacidade dos utilizadores só se torna uma preocupação se a base de dados se tornar disponível para entidades terceiras, como por exemplo investigadores ou em ultimo caso o público em geral.
- **Privacidade das bases de dados** – está relacionada sobre o facto de duas ou mais entidades autónomas poderem fazer *queries* através das respetivas bases de dados de forma a que apenas os resultados das *queries* sejam revelados (caso exista autorização para tal).
- **Privacidade dos utilizadores** – garantir a privacidade das *queries* efetuadas em bases de dados interativas, prevenindo a definição de perfis e identificação dos utilizadores.

3.2.4 Gestão de identidades

A gestão de identidades e acessos (IAM) é uma componente chave da infraestrutura de IT de uma organização [51]. É um processo funcional que compreende diversas medidas de segurança tradicionais fornecendo proteção contra acessos não autorizados aos sistemas, e pode ser automatizado para uma maior eficiência e eficácia. Sem a existência de um componente de IAM apropriado, é muito difícil provar, controlar e monitorizar quais os utilizadores que acedem a determinada informação, assim como determinar se esses acessos estão em conformidade com os regulamentos internos ou externos.

Para garantir uma gestão de identidades que mantenha a privacidade dos utilizadores de um determinado sistema, é necessário garantir uma infraestrutura segura com a habilidade de suportar a pseudonimização, ao mesmo tempo que suporta os níveis necessários de confidencialidade, integridade, autenticidade, não-repúdio. Para além disso, deve considerar uma robustez que permita a proteção do componente IAM como sendo um ativo central de todo o sistema [52]. Hoje em dia, este tipo de infraestrutura é possível, mas não é ainda amplamente utilizado e são ainda comummente utilizados como identificadores dados sensíveis, como por exemplo os números de segurança social, o que facilita um maior número de crimes relacionados com o roubo de identidades.

De acordo com a situação e contexto, tais sistemas devem suportar o utilizador na tomada de decisões conscientes sobre a utilização de pseudónimos, que representam identidades parciais. Um sistema de gestão de identidades que favoreça a privacidade suporta os utilizadores no processo de assumir diferentes perfis, e funciona como elo de ligação para todas as comunicações entre diferentes aplicações.

Em última análise, são os próprios utilizadores que têm de aceitar a responsabilidade sobre a sua própria privacidade, caso contrário nenhuma entidade poderá garantir a proteção dos mesmos [52]. Os utilizadores devem estar conscientes sobre os seus direitos e obrigações, assim como conhecer o contexto em que atuam, para que possam escolher uma identidade parcial apropriada. A combinação entre a usabilidade do sistema e a consciencialização sobre os direitos de privacidade são um pré-requisito para garantir a privacidade dos seus dados.

Diferentes níveis de minimização de dados

Apesar da minimização dos dados ser um dos objetivos de uma gestão de identidades eficiente, não é o único objetivo, uma vez que anonimidade incondicional não é sempre desejada [52]. Uma gestão de identidades que proteja a privacidade pode ser desenhada para oferecer diferentes níveis de autenticidade e acoplamento, ou seja, encontrar um balanço entre a anonimidade e a identificação total. As aplicações que utilizam este tipo de gestão de identidades definem os requisitos para a utilização de identidades parciais e a variedade de opções possíveis para os utilizadores, sendo adaptáveis às respetivas situações. Um exemplo é o sistema de gestão de identidades Shibboleth [53], desenhado para permitir aos utilizadores fornecer apenas um subconjunto dos atributos para um fornecedor de serviços após autenticação bem-sucedida. Até mesmo o id de utilizador é considerado um atributo que pode ou não ser fornecido dependendo da necessidade do serviço em questão precisar de identificar os utilizadores. Por exemplo, uma loja online que fornece material aos membros de uma universidade poderá não necessitar de identificar univocamente o utilizar, mas apenas que este frequenta a universidade em questão. Desta forma é incorporado e suportado o principio de minimização dos dados.

Em alguns casos, poderá ser possível anonimidade absoluta, enquanto que em outros casos poderá ser necessário existir dados que permitam a identificação dos utilizadores. Autorização específica para um determinado contexto maximiza a eficácia da troca de informação ao mesmo tempo que minimiza o

acoplamento entre diferentes contextos prevenindo o cruzamento de informação para criar o perfil de um determinado indivíduo.

Front-end seguro

Uma vez que a componente de IAM armazena e processa dados pessoais sensíveis, a ferramenta em si e a infraestrutura correspondente deverão oferecer um nível apropriado de segurança [52]. Idealmente, esta componente deverá estar localizada num ambiente da confiança do utilizador. Por razões de disponibilidade, replicação, ou serviços de back-up, os utilizadores poderão querer externalizar total ou parcialmente a componente de gestão de identidades a um fornecedor de serviços de confiança.

Pseudónimos e credenciais

A pseudonimização compreende vários níveis de acoplamento a um indivíduo, variando desde a anonimidade até a identificação total [52]. Adicionalmente, a utilização de pseudónimos pode ser utilizada para implementar a responsabilização. Quanto à reputação associada a um indivíduo esta pode ser estabelecida ou consolidada ao reutilizar um pseudónimo.

Um pseudónimo, em conjunto com os dados pessoais, forma uma identidade parcial. Propriedades relevantes dos pseudónimos incluem [52]:

- **Autenticação e autorização** – as credenciais ou atributos associados a um pseudónimo digital podem suportar os processos de autenticação e autorização.
- **Prova de posse** – pseudónimos podem ser gerados pelos próprios utilizadores ou gerados e atribuídos por entidades terceiras. No contexto da gestão de identidades, a correspondência entre um pseudónimo e o titular a que o mesmo está associado não é publica por definição. Prova de posse é a capacidade de provar a propriedade sobre um pseudónimo sem divulgar informação pessoal adicional. Pseudónimos digitais podem ser implementados com uma chave pública para testar assinaturas digitais, onde o proprietário do pseudónimo prova essa propriedade ao formar uma assinatura digital, que é criada utilizando a respetiva chave privada. Por exemplo, chaves públicas PGP associadas a endereços de email, são pseudónimos digitais.
- **Acoplamento entre contextos** – se o mesmo pseudónimo for utilizado várias vezes, ou em contextos diferentes, os respetivos dados sobre o titular, divulgados em alguma das circunstâncias, podem ser ligados. De um modo geral, a anonimidade é maior quanto menor for o número de utilizações, e entre um menor número de contextos, de um mesmo pseudónimo. Existe a distinção entre pseudónimos transacionais, que são utilizados para uma única ação; grupo de pseudónimos, que são utilizados em um contexto específico (por exemplo de acordo com a função do titular ou parceiros de comunicação); e pseudónimos que abrangem diferentes contextos, como substitutos do nome ou identificador do respetivo titular.
- **Convertibilidade –** num sistema de credenciais anónimas, tal como introduzido por David Chaum [54], os utilizadores são conhecidos por diferentes organizações através de diferentes pseudónimos, e estes pseudónimos não podem ser interligados. No entanto, uma organização pode emitir um certificado para um determinado pseudónimo, e o utilizador correspondente pode provar a posse dessa mesma credencial perante uma outra organização, sem relevar nada mais que a própria posse da credencial. A posse de uma credencial pode ser provada repetidamente para diferentes pseudónimos sem que estes possam ser ligados. De qualquer forma, provar a posse de diversas credenciais obtidas a partir de diferentes pseudónimos é apenas possível quando estas forem de facto emitidas para o mesmo utilizador.

Para que um sistema de gestão de identidades possa ser considerado um potenciador de privacidade é necessário que permita aos seus utilizadores escolherem o nível de pseudonimização apropriado, ao mesmo tempo que mantém as capacidades convencionais para identificação, autenticação, autorização e não-repúdio.

O sistema de gestão de identidades deverá ser flexível na adaptação das suas propriedades a cada situação. Para maximizar a privacidade, por definição deverão existir pseudónimos transacionais e pseudónimos associados à função dos utilizadores, onde o acoplamento é desejado para um contexto específico.

Para atingir o desacoplamento entre "quem" e "o quê" deverá ser aplicada a separação do conhecimento. Por exemplo, numa loja online poderá ser feita a separação entre "quem compra" e "o que compra" ao aplicar a separação entre os serviços de pagamento e entrega, e deste modo nem a entidade que trata o pagamento ou a entidade que faz a entrega tem todos os detalhes sobre o utilizador.

Role-Based Access Control (RBAC)

O principal objetivo em garantir um controlo de acessos com base em perfis é garantir que os utilizadores de um determinado sistema não têm acesso indiscriminado às funcionalidades e informação existentes no mesmo [55]. Com recurso a este tipo de controlo de acessos os utilizadores apenas acedem aos recursos que lhe são necessários através dos perfis que lhes são atribuídos. Deste modo, é simplificada a gestão das autorizações ao mesmo tempo que é acrescentada flexibilidade em especificar e garantir o cumprimento de políticas de proteção adequadas. Os utilizadores podem tornar-se membros de um determinado perfil de acordo com as suas responsabilidades e qualificações e para além disso podem facilmente ser atribuídos ou retirados perfis aos utilizadores. Os perfis podem ainda ser alterados sem que exista a necessidade de se alterar também a infraestrutura de acessos. Com a utilização de RBAC, as decisões de quem pode aceder a que informação são tomadas tendo em conta a função que o utilizador desempenha na organização, e estes estarão sempre ao abrigo das políticas de proteção aplicadas para a sua função.

De uma perspetiva funcional, o RBAC significa que a cada perfil estará associado um conjunto de operações, e aos utilizadores são apropriadamente atribuídos determinados perfis [55]. Por exemplo, um utilizador pode estar associado a um ou mais perfis, e um determinado perfil poderá estar associado a um ou mais utilizadores. Os perfis poderão ser criados de acordo com a função dos utilizadores na organização, e o conjunto de operações associadas a um perfil irão restringir os utilizadores a esse mesmo conjunto de operações. Por exemplo, no sistema de um hospital poderão existir os perfis de médico, enfermeiro, paciente, etc. e cada um deles fará uma utilização diferente do sistema.

Este tipo de controlo de acessos é eficaz para sistemas que tratam informação sensível, uma vez que suporta a especificação de competências para realizar determinadas tarefas, especificação de regras que evitam conflitos de interesse e ainda promove o princípio do privilégio mínimo.

3.2.5 Logging e monitorização

Com o crescente aumento do risco de fugas de dados não autorizadas, a segurança dos dados é nos dias de hoje uma prioridade elevada para as organizações. Manter protegida a informação de clientes e colaboradores é um requisito essencial para a maior parte das empresas, principalmente nos setores financeiro e da saúde. No entanto, saber onde procurar para identificar uma fuga de dados pode ser um grande desafio. É aí que as soluções de monitorização e SIEM podem ajudar [81], ao fornecer informação sobre o estado de um sistema, permitindo às organizações identificar e reagir às ameaças à medida que estas surgem.

Enquanto que gestão de *logs* e plataformas SIEM são por norma referidas em conjunto, estes dois sistemas oferecem funcionalidades e benefícios destintos [81]. Compreender as capacidades únicas de cada solução é essencial para encontrar o sistema de monitorização adequado para cada negócio.

Um sistema para gerir a privacidade dos dados consiste num sistema de gestão de bases de dados para armazenar todos os dados de um conjunto de bases de dados sobre os utilizadores, incluindo o registo irrevogável de todos os acessos, sejam estes autorizados ou negados. Deste modo, este sistema de gestão de privacidade, administra e regista todos os dados, utilizadores e acessos a informação que contenha elementos de privacidade, do mesmo modo que inclui os meios para gerir as notificações aos utilizadores, acessos, correções e alteração de preferências de privacidade ou proteção de dados.

O RGPD não é explícito no que diz respeito aos registos que devem ser mantidos relativos às atividades de tratamento de dados pessoais [82]. No entanto, diversas autoridades de proteção de dados consideram os *logs* sobre as diversas atividades de tratamento como um bom meio para demonstrar a conformidade, o que é um dos princípios do RGPD (responsabilização). Manter os *logs* de todas as instâncias de atividades de tratamento de dados pessoais é uma boa prática, e deve ser feita nos seguintes cenários [82]:

- **Registar acesso aos dados** – quem acedeu, dados acedidos e quando. Se o acesso aos dados acontecer através de uma interface centralizada (UI e/ou API) é possível manter o registo de todos os acessos aos dados, e desta forma demonstrar que apenas pessoas autorizadas tiveram acesso aos mesmos;
- **Registar modificações aos dados** – um dos princípios do RGPD é a integridade dos dados, ou seja, os dados têm de ser atuais e corretos, o que significa que qualquer modificação deverá ser registada. Deste modo, é possível recuperar uma modificação ou demonstrar que a modificação aconteceu por um motivo específico.
- **Registar atividades específicas do RGPD –** por exemplo quando o titular dos dados invoca algum dos seus direitos. Cada pedido deverá ser devidamente registado para que se possa demonstrar perante as autoridades a sequência de eventos relacionada com um determinado titular dos dados.
- **Registar o consentimento e as respetivas circunstâncias –** deve ser registada a data, hora, endereço IP, etc. Deverá ser também registada a retirada de consentimento. Desta forma, o histórico de consentimento do titular dos dados será visível num único sítio e é possível demonstrar perante as autoridades reguladoras quando é que existiu, ou não, o consentimento para o tratamento dos dados.

Alguns destes cenários podem ser facilmente geridos a partir de algumas entradas numa base de dados, no entanto, recorrer a uma solução mais robusta e inviolável fornece maiores garantias e reduz o risco de existirem complicações devido a modificações não autorizadas nos registos.

3.2.6 Data Loss Prevention

Na economia digital dos dias de hoje, o volume de dados que entram e saem de uma organização é cada vez maior. Uma empresa típica envia e recebe, numa base diária, milhões de mensagens de email e downloads e efetua milhares de transferências de ficheiros a partir de diversos canais.

Infelizmente, as organizações acabam por se tornar vítimas de enormes perdas de dados, muitas vezes envolvendo dados pessoais e sensíveis, o que representa enormes perdas financeiras e reputacionais.

Tipos de perdas

A perda de dados pode ser dividida em duas categorias distintas que podem sobrepor-se [56]:

- **Divulgação não autorizada –** quando os dados sensíveis deixam de estar sob o controlo da organização, ou seja, quando existe perda de confidencialidade. Esta forma de perda de dados está muitas vezes relacionada com o comprometimento de bases de dados, fazendo com que a consequência mais comum seja o roubo de identidades.
- **Desaparecimento ou danos** – quando deixa de existir uma cópia correta dos dados da organização, ou seja, quando existe perda de integridade ou disponibilidade.

No caso em que a última cópia física dos dados seja roubada, a organização enfrenta ambos os problemas descritos anteriormente. Noutros casos, poderá não ser imediatamente claro qual das situações está em causa. Por exemplo, um problema comum para as organizações é a perda ou roubo dos computadores pessoais dos colaboradores. Se um colaborador estiver a trabalhar sobre determinada informação, a sua cópia poderá ser a mais atual. Sem a garantia de uma gestão de registos eficaz, poderá ser difícil determinar qual a versão mais correta.

A prevenção da perda de dados consiste em impedir que informação sensível deixe de estar sob o controlo da organização [56], e com os recentes incidentes de fugas de dados [57, 58], tecnologias de prevenção de perda de dados estão a emergir como um controlo importante para manter a segurança da informação e privacidade.

Estado dos dados

Os dados das empresas, tipicamente, podem existir em 3 estados:

- **Dados em repouso –** significando que os dados existem em sistemas de ficheiros, desktops distribuídos e grandes data stores centrais, bases de dados ou outros centros de armazenamento;
- **Dados nos *endpoints* –** significando que os dados existem nos *endpoints* da rede tais como computadores pessoais, dispositivos USB, drives externos, CDs/DVDs, fitas de arquivo, leitores MP3, smartphones, ou outros dispositivos móveis;
- **Dados em trânsito –** significando que os dados existem em movimento na rede, saindo para o mundo exterior à organização através de email, mensagens instantâneas, *peer-to-peer* (P2P), FTP ou outros mecanismos de comunicação.

Os dados em cada um dos diversos estados exigem diferentes técnicas para prevenção da perda. Assim sendo, um programa de proteção de dados efetivo deverá adotar as técnicas apropriadas para cobrir a perda em cada um dos possíveis estados.

Capacidades de prevenção

Um programa eficiente de prevenção de perda de dados consiste nas seguintes capacidades [56]:

- **Gestão** – consiste em definir políticas de manuseamento dos dados ao nível da organização, reportar incidentes de perda de dados, e estabelecendo uma capacidade de resposta a tais incidentes através da implementação de medidas corretivas. A prevenção de perda de dados não é apenas uma questão tecnológica, é também uma questão de implementar as políticas de gestão adequadas. Estas políticas devem determinar questões como deve ser determinado e autenticado o acesso aos dados, assim como determinar de que forma é que o cumprimento das políticas é garantido. Funcionalidades de gestão devem também incluir a capacidade de reportar a perda de dados e o *workflow* de gestão da remediação de incidentes.
- **Descoberta** – consiste em definir a sensibilidade dos dados da organização, criar um inventário dos dados sensíveis, localizar os dados sensíveis e onde estes estão armazenados e gerir o

apagamento de dados desnecessários. Isto inclui descobrir e inventariar os dados sensíveis em repouso ou armazenados em *endpoints*, para que os mesmos possam ser devidamente protegidos ou realocados.

- **Monitorização** – consiste em monitorizar o tratamento de dados sensíveis, compreender os padrões de utilização e aumentar essa visibilidade ao nível da organização. Isto poderá incluir a monitorização dos dados em trânsito ao inspecionar as comunicações na rede que não estão em cumprimento para com as políticas de segurança e monitorizar os dados nos *endpoints* para verificar se são copiados para armazenamento local, USBs ou outros dispositivos de armazenamento móvel ou impressos.
- **Proteção** – consiste em garantir o cumprimento das políticas de segurança para proteger os dados de forma proativa e prevenir a saída da organização. A proteção automática dos dados sensíveis de forma transversal a *endpoints*, rede e sistemas de armazenamento deverá incluir a proteção dos dados em repouso através de criptografia, quarentenas e remoção. Deverá garantir também a restrição de impressão, cópia, acesso, transferência e download dos dados sensíveis para dispositivos móveis. Por fim, deverá impedir os dados em trânsito de serem transferidos em violação das políticas de segurança ou utilizar criptografia para garantir transferência segura.

O problema das fugas de dados deve ser resolvido ao utilizar sistemas de prevenção de perda de dados, Data Leakage/Loss Prevention System (DLP) [59]. As soluções DLP ajudam a identificar, monitorizar, proteger e reduzir os riscos de perda de dados sensíveis. São utilizadas para detetar e prevenir que utilizadores não autorizados tenham acesso a dados sensíveis ou até que estes sejam partilhados de forma acidental [60]

Tecnologias utilizadas para a proteção de dados

Existem diversas tecnologias utilizadas para a proteção de dados, a maior parte foca-se em proteger os dados a partir do exterior, enquanto que os sistemas DLP focam-se na proteção a partir do interior [59]. De seguida são descritas algumas das tecnologias mais comuns.

IDS/IPS

Um sistema de deteção de intrusões (IDS) é um dispositivo ou software que monitoriza a rede ou sistema para identificar atividades maliciosas [59]. Um sistema de prevenção de intrusões (IPS) monitoriza a rede ou sistema para identificar e bloquear atividades maliciosas. Um IPS é, portanto, uma extensão de um IDS e pode realizar ações desde acionar alarmes até descartar os pacotes maliciosos.

A maior parte dos sistemas IDS/IPS consistem em mais do que uma aplicação ou dispositivo hardware, e são compostos pelas seguintes partes [61]:

- **Sensores de rede** – detetam e enviam dados aos sistemas;
- **Sistema de monitorização central** – processa e analisa os dados enviados pelos sensores;
- **Análise de relatórios** – oferecem informação sobre como reagir a um determinado evento;
- **Bases de dados** – armazenam os endereços IP e restante informação sobre os atacantes;
- **Mecanismo de resposta** – correlaciona a informação dos restantes componentes e formaliza uma resposta adequada.

As técnicas utilizadas pelos sistemas IDS/IPS dividem-se em duas abordagens:

- **Com base em assinaturas ou *pattern matching* –** quando o sistema depende de uma base de dados sobre ataques conhecidos que são carregadas no sistema na forma de assinaturas. A maior desvantagem deste tipo de sistemas é que apenas são capazes de detetar ataques conhecidos, e deste modo ataques inovadores poderão não ser detetados.

- **Com base em anomalias** – requerem que o sistema passe por um período de aprendizagem para que seja identificado o comportamento normal dos utilizadores. Este tipo de sistemas funciona bem para identificar comportamentos que se desviem do que constitui a atividade normal do sistema.

Anti-malware

Malware é um tipo de software desenhado para causar danos às operações computacionais, colecionar dados sensíveis, e ganhar acesso não autorizado a um sistema [59]. Existem diversos tipos de *malware*, nomeadamente vírus, Trojans, spyware, backdoors, etc.

Anti-malware é qualquer tipo de software que forneça proteção dos computadores e sistemas de qualquer programa prejudicial [59]. Este tipo de software funciona em tempo real de forma bastante eficiente, mas é apenas indicado para proteção de ameaças exteriores, ao verificar e validar assinaturas garante que as infeções por *malware* serão removidas.

Firewalls

Firewalls são dispositivos ou software que permitem ou negam transmissões na rede com base num conjunto de regras de acesso definidas, e são utilizadas para proteger as redes de acessos não autorizados ao mesmo tempo que permitem que as comunicações autorizadas aconteçam [62]. Este software ou hardware ajuda a manter as redes seguras, uma vez que controla o tráfego a entra e sair da rede ao analisar os pacotes e determinar se estes são ou não permitidos.

SIEM

Security Information Event Management (SIEM) é uma ferramenta utilizada pelas organizações para centralizar o armazenamento de *logs* gerados pelo software a correr na rede, assim como reunir informação, analisar e apresentar a mesma [63]

Muitas das técnicas descritas anteriormente utilizam o método Deep Packet Inspection (DPI). Este método olha para os pacotes, encontra anomalias no tráfego e gera alertas ou previne o tráfego. DPI classifica o tráfego com base em regras sobre os metadados dos pacotes, assim como a informação que descreve o conteúdo dos mesmos [64].

Capítulo 4

Framework de Segurança e Privacidade (proposta)

Neste capítulo é apresentada uma proposta de uma arquitetura para um sistema de software seguro, criada com base no trabalho de pesquisa efetuado e descrito nos capítulos anteriores. É proposta também uma metodologia de segurança criada com base não só no trabalho de pesquisa desenvolvido como também recorrendo a uma meta-metodologia proposta na literatura para o desenvolvimento de metodologias de segurança.

4.1 Arquitetura para um sistema de software seguro

Qualquer que seja a aplicação e o contexto em que a mesma é utilizada, e com o objetivo de proteger os dados dos seus utilizadores, existe um conjunto de controlos que devem existir para garantir que os requisitos de privacidade são endereçados. Este tipo de controlos pode ser dividido em duas vertentes, uma vertente mais técnica, orientada à própria solução, e uma vertente processual, orientada à gestão dos processos organizacionais respetivos à aplicação.

Para garantir a proteção da informação existente, de um ponto de vista técnico, existe um conjunto de componentes que devem existir e que devem ser considerados os mínimos para garantir uma proteção da informação adequada e em conformidade com o atual regulamento em vigor (RGPD).

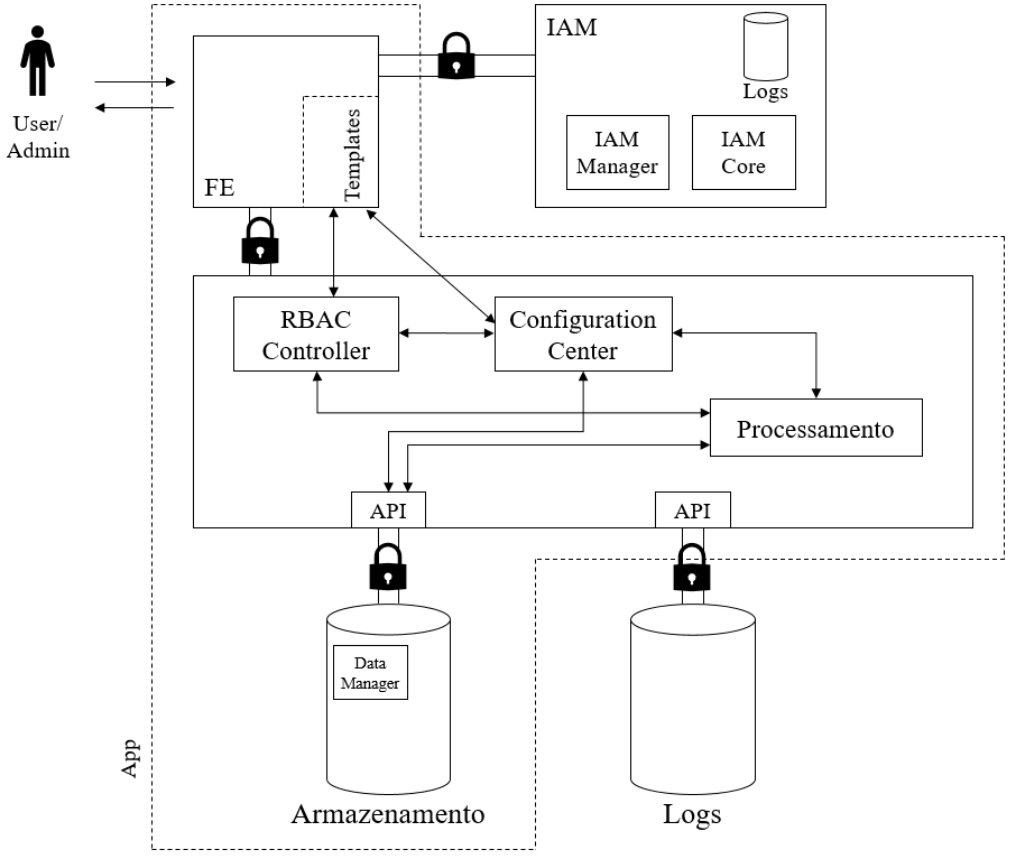

Figura 4.1 *– Arquitetura proposta.*

Na **Figura 4.1** é apresentado um diagrama que propõe uma arquitetura, onde são identificados os componentes essenciais e as principais relações entre eles, assim como uma descrição adequada e uma justificação para a existência de tais componentes de um ponto de vista da privacidade dos utilizadores e dos seus dados.

A arquitetura sugerida assenta em 3 camadas, o *front-end*, a camada aplicacional e o armazenamento. Para além disso, recorre a dois componentes externos à aplicação sendo eles o IAM e um sistema de *logs*. Toda a comunicação entre os diversos componentes deverá ser feita de modo seguro, para que não exista o comprometimento dos dados na transferência dos mesmos.

Por uma questão de simplicidade, é considerado que existem dois tipos de utilizadores: um *user* que apenas tem acesso à sua informação, e um *admin* que terá acesso a toda a informação no sistema e de todos os utilizadores respetivamente. De notar que dependendo das opções de privacidade que cada utilizador tome, o *admin* poderá não ter visibilidade sobre todos os dados do respetivo utilizador, da mesma forma que qualquer tipo de tratamento sobre os mesmos poderá também ser limitado.

4.1.1 Front-End

É nesta camada que é realizada toda a interação com os utilizadores do sistema. Este componente irá ser bastante importante uma vez que é onde efetivamente os utilizadores do sistema visualizam a informação nele contida. Este componente deverá comunicar com o sistema de IAM, para fazer a autenticação do utilizador, e posteriormente com a camada aplicacional.

Templates

Deverão existir *templates* pré-configurados para possibilitar a visualização da informação por parte dos utilizadores. Estes *templates* serão independentes do perfil de cada utilizador. Cada perfil apenas visualiza a informação que lhe é permitida e as respetivas configurações de privacidade. Não será de esperar que o comportamento do sistema seja alterado pelas definições de privacidade, mas uma vez que o utilizador poderá optar por não exibir determinada informação, poderão existir diferenças na forma como a informação é visualizada.

4.1.2 Camada aplicacional

RBAC Controller

Dependendo do perfil de cada utilizador da aplicação, o conjunto de operações que o mesmo poderá realizar será diferente. Enquanto a autenticação dos utilizadores é feita pelo componente IAM, deverá ser na própria lógica da aplicação que este conjunto de operações é disponibilizado ou limitado a cada perfil. O RBAC Controller serve como um intermediário entre o componente IAM e a lógica da aplicação, limitando o conjunto de operações realizado ao estritamente necessário para o desempenhar de funções (i.e., as funções associadas a um determinado perfil).

Configuration Center

O consentimento explícito do utilizador para o tratamento de dados é introduzido a partir do *front-end*. É a partir do *configuration center* que o utilizador poderá controlar todas as suas configurações de privacidade. Esta ação irá influenciar praticamente todos os restantes componentes da aplicação que dependam destas configurações, desde a visualização, processamento (i.e., lógica de negócio) e armazenamento. Para além disso, é a partir deste componente que o utilizador terá a capacidade de

intervir no que diz respeito aos seus dados, desde a retirada de consentimento relativo ao tratamento, atualização ou até mesmo remoção dos mesmos.

Como configuração padrão, deverão ser recolhidos o mínimo dados pessoais possível e o consentimento do utilizador não deverá nunca ser assumido. Para além disso, deverá existir granularidade nas opções de privacidade, e deverão existir os avisos de privacidade adequados relativamente aos dados recolhidos, incluindo quando existirem alterações ao tratamento dos dados.

Processamento

Este componente é a parte central de toda a lógica de negócio da aplicação. Apenas serão tratados os dados dos utilizadores cujo consentimento explícito tenha sido previamente recolhido.

4.1.3 Armazenamento

É o componente onde será armazenada toda a informação relativa a cada utilizador da aplicação. Para além dos dados de cada utilizador, são também armazenadas todas as configurações necessárias para o funcionamento da aplicação. O armazenamento de dados é realizado a partir de uma API limitada, permitindo que os mesmos sejam armazenados de forma estruturada, por utilizador e em alinhamento com a lógica da aplicação, facilitando posteriormente a sua remoção ou atualização em caso de intervenção.

Data manager

Para garantir que os dados pessoais armazenados estão em conformidade com o RGPD, deverá existir um módulo no armazenamento que garanta que os mesmos são atualizados, quando os seus titulares assim o requisitarem, ou removidos de forma automática quando os mesmos não forem mais necessários. Nos casos em que os utilizadores não desejarem que os seus dados sejam removidos, mas que também não pretendam que os mesmos sejam tratados para um determinado efeito, os dados devem ser mantidos, mas deverão ser anotados de forma a que o sistema tenha em consideração esta limitação.

Deverá também existir a capacidade de exportar todos os dados armazenados, relativos a um determinado utilizador, num formato eletrónico comummente aceite.

4.1.4 Comunicações

Todas as comunicações realizadas entre as diversas camadas da aplicação, assim como a comunicação realizada com os componentes externos à mesma, deverão considerar todas as boas práticas para a transferência segura de dados. Deverá ser utilizada criptografia para evitar ataques de observadores passivos, e deverão existir também proteções para prevenir ataques de *phishing*, nomeadamente através da utilização de certificados digitais.

4.1.5 IAM

Apesar de não existir uma necessidade imperativa, por questões de disponibilidade, replicação e escalabilidade, o componente de gestão de identidades e acessos poderá ser externo à aplicação. Para garantir que questões de privacidade dos utilizadores do sistema não sejam postas em causa, deverá recorrer-se a um fornecedor de serviços de confiança que garanta a confidencialidade e integridade da informação. Para garantir uma gestão de identidades que potencie a privacidade dos utilizadores do sistema, o IAM deverá consistir em dois módulos, IAM Manager e IAM Core. Deverá também ser

considerado como requisito mínimo, para um sistema de IAM, o registo (*logging*) de todos os utilizadores e respetivos acessos.

IAM Core

Este é o módulo central de qualquer sistema de IAM, deverá implementar as funcionalidades *core* como autenticação, autorização, *single-sign-on* e gestão de identidades.

IAM Manager

Este é o módulo que deverá implementar as políticas específicas de privacidade para cada aplicação. Por exemplo, o tipo de identificador a utilizar na aplicação deve ser definido neste componente. Por exemplo, se a aplicação permitir a utilização de pseudónimos por parte dos seus utilizadores, deverá ser neste componente que é feita a gestão dos respetivos IDs (identificadores).

4.1.6 Logs

Para possibilitar a demonstração de conformidade para com o RGPD, na medida em que os dados pessoais são tratados de acordo com os requisitos impostos pelo regulamento, e apenas quando existe o consentimento explícito para tal, deverá existir um sistema de *logs* que registe todas as ações realizadas com os dados. Qualquer ação de criação, visualização, atualização e remoção dos dados (CRUD) deverá ser registada, assim como quem e quando realizou a ação. Para além disso, todo o consentimento sobre o tratamento de dados pessoais, ou retirada do mesmo, deverá ser registado com o *timestamp* em que o mesmo aconteceu. Desta forma é possível demonstrar que o tratamento dos dados apenas foi realizado quando existiu consentimento para tal. Por questões de conflito de interesses, este componente deverá ser externo à aplicação e é imperativo que a solução seja robusta e inviolável, fornecendo maiores garantias e reduzindo o risco de modificações não autorizadas nos registos.

Manter todos os registos relacionados com o RGPD requer algumas decisões a nível de arquitetura. Muitas vezes a solução adotada passa por ter um armazenamento centralizado que é acedido apenas a partir de uma API limitada, permitindo que cada invocação da própria API constitua uma evidência de auditoria.

Nos casos em que exista uma fuga de dados, o sistema de *logs* é um bom ponto de partida para a investigação do sucedido, e poderá ter informação importante para conter a fuga de informação, como por exemplo ao identificar as contas que acederam aos dados, e bloquear as mesmas devido a um possível comprometimento.

Por fim, e com o objetivo de garantir uma demonstração de conformidade para com o regulamento, poderá ser feito um cruzamento dos *logs* gerados a partir da aplicação com os que foram gerados pelo componente de IAM.

4.2 Metodologia de segurança (PMD)

Incorporar funcionalidades de segurança e privacidade é uma das tarefas mais importantes e desafiantes no desenho de sistemas seguros. Para desenvolver sistemas com sucesso, as funcionalidades de segurança devem ser incorporadas no contexto de uma metodologia de segurança, ou seja, uma abordagem sistemática, estruturada, e que combina princípios de engenharia de software e de segurança [65]. A incorporação de atributos de segurança e privacidade começa desde as fases iniciais e procede ao longo de todo o ciclo de desenvolvimento de software.

Uma metodologia de segurança constitui "uma abordagem sistemática para fazer coisas de uma determinada maneira" [68], em que neste caso a disciplina é a engenharia de software seguro. O trabalho desenvolvido pretende apresentar uma metodologia de segurança, com o nome de PMD, especificamente desenhada para o desenvolvimento de sistemas tendo em conta as especificidades de um sistema que trata dados pessoais. A metodologia proposta tem maior incidência nas fases iniciais do ciclo de vida de desenvolvimento de software (análise de requisitos e desenho) mas endereça também as preocupações que devem ser tidas na fase de implementação e *deployment*. Esta metodologia é capaz de endereçar todas ou a maior parte das preocupações de privacidade existentes, fornecendo às equipas de desenvolvimento orientação sobre como e onde introduzir funcionalidades de segurança relevantes na arquitetura de um sistema durante o seu desenvolvimento.

A abordagem utilizada para o desenvolvimento da metodologias de segurança apresentada em [66] é composta por 3 partes interligadas: uma *framework* interligada de *process patterns* e pattern modifiers [69] de segurança chamada SPPF (Security Process Pattern Framework), para a construção dos processos de segurança de uma metodologia de segurança; um meta-modelo para a construção da *framework* conceptual; e uma meta-metodologia unificadora (S-SMEP) para orientar os engenheiros na utilização dos artefactos.

Apesar de ser uma metodologia autónoma, PMD é também um exemplo de uma metodologia reestruturada, desenvolvida ao aplicar a abordagem para desenhar metodologias de segurança apresentada em [66], na sua capacidade de *tailoring*, utilizando a metodologia proposta em [74] como metodologia base (que por sua vez foi obtida a partir de [67]). Visto que uma metodologia de segurança é considerada uma solução para o problema de introduzir a segurança no software de forma sistemática, e que a abordagem para desenhar metodologias referida é uma meta-solução capaz de gerar soluções, então PMD pode ser vista como um exemplo de uma solução particular, com o seu próprio conjunto de funcionalidades benéficas.

Para garantir que a metodologia proposta é aplicada corretamente, todos os colaboradores envolvidos no desenvolvimento de software devem receber treino adequado, garantindo que conhecem os princípios básicos de segurança e privacidade tais como *design* seguro, modelagem de ameaças, código seguro e testes de segurança. Indivíduos com funções técnicas (*developers*, *testers)* devem receber treino com maior frequência (no mínimo anualmente) para garantir que se mantêm atualizados relativamente a novas tendências. Devem ser conhecidos conceitos fundamentais de segurança aplicacional, tanto de uma perspetiva conceptual como técnica, tais como validação de *inputs*, *output encoding*, tratamento de erros, *logging*, autenticação, autorização, etc. Devem também ser conhecidas as vulnerabilidades mais comuns (e.g., OWASP Top 10, CWE) para cada tipo de sistema a ser desenvolvido.

4.2.1 Visão geral da PMD

A PMD pode ser vista como uma evolução da metodologia de segurança proposta em [74] obtida a partir da aplicação de uma iteração S-SMEP, com o objetivo de customizar a mesma tendo em conta as especificidades de um sistema que trata dados pessoais. Como tal, foram consideradas as atividades e boas práticas propostas pelas metodologias descritas na **secção 2.3**, com o objetivo de aprimorar a metodologia base. Esta iteração resultou num conjunto de alterações que tiveram em conta as especificidades de um sistema desenhado para proteger a privacidade dos seus utilizadores.

De um modo geral, a estrutura da PMD (apresentada na **Figura 4.2**, cuja nomenclatura é adaptada de [70] e pode ser consultada em [66]) segue a ordem das fases de desenvolvimento de um ciclo de desenvolvimento de software genérico: Requirements analysis (*ReqAn*), compreendendo as atividades

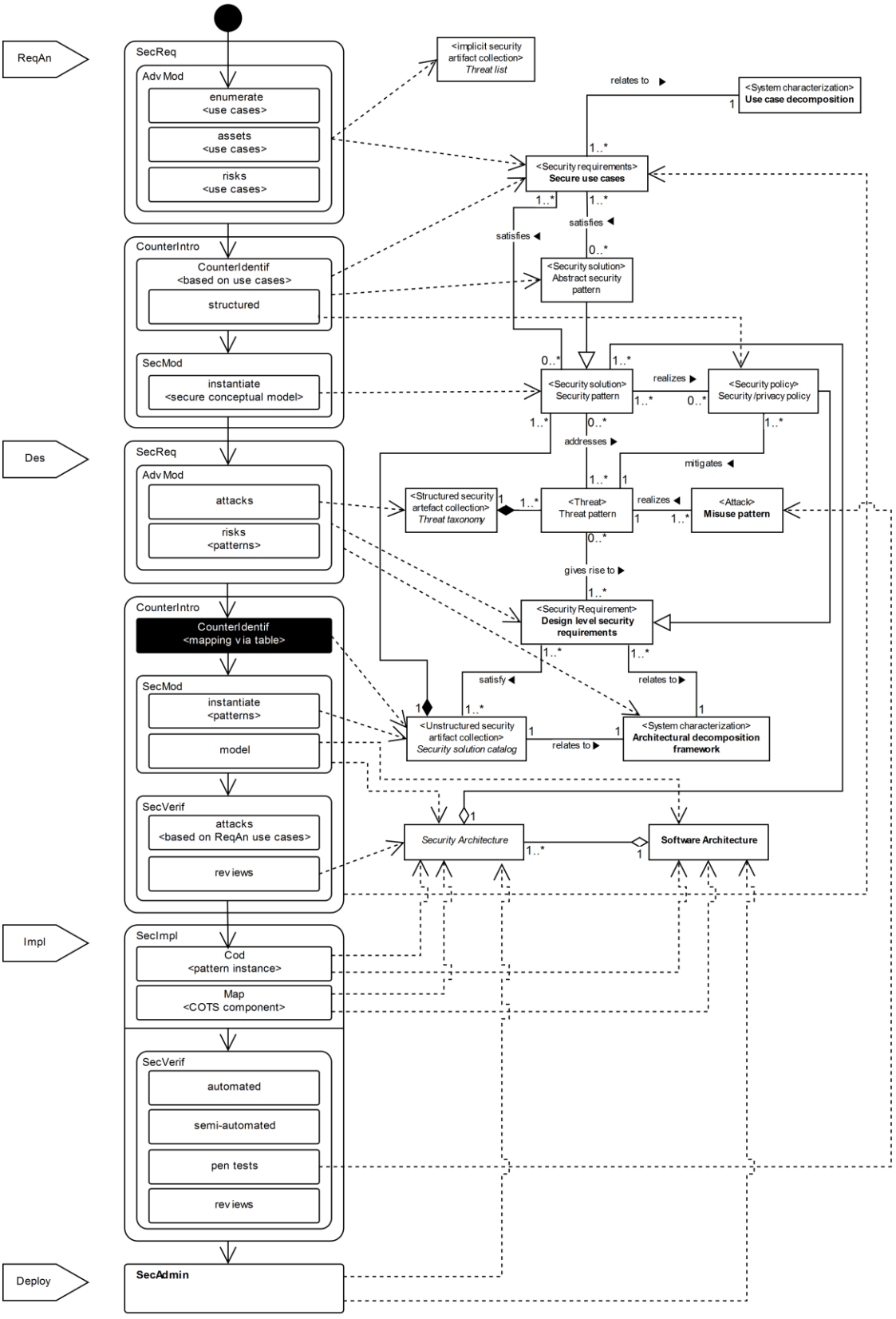

Figura 4.2 *– Modelo pattern-based da metodologia proposta.*

de levantamento (*Req*) e análise (*An*) de requisitos; Design (*Des*), compreendendo as atividades de desenvolvimento de arquitetura (*Arch*) e desenho detalhado (*DetDes*); Implementation (*Impl*), compreendendo as atividades de desenvolvimento de código (*Cod*) e testes (*Test*); e Deployment (*Deploy*), compreendendo as atividades de instalação e administração.

De um modo geral, a PMD começa por considerar os casos de uso de um sistema durante a análise de requisitos (*ReqAn*), e de que forma é que os mesmos podem ser subvertidos tanto por *insiders* como por *outsiders* (SecReq – phase; AdvMod – stage). Desta forma é possível identificar (*CounterIdentif*) nesta fase inicial do desenvolvimento as medidas de segurança apropriadas e modelar as mesmas no modelo conceptual do sistema (*SecMod*). Durante a fase de *design* (*Des*) é realizada uma sequência semelhante de atividades, desta vez relativa à arquitetura do sistema, considerando em detalhe as funcionalidades do mesmo, assim como as ameaças e medidas de segurança apropriadas, e desta forma modelar as medidas de segurança para criar uma arquitetura de software segura. As restantes atividades dizem respeito à verificação (*SecVerif*) em diferentes fases, à implementação de segurança (*SecImlp*), e à administração de segurança (*SecAdmin*).

O aspeto gráfico da metodologia, representado numa notação inspirada em UML, consiste em duas partes interligadas: um diagrama de atividades estendido (lado esquerdo) para representar os processos de segurança, e um diagrama de classes (lado direito) para representar a *framework* conceptual. Os dois diagramas estão ligados através de setas em tracejado que partem dos processos para os elementos da *framework* conceptual, denotando a relação de "utiliza" ou "pertence a".

4.2.2 PMD: framework conceptual

Os elementos que constituem a *framework* conceptual da PMD aderem às classes originais do meta-modelo proposto em [66]. Do lado direito da **Figura 4.211**, para além dos artefactos relacionados com os casos de uso seguros, existe um conjunto de artefactos ofensivos e defensivos (elementos *security solution* e *threat* respetivamente). Na **Tabela 4.1** serão descritos cada um dos elementos conceptuais utilizados na metodologia. Na **Tabela 4.2** é acrescentado algum detalhe sobre alguns dos artefactos conceptuais. Por fim, serão descritos os artefactos mais relevantes utilizados nos processos de segurança durante a fase de desenho.

Tabela 4.1 *– Descrição dos elementos conceptuais selecionados a partir do meta-modelo da framework conceptual. Adaptado de [66].*

Elemento conceptual	Descrição
Attack	Um ataque representa um conjunto de ações de natureza adversa realizadas para comprometer os atributos de segurança (confidencialidade. Integridade, disponibilidade, etc.) de um sistema. Os ataques realizam as ameaças e exploram vulnerabilidades.
Threat	Uma ameaça representa alguma forma de potencial impacto negative (e.g., execução de código inseguro) que pode ser realizada por múltiplos ataques em diferentes contextos (e.g., vírus, buffer overflows).
Security Requirement	Um requisito de segurança representa qualquer requisito de projeto ou sistema que diga respeito à segurança enquanto atributo, determinado, pelo menos nominalmente, no domínio do problema. Os requisitos de segurança ditam as necessidades de segurança de um sistema tanto de uma perspetiva de alto nível como ao nível técnico.
Security policy	Uma política de segurança representa um requisito de segurança no domínio da solução, que é por sua vez realizado por uma determinada solução de segurança.

Supportive artefact	Um artefacto de segurança é qualquer artefacto que não esteja diretamente relacionado com a segurança, mas que é essencial para, ou no mínimo contribui, para a utilização de um ou mais artefactos de segurança (e.g., uma tabela de mapeamento ou uma coleção de modelos).
Security solution	Uma solução de segurança é qualquer medida tomada para garantir que determinado atributo de segurança de um sistema é salvaguardado, ou seja, uma solução de segurança satisfaz determinados requisitos de segurança
System characterization	Uma caracterização de um sistema é uma representação de todo, ou parte, de um sistema ao qual os diversos artefactos de segurança podem ser relacionados.

Tabela 4.2 *– Descrição dos artefactos de segurança conceptuais da metodologia. Adaptado de [66] e [74].*

Artefacto conceptual de segurança	Descrição
Misuse patterns	*Misuse patterns* descrevem em detalhe como é realizado um ataque (incluindo quais os componentes de software necessários) da perspetiva do atacante, quais os *security patterns* que podem mitigar o ataque, e como é que o ataque pode ser identificado quando ocorrer. Mais informação sobre como modelar *misuse cases* pode ser consultada em [71].
Security patterns	*Security patterns* encapsulam as medidas de segurança em um formato reutilizável adequado para o desenvolvimento da arquitetura.
Security architecture	O conjunto de todas as medidas de segurança ao nível do desenho constituem a arquitetura de segurança, ou seja, a arquitetura de segurança é a parte de uma arquitetura de software que lida com a segurança.
Threat patterns	*Threat patterns* encapsulam as ameaças (potenciais ataques) e permitem considerar as mesmas em um contexto arquitetural bem definido.

Architecture decomposition framework

A decomposição arquitetural é uma forma de caracterização de um sistema que, quando acompanhada por um processo de análise, fornece a estrutura necessária para guiar na incorporação inicial de requisitos não funcionais, neste caso requisitos de segurança, assim como ajuda na determinação de novos requisitos. Mais informações sobre este tipo de caracterização de um sistema podem ser consultadas em [72].

O artefacto *Architectural decomposition framework* funciona como um complemento abstrato dos modelos de software arquiteturais e está dividido em 3 camadas. A primeira camada consiste num modelo conceptual de sistemas distribuídos utilizável como meta-modelo para criar modelos arquiteturais. A segunda camada consiste num meta-modelo arquitetural em camadas que permite a separação da arquitetura de acordo com áreas de funcionalidade (camadas de decomposição funcionais). A terceira e última camada consiste num conjunto de meta-elementos (abstrações de realizações técnicas) que permitem detalhar a decomposição.

Threat taxonomy (and library)

A taxonomia de ameaças consiste numa coleção dos diferentes *threat patterns* existentes. Em [73] é possível consultar uma coleção de *threat patterns* para sistemas distribuídos que compreende tanto as ameaças ao sistema como as ameaças à própria infraestrutura de segurança. A taxonomia referida pode ser estendida, ao especificar cada um dos *patterns* em relação a diferentes contextos arquiteturais, para produzir taxonomias específicas de um determinado sistema ou taxonomia.

Estas taxonomias, tal como a taxonomia base, combinam os benefícios de uma biblioteca estruturada e um sistema de classificação STRIDE [85], e dão suporte à modelagem de ameaças uma vez que permitem considerar uma vasta variedade de ameaças adaptadas a contextos específicos. O artefacto *Threat taxonomy* está relacionado com o artefacto *Architectural decomposition framework* uma vez que este define o contexto arquitetural para cada um dos *threat patterns*.

Security solution catalog

O catálogo de soluções de segurança consiste no conjunto de soluções que encapsulam e organizam os *security patterns*, em diferentes níveis de abstração, e em diferentes facetas, fornecendo diretrizes na realização dos *security patterns* de um formato abstrato e conceptual para concretizações ao nível do desenho, e simultaneamente ao longo de todo o ciclo de desenvolvimento.

Existe um conjunto de soluções propostas na literatura para grande parte das preocupações essenciais de segurança em sistemas de software. Na **Tabela 4.3** é apresentado um conjunto de soluções de segurança que endereçam as principais preocupações existentes (consultar **secção 3.2** para uma descrição do estado da arte de cada uma destas soluções).

Tabela 2.3 *– Descrição das soluções de segurança. Adaptado de [74].*

Solução de segurança	Descrição
Autorização	Encapsula *security patterns* que permitem a construção de modelos conceptuais de autorização e a sua realização em diferentes arquiteturas.
Gestão de identidades	Encapsula *security patterns* relacionados com a gestão de identidades (atribuição, remoção, validação) dos utilizadores e/ou subsistemas.
Comunicações seguras	Encapsula *security patterns* que permitam que duas ou mais partes comuniquem de forma segura através de um canal de comunicação.
Data Loss Prevention	Encapsula *security patterns* para a filtragem de informação tanto ao nível da rede como da aplicação, incluindo *firewalls* e filtros de dados personalizados.
Armazenamento seguro	Encapsula *security patterns* para o armazenamento seguro de informação, incluindo armazenamento de passwods, segurança de bases de dados, entre outros.
***Logging* e monitorização**	Encapsula *security patterns* para o registo e monitorização de eventos em um sistema (desde a criação de simples ficheiros de *logs* até à implementação de sistemas de deteção de intrusões (monitorização da rede).
Gestão de informação de segurança	Encapsula *security patterns* relacionados com a gestão da informação de segurança tal como políticas, credenciais, chaves criptográficas, etc.

4.2.3 PMD: processos de segurança

Nesta secção serão descritos os processos de segurança da metodologia proposta, alinhados com as fases genéricas do ciclo de desenvolvimento de software. As fases genéricas consideradas, ou seja, o primeiro nível SPPF, foram análise de requisitos (ReqAn), *design* (Des), implementação (Impl) e *deployment* (Deploy). A fase de implementação engloba as atividades relativas ao que seria considerada a fase genérica de verificação e a fase de *deployment* engloba as atividades relativas ao lançamento e operação do sistema.

Quanto aos processos de segurança a serem considerados em cada uma das fases de desenvolvimento, ou seja, o segundo nível SPPF, os mesmos são detalhados nas secções subsequentes. Por questões de legibilidade, esta secção foi numerada de forma independente do resto do documento.

1. Requirements Analysis (ReqAn)

As atividades realizadas durante a fase de análise de requisitos consistem em especificar os requisitos não funcionais de um determinado sistema, neste caso requisitos de segurança (SeqReq), com o objetivo de definir quais as medidas de segurança devem ser implementadas (CounterIntro). No contexto de um paradigma com base em modelos poderão ser criados diversos modelos (conceptuais e/ou arquiteturais) para incorporar os requisitos. A fase de análise de requisitos representa o espaço do problema do projeto.

1.1. Security Requirement Determination (SeqReq) phase

Os requisitos de segurança determinados nesta fase devem ser divididos em 2 tipos: prescritivos e resultantes.

Os requisitos de segurança prescritivos são equivalentes a políticas de alto nível que descrevem um conjunto de requisitos de segurança que devem ser endereçados. Este tipo de política poderá ser definido pela própria organização (e.g., política de acordos com fornecedores) ou poderá corresponder a requisitos legais ou regulamentares (e.g., RGPD – **secção 2.4.4**). Como tal, é necessário identificar quais são requisitos de conformidade, e criar diretrizes para garantir a conformidade com os mesmos, ou seja, é necessário criar políticas e normas para a segurança e conformidade.

Os requisitos de segurança resultantes são, como o nome indica, o resultado da realização de algum tipo de avaliação ao sistema. Os requisitos de segurança devem ser derivados das funcionalidades do próprio sistema (i.e., funcionalidades de negócio) considerando riscos conhecidos. Este tipo de avaliação é, por norma, algum tipo de modelagem de ameaças em que os possíveis ataques ao sistema são considerados. No que diz respeito a requisitos de privacidade de um sistema (**Capítulo 3**), os mesmos são em grande parte derivados das funcionalidades de negócio, e como tal é necessário especificar os mesmos (e.g., criação de uma matriz para controlo de acessos entre perfis e recursos) para serem considerados nas avaliações realizadas.

O conjunto de requisitos de segurança prescritivos e resultantes formam o conjunto final de requisitos de segurança do sistema, ditando quais são as medidas de proteção necessárias. Este tipo de artefactos é ainda do domínio do problema e não representam ainda as contramedidas de segurança, que são determinadas numa fase posterior.

1.1.1. Adversary Modeling (AdvMod) stage

Nesta etapa é determinado de que forma é que o sistema pode ser atacado, e quais as medidas que podem ser introduzidas para garantir a sua proteção. Desta forma, é necessário identificar as potenciais

vulnerabilidades do sistema para posteriormente modelar as possíveis ameaças, relacionando os atacantes com os seus potenciais ataques (*threats*) e tendo em conta as suas capacidades (e.g., disponibilidade de acessos) e objetivos. Durante a fase de análise de requisitos, esta modelagem pode ser feita considerando um conjunto de casos de uso ou, caso exista, uma arquitetura preliminar do software.

No que diz respeito a esta etapa, existe um conjunto de tarefas relacionadas que devem ser realizadas no seu decorrer. A modelagem de ameaças, por norma, segue os seguintes passos (sendo que o passo 3 e 4 são por vezes opcionais):

- **1** – Identificar os ativos (Assets) ou casos de uso do sistema;
- **2** – Enumerar (Enumerate) as ameaças;
- **3** – Detalhar e refinar (Refine) as ameaças;
- **4** – Estimar riscos (Risks).

Quanto à tarefa de identificar os ativos, o objetivo passa por identificar qual a informação e/ou partes de um sistema necessitam ser protegidos. A partir da representação inicial do sistema, ou de uma representação secundária (e.g., diagramas de fluxo de dados), devem ser determinadas quais os elementos que devem ser protegidos (e.g., dados, interfaces, subsistemas, processos, etc.). Este levantamento pode ser realizado em colaboração com as equipas de desenvolvimento, clientes e restantes *stakeholders*. A proteção dos diversos elementos pode ser considerada como um todo (i.e., uma parte do sistema deve ser protegida contra todas as ameaças) ou de acordo com um conjunto específico de propriedades de segurança como confidencialidade, integridade, disponibilidade, entre outros (poderá ser consultada uma taxonomia completa em [75]).

Quanto à tarefa de enumerar as ameaças, é necessário considerar todos os possíveis ataques que possam comprometer uma parte vulnerável do sistema seguindo uma abordagem estruturada. Esta tarefa poderá tornar-se mais rápida ao recorrer-se a listas de ataques (e.g., CWE) ou padrões de ataque comuns. Existe um conjunto variado de técnicas que podem ser utilizadas para a realização desta tarefa, incluindo *threat trees* [76], *attack trees* e *misuse cases* [77, 78].

Por fim, existindo um conjunto de ameaças ou possíveis falhas de um sistema, deve ser avaliado o risco da ocorrência de cada uma das ameaças, individualmente ou em conjunto com outras, atribuindo a cada ameaça valores ou medidas qualitativas. Estes riscos podem posteriormente ser tabulados e analisados para ajudar na tomada de futuras decisões. Uma forma de realizar a avaliação de risco de forma estruturada é, por exemplo, recorrendo ao sistema de ratings DREAD da Microsoft [79].

Este último passo, a análise de risco, é especialmente relevante na conceção de sistemas que tratam dados pessoais, uma vez que este tipo de sistema poderá levar a riscos significativos para a privacidade dos seus utilizadores, e como tal deverá ser conduzida uma análise de impacto da privacidade (DPIA) para garantir que são mitigados à partida todos os riscos identificados (em [80] é possível consultar uma introdução às noções básicas, requisitos e principais passos para conduzir uma análise de risco de privacidade). Com base na avaliação de risco, os dados e aplicações deverão ser classificados de acordo com o perfil de risco do negócio, permitindo estabelecer diferentes objetivos de segurança por classificação.

1.2. Countermeasure Introduction (CounterIntro) phase

O objetivo desta fase é a introdução das medidas de segurança necessárias, através de uma abordagem estruturada, convertendo os requisitos de segurança do domínio do problema, identificados na fase anterior (1.1. Security Requirement Determination), em objetivos e políticas do domínio da solução. Durante a fase de análise de requisitos devem ser desenvolvidos os modelos conceptuais que

representam os mecanismos de segurança a serem introduzidos. Como tal, o resultado é um conjunto coerente de representações das medidas de segurança, ou parcialmente coerente se estiver planeado um refinamento dos mesmos, que podem ser implementadas no software.

1.2.1. Countermeasure Identification (CounterIdentif) stage

O objetivo desta fase é determinar o conjunto de medidas necessárias para dar resposta aos requisitos de segurança de um sistema. Como tal, deverá ser possível identificar as medidas apropriadas para dar resposta a cada ameaça identificada. Cada requisito do domínio do problema deverá ser mapeado num conjunto de abstrações de medidas de segurança do domínio da solução. Em primeiro lugar, os requisitos devem ser mapeados em objetivos e políticas, essencialmente convertendo a terminologia do domínio do problema para a terminologia do domínio da solução, para serem posteriormente mapeados em representações particulares de cada solução.

Esta fase deverá considerar todos os requisitos de segurança, ou seja, tanto os requisitos prescritivos como resultantes devem ser considerados e associados a uma solução apropriada. Como tal, deverá recorrer-se a uma abordagem estruturada para realizar o mapeamento de forma sistemática entre os requisitos de segurança e um conjunto apropriado de medidas correspondentes (e.g., recorrendo a tabelas). Uma vez que o conjunto de possíveis medidas de segurança é conhecido antecipadamente, o processo de mapeamento permite a rastreabilidade das medidas de segurança em relação aos respetivos requisitos endereçados.

1.2.2. Security Modeling (SecMod) stage

Após a identificação do conjunto de medidas de segurança que devem ser introduzidas no sistema, as mesmas devem ser incorporadas nos respetivos modelos arquiteturais. As medidas de segurança podem ser representadas de diversas formas, e cada representação poderá exigir uma abordagem diferente para a sua incorporação. Os mecanismos de segurança devem ser desenvolvidos em conjunto com os aspetos funcionais do sistema, no entanto, devem ser separados na medida em que possam sofrer alterações e evoluir por si próprios.

Durante a fase de análise de requisitos, as medidas de segurança devem ser incorporadas (i.e., modeladas ou de alguma forma representadas) nos modelos conceptuais do sistema de software (i.e., na arquitetura de software preliminar), ou como uma arquitetura de segurança separada. Poderá não existir um mapeamento de um para um entre as medidas de segurança identificadas e os elementos dos modelos disponíveis ou soluções de segurança representáveis, e como tal as relações entre os diferentes modelos devem ser consideradas para garantir a sua consistência (entre si e os elementos arquiteturais existentes, quer sejam funcionais ou não funcionais), correção (de cada modelo) e completude (cobrindo tomas as medidas de segurança identificadas).

Esta fase poderá ser realizada com a ajuda de peritos em segurança, que guiam o processo, ou recorrendo a ferramentas e técnicas mais acessíveis a *developers* inexperientes. A abordagem para a incorporação das medidas de segurança pode ser automática ou semiautomática recorrendo a ferramentas de suporte, ou manual.

2. Design (Des)

As atividades realizadas durante a fase de desenho consistem em concretizar as necessidades e requisitos de um sistema num conjunto de modelos arquiteturais expressos em uma ou mais Architecture Description Languages (ADLs), i.e., determinar os requisitos de desenho de um sistema, identificar as medidas de segurança apropriadas para satisfazer esses requisitos e modelar as medidas identificadas na arquitetura de software do sistema.

Os modelos podem ser organizados em diversas *views* e podem ser de diversos tipos, dependendo do paradigma de desenvolvimento. À medida que o desenho evolui para níveis mais elevados de detalhe, os modelos de alto nível são refinados para desenhos de baixo nível para cada um dos componentes. A fase de desenho representa o espaço da solução do projeto.

2.1. Security Requirement Determination (SeqRec) phase

Nesta fase será determinado um conjunto de requisitos de segurança ao nível do desenho. Como tal, será necessário determinar um conjunto de requisitos técnicos (ao nível do desenho) para dar resposta a cada uma das ameaças identificadas. Estes requisitos podem ser determinados após realizada a modelagem de ameaças ou em paralelo com a mesma. Estes requisitos técnicos devem ser especificados em um de dois formatos canónicos:

- **Negativo** – <proteger de X>, onde X é uma instância de um *threat pattern*; ou
- **Positivo –** <impor Y>, onde Y é a política de segurança genérica, ou específica, encapsulada pelo *threat pattern* correspondente à ameaça X.

A especificação dos requisitos neste formato reflete a dualidade existente entre os *threat patterns* e as soluções de segurança que constituem a *framework* conceptual da metodologia, tal como explicado na **secção 4.2.2**. Esta *framework* será então explorada na fase seguinte da metodologia para determinar as medidas de segurança apropriadas para mitigar as ameaças.

2.1.1. Adversary Modeling (AdvMod) stage

Para determinar um conjunto de requisitos de segurança ao nível do desenho, deverá ser realizado um processo de modelagem de ameaças que relacione os diversos *threat patterns* existentes às diversas partes do sistema de software. O primeiro passo para a realização deste processo é a caracterização do sistema, que pode ser vista como uma decomposição arquitetural do mesmo. Esta decomposição é a base para a análise das ameaças, ou seja, servirá para criar e manter modelos de ameaça específicos para o software, desenvolvendo os perfis dos atacantes a partir da arquitetura do software.

Após a decomposição do sistema de software em um conjunto de modelos e abstrações, o resto do processo de modelagem de ameaças irá depender dos *threat pattern*s considerados. Estes poderão ser genéricos ou específicos do contexto do sistema ou tecnologia utilizada. A análise de ameaças poderá ser realizada a partir de duas abordagens complementares, que em ambos os casos recorrem à taxonomia de ameaças definida anteriormente.

A primeira abordagem (*context-driven*) consiste em considerar cada camada funcional do sistema e selecionar os *threat patterns* apropriados para cada contexto (um exemplo de mapeamento entre camadas funcionais do sistema e taxonomia de ameaças pode ser consultada no **Anexo D**). Da mesma forma, devem ser selecionadas as realizações técnicas (abstratas) para cada um dos *threat patterns* considerados (um exemplo de mapeamento entre realizações técnicas abstratas e taxonomia de ameaças pode ser consultada no **Anexo E**). Desta forma, as ameaças podem ser mapeadas ou relacionadas com os elementos correspondentes do modelo de alto nível da decomposição, consideradas relativamente às diversas áreas funcionais do sistema e consideradas em relação às realizações particulares de cada funcionalidade (e.g., protocolos de rede, integridade e confidencialidade das mensagens, etc.).

A segunda abordagem (*threat-driven*) consiste em considerar cada *threat pattern* e relacioná-lo com as diferentes partes da arquitetura do sistema.

Assim que um *threat pattern* tenha sido selecionado, este deverá ser instanciado em relação à arquitetura do sistema para que seja determinada a ameaça concreta. Cada ameaça deverá também ser

analisada em relação a todas as camadas da decomposição, considerando as suas repercussões verticalmente ao longo das mesmas.

Após a criação da lista das ameaças a considerar, o impacto e a probabilidade de cada ameaça deverão ser considerados e a ameaça respetivamente categorizada, i.e., deverá ser realizado algum tipo de análise de risco.

Assim que alguma política ou requisito tenha sido determinado, este deverá ser considerado ao longo de todas as camadas de decomposição, tal como foi feito com a ameaça, para determinar as estratégias aplicáveis ou ainda novos requisitos que possam surgir. De notar que o mesmo se aplica à incorporação dos requisitos de segurança identificados nas fases de desenvolvimento anteriores.

Por fim, é importante referir que a combinação entre a decomposição e modelagem de ameaças é iterativa por natureza, i.e., os passos anteriormente descritos poderão ter de ser repetidos à medida que a arquitetura vai sendo criada.

2.2. Countermeasure Introduction (CounterIntro) phase

O objetivo desta fase é a introdução dos requisitos de desenho, identificados na fase anterior (4.2.3 - 2.1. Security Requirement Determination), traduzindo-os em modelos completos de arquitetura de software.

2.2.1. Countermeasure Identification (CounterIdentif) stage

Uma vez identificados os requisitos, os mesmos devem ser mapeados em medidas de segurança concretas. As medidas de segurança apropriadas são determinadas a partir de políticas genéricas encapsuladas pelos *threat patterns*. Um exemplo deste mapeamento pode ser visto no **Anexo F**.

O primeiro passo para determinar as medidas de segurança a partir dos requisitos é converter todos os requisitos na forma negativa para o seu formato positivo recorrendo à política de segurança genérica encapsulada pelo *threat pattern* relevante, ou seja:

<proteger de X> -> <impor [política genérica de segurança encapsulada por X]>

Deste modo todos os requisitos são convertidos num conjunto de políticas que necessitam ser impostas. Este conjunto de políticas poderá ser subsequentemente mapeado em medidas concretas em vários níveis de abstração e granularidade, ou seja, são selecionados os *security patterns* apropriados.

2.2.2. Security Modeling (SecMod) stage

Quando as medidas de segurança apropriadas tiverem sido identificadas, as respetivas soluções devem ser concretizadas na arquitetura do sistema de software. Independentemente de as soluções identificadas terem sido obtidas a partir de políticas genéricas de segurança, ou selecionadas diretamente, o processo para a incorporação das soluções identificadas passa por concretizar os diversos níveis de abstração da solução identificada. Mais especificamente, este processo implica que os *abstract patterns* sejam selecionados, definindo os conceitos da solução e uma arquitetura abstrata, de seguida a arquitetura abstrata seja sucessivamente refinada, levando a um desenho detalhado e, finalmente, seja selecionado um conjunto de realizações para o desenho que serão depois implementados ao nível do código. Deste modo, as soluções de segurança são introduzidas passo a passo, e os *security patterns* são relacionados com a caracterização do sistema, começando com a camada mais alta da decomposição funcional do sistema, e descendo nas camadas, relacionando os *security patterns* aos diferentes níveis de abstração.

No que diz respeito a questões relacionadas com a privacidade, existe um conjunto de padrões e estratégias de *design* que devem ser considerados na arquitetura de sistemas seguros (ver **secção 3.1**). Para além disso, existe um conjunto de soluções de segurança que deverá ser considerado o mínimo para sistemas que tratem dados pessoais (ver **secção 3.2**). Na **secção 4.1** é descrita uma proposta (exemplo) de uma arquitetura de alto nível de um sistema que garante a privacidade dos seus utilizadores.

2.2.3. Security Verification (SecVerif) stage

Uma vez obtida a arquitetura de segurança através da instanciação dos diferentes *security patterns* o próximo passo é verificar que a arquitetura resultante cumpre com os requisitos identificados e, em particular, que a mesma endereça as ameaças enumeradas anteriormente. Isto é realizado através da correspondência das soluções aos *threat patterns* iniciais e ao conjunto de casos de uso seguros, construídos durante a fase de desenho e análise de requisitos respetivamente, e inspecionar se os mesmos são devidamente endereçados pela arquitetura de segurança. Por outras palavras, o efeito de cada uma das medidas de segurança deve ser verificado e monitorizado, garantindo que as medidas são corretas, consistentes e completas. As revisões e inspeções devem ser realizados de forma estruturada, considerando os requisitos de forma sistemática ou recorrendo a uma *checklist* para guiar o processo.

3. Implementation (Impl)

As atividades realizadas durante a fase de implementação consistem em realizar as medidas de segurança exigidas pelo desenho do sistema. A implementação da segurança deve acontecer em paralelo com a implementação dos aspetos funcionais do sistema, implicando que a abordagem utilizada esteja alinhada com o paradigma de desenvolvimento.

As atividades de implementação de segurança podem apenas ser realizadas na fase de implementação do paradigma do desenvolvimento (a atividade de desenvolvimento de código em particular). Quanto às atividades de verificação e validação, as mesmas podem ser realizadas durante a fase de testes do paradigma do desenvolvimento.

3.1. Security Implementation (SecImpl) phase

Na fase de implementação de segurança os modelos arquiteturais de segurança são traduzidos em componentes de software concretos. Independentemente dos modelos de segurança serem uma versão aprimorada do modelo funcional ou um modelo independente, os mesmos podem ser traduzidos em software através da sua implementação em código ou utilizando componentes COTS.

O resultado desta fase deverá ser uma implementação completa, ou parcial se estiver planeado o seu refinamento, de alguns ou todos os modelos arquiteturais, resultando num conjunto de subsistemas de software concretos que endereçam as preocupações de segurança do sistema.

3.1.1. Coding (Cod) stage

As medidas de segurança são implementadas pelas equipas de desenvolvimento, como subsistemas concretos, em conjunto com as funcionalidades do sistema. Durante este processo, todas as boas práticas de desenvolvimento e de gestão de projetos devem ser aplicadas para garantir a qualidade das soluções resultantes. Este processo assume que todas as considerações foram tidas no que diz respeito às eventuais contrapartidas das medidas de segurança em relação às funcionalidades do sistema, e que a implementação é baseada nas descrições detalhadas da arquitetura do sistema de software.

Este processo implica que quaisquer configurações necessárias para os componentes de segurança sejam realizadas pelas equipas de desenvolvimento, ou preparadas para serem modificadas na fase de *deployment*.

3.1.2. Map (Map) stage

Uma outra forma de implementar as medidas de segurança determinadas, ao mesmo tempo que permite a redução do esforço necessário para a sua implementação, é mapear todos os elementos existentes nos modelos de segurança a componentes já existentes (e.g., COTS, *legacy*), garantindo que os requisitos de segurança existentes nos modelos são alcançados pelos mecanismos dos componentes selecionados.

3.2. Security Verification (SecVerif) stage

Uma vez terminadas todas as iterações da fase de implementação de segurança, deverá ser realizada uma revisão da superfície de ataque, ou seja, o sistema de software resultante deverá ser verificado de forma a garantir que o mesmo satisfaz a arquitetura de segurança. Isto é conseguido ao considerar a realização de cada *threat pattern* identificado nas fases anteriores e realizando testes de intrusão ao sistema. Os testes de intrusão devem tentar de alguma forma contornar as medidas de segurança introduzidas, explorando sistematicamente fraquezas conhecidas do sistema. Devem ser considerados os possíveis *exploits* das medidas de segurança introduzidas e também as fraquezas do sistema, recorrendo a ferramentas de análise de segurança automáticas e listas de vulnerabilidades conhecidas.

Para além da revisão da superfície de ataque, deverá também ser realizada uma revisão de código. Esta tarefa pode ser realizada através de análises automáticas ou semiautomáticas que utilizam ferramentas para fazer verificações ao código. Quanto possível, estas ferramentas deverão estar integradas nos próprios processos de desenvolvimento.

4. Deployment (Deploy)

A fase de *deployment* consiste no lançamento e manutenção do sistema de software.

4.1. Security Administration (SecAdmin) phase

É nesta fase que o sistema de software desenvolvido é colocado em produção e, como tal, é necessário configurar o sistema para garantir a sua correta operação no que diz respeito às funcionalidades de segurança. Mais precisamente, é necessário configurar cada um dos componentes implementados da infraestrutura de segurança, definindo políticas apropriadas para cada um dos controlos. Toda a infraestrutura subjacente deverá, da mesma forma, ser configurada de acordo com os requisitos de segurança existente. Podem ser utilizadas ferramentas de observação em tempo de execução como um meio para recolher informação e monitorizar as operações de segurança, fornecendo *feedback* contínuo. Esta fase é apenas aplicável uma vez que o sistema e a infraestrutura de segurança correspondente tenham sido implantados em ambiente de produção. O resultado desta fase é um conjunto de definições e configurações da infraestrutura de software.

Antes do sistema ser colocado em produção, é também importante que sejam definidos um plano e uma equipa de resposta a incidentes. Deste modo, é assegurado que qualquer incidente que ocorra durante a operação do sistema é identificado, e analisado, atempadamente, garantindo que a as medidas de segurança se mantêm apropriadas e que o sistema não é comprometido. Por fim, deverão ser criados os procedimentos apropriados para a gestão de alterações em cada lançamento.

Capítulo 5

Avaliação

Neste capítulo é apresentada uma avaliação da solução proposta com o objetivo de demonstrar que a mesma endereça os requisitos de segurança existentes para um sistema de software seguro. Na **secção 6.2** é descrito o trabalho futuro relativo à avaliação da metodologia.

Uma vez que o âmbito deste trabalho residiu na definição de uma metodologia que endereçasse os requisitos de segurança em todo o ciclo de desenvolvimento de software, não foram produzidos artefactos de software concretos que permitissem quantificar a melhoria, em termos de deteção e mitigação de falhas de segurança comparativamente com soluções existentes. Sendo assim, e com o objetivo de garantir que a metodologia é adequada e eficiente, foi decidido que a mesma seria avaliada de acordo com os requisitos impostos pela Resolução de Conselho de Ministros nº41/2018.

Uma vez que a Resolução de Conselho de Ministros nº41/2018 (RCM) lista os controlos (i.e., orientações técnicas), obrigatórios e recomendados, necessários para a Administração Pública, em matéria de segurança das redes e sistemas de informação, assim como procedimentos a adotar de modo a cumprir as normas do RGPD, podemos assumir que se a metodologia proposta endereçar todos os requisitos impostos então a mesma é adequada para o desenvolvimento de software que endereça as preocupações derivadas do RGPD. Assim sendo, a **Tabela 5.1** apresenta a avaliação da metodologia proposta baseada num racional sobre de que forma é que a PMD endereça cada um dos requisitos da RCM. Por uma questão de apresentação optou-se por apresentar apenas o identificador do requisito e o respetivo racional (no **Anexo G** podem ser consultados os requisitos em maior detalhe).

Quanto à RCM propriamente dita, a mesma está dividida em requisitos gerais, que agrupam requisitos específicos, aplicáveis a diferentes camadas de um sistema de software (i.e., *front end*, camada aplicacional e base de dados). Existem requisitos que são iguais entre diferentes camadas, e uma vez que a PMD não diz como deve ser feito, mas sim o que deve ser feito, foi considerado que se a PMD endereça um requisito para uma das camadas então endereça também para as restantes, deixando os detalhes de como deve ser feito para a implementação concreta. Ainda quanto à RCM, existem requisitos obrigatórios e recomendados. Uma vez que os requisitos recomendados são ainda mais específicos (i.e., ditam como deve ser feito) foram apenas considerados os obrigatórios, evitando fugir à natureza conceptual da PMD. Por exemplo, a PMD dita que deverão ser aplicados *security patterns* relativos à autenticação dos utilizadores, mas não especifica que as *passwords* utilizadas devam ser de uma determinada complexidade, uma vez que este detalhe já diz respeito à implementação concreta da solução de segurança.

Sendo a PMD uma metodologia de segurança aplicada ao desenvolvimento, não detalha todos processos organizacionais de gestão que deverão existir durante o funcionamento do sistema, como por exemplo atribuição e remoção de acessos. Sendo assim, os requisitos relativos aos processos de gestão que devem existir foram considerados endereçados nos casos em que o software desenvolvido tem as capacidades necessárias para que esses processos possam existir (de forma automática ou manual) sem a necessidade de acrescentar ou retirar funcionalidades ou componentes de segurança ao sistema.

Tabela 3.1 *– Avaliação da metodologia proposta*

#	Racional
#01.1	Sendo a PMD uma metodologia de segurança a ser integrada nas fases genéricas do ciclo de desenvolvimento, e uma vez que a mesma considera para cada uma das diferentes fases as atividades de segurança necessárias (i.e., boas práticas de desenvolvimento), então podemos considerar que a PMD segue as boas práticas de desenvolvimento. Em contraste com o exemplo apresentado, a PMD tem em conta o desenvolvimento de código seguro (ver secção 4.2.3 – 3.1.1.) e a submissão desse código a testes de segurança (ver secção 4.2.3 – 3.2.).
#01.2	Uma sessão segura é um mecanismo que garante a segurança qualquer tipo de comunicação através de uma rede, seja ela pública ou privada, incluindo a internet (ver secção 3.2.2 para mais detalhe sobre comunicações seguras). Sendo a PMD uma metodologia que recorre a um conjunto de soluções de segurança, e sendo as "comunicações seguras" uma dessas soluções de segurança, podemos então considerar que a PMD garante a utilização de sessões seguras com protocolos de segurança (ver secção 4.1.4 e 4.2.2).
#01.4	Uma vez que a PMD garante a utilização de sessões seguras (cf. #01.2) e que para além disso considera a solução de segurança "armazenamento seguro" (ver secção 4.1.3 e 4.2.2), podemos então considerar que a PMD garante que não será guardada informação pessoal no browser (e.g., no formato de cookies) para além do tempo da sessão e apenas na medida do necessário.
#01.5	(cf. #01.2)
#01.11	Uma vez que a solução de segurança "comunicações seguras" encapsula *security patterns* que permitem que duas ou mais partes comuniquem de forma segura (ver secção 4.1.4 e 4.2.2), e que a camada aplicacional e a base de dados são exemplos de partes comunicantes, podemos então considerar que a PMD garante a comunicação segura entre a camada aplicacional e a base de dados, sendo o padrão X.509 um exemplo de uma implementação concreta de uma solução.
#01.12	Uma vez que a solução de segurança "armazenamento seguro" encapsula *security patterns* para o armazenamento seguro de informação (ver secção 4.1.3 e 4.2.2), e que armazenamento seguro de dados pessoais passa pela cifra dos mesmos, podemos então considerar que a PMD considera o referido (ver secção 3.2.3 para mais detalhe sobre armazenamento seguro).
#02.1	Uma vez que a solução de segurança "gestão de identidades" (ver secção 4.1.5 e 4.2.2) encapsula *security patterns* que permitem a validação da identidade de um utilizador (i.e., autenticação - ver secção 3.2.1 para mais detalhe sobre autenticação), e que a PMD garante a utilização de sessões seguras (cf. #01.2), podemos então considerar que a PMD garante que o processo de autenticação é iniciado e mantido em sessão segura.
#02.3	Uma vez que a PMD garante a utilização de sessões seguras (cf. #01.2), e que o facto de os dados pessoais estarem guardados em variáveis visíveis ao utilizador não é considerado seguro, podemos então considerar que a PMD garante a exclusão de dados pessoais de sessão das variáveis URL ou de outras variáveis visíveis.

#02.4	Uma vez que a PMD garante que o processo de autenticação é mantido em sessão segura (cf. #02.1), que a solução de segurança "comunicações seguras" encapsula *security patterns* que permitem a comunicação segura entre duas partes (ver secção 4.1.4 e 4.2.2), e que a transmissão de credenciais deverá ser feita através do seu HASH (mínimo SHA-256) para ser considerada segura, podemos então considerar que a PMD garante que as credenciais de início de sessão são transmitidas através do seu HASH e que a restante informação é cifrada.
#02.5	Uma vez que a solução de segurança "gestão de identidades" (ver secção 4.1.5 e 4.2.2) encapsula *security patterns* que permitem a validação da identidade de um utilizador (i.e., autenticação - ver secção 3.2.1 para mais detalhe sobre autenticação), e que para este processo ser considerado seguro existem requisitos mínimos para as credenciais utilizadas, podemos então considerar que a PMD garante que as credenciais utilizadas cumprem com os requisitos mínimos de segurança, correspondendo a sua complexidade ao que é considerado atualmente como boa prática (considerando o risco associado a cada elemento que se autentica).
#02.7	(cf. #02.5)
#02.8	(cf. #02.5)
#02.10	(cf. #01.2 e #02.1)
#02.13	(cf. #02.10)
#02.14	(cf. #02.5)
#03.1	Uma vez que a solução de segurança "gestão de identidades" (ver secção 4.1.5 e 4.2.2) encapsula *security patterns* que permitem a atribuição de perfis de acordo com as necessidades de cada utilizador (ver secção 3.2.4 para mais detalhe sobre Role Based Access Control), e que a atribuição de privilégios para além daqueles considerados os necessários é considerado inseguro, podemos então considerar que a PMD garante a criação de perfis com privilégios mínimos e de acordo com a necessidade de conhecer.
#03.2	Uma vez que a solução de segurança "*logging* e monitorização" (ver secção 4.1.6 e 4.2.2) encapsula *security patterns* para o registo e monitorização de eventos em um sistema (ver secção 3.2.5 para mais detalhe sobre registos e monitorização), e que os acessos à informação presente no sistema (i.e., tratamento da informação) são um exemplo de um evento, podemos então considerar que a PMD garante o registo dos acessos, alterações e remoção de dados, incluindo quem acedeu, de onde acedeu, quando acedeu, a que dados acedeu e a ação efetuada sobre os mesmos.
#03.3	(cf. #03.1)
#03.4	(cf. #03.2)
#03.5	(cf. #03.1)
#03.6	(cf. #03.2)

#04.1	Uma vez que a solução de segurança "gestão de identidades" (ver secção 4.1.5 e 4.2.2) encapsula *security patterns* que permitem a atribuição e remoção de direitos de acesso a cada utilizador, que a solução "autorização" (ver secção 4.2.2) encapsula *security patterns* que permitem a construção de modelos conceptuais de autorização, e que para garantir que a atribuição e remoção é feita de forma segura deverá existir um processo definido associado, podemos então considerar que a PMD garante a atribuição de direitos de acesso e privilégios de forma restrita e controlada.
#04.2	Uma vez que a solução de segurança "gestão de informação de segurança" encapsula *security patterns* relacionados com a gestão da informação de segurança (ver secção 4.2.2), e que as credenciais de acesso são um exemplo de um tipo de informação de segurança, podemos então considerar que a PMD garante a atribuição de credenciais de acesso e forma a permitir a sua auditoria, e sem permitir outro acesso que não o do destinatário da informação.
#04.3	(cf. #04.1)
#04.4	(cf. #04.2)
#04.5	(cf. #04.1)
#04.6	(cf. #04.2)
#05.1	(cf. #04.1)
#05.2	(cf. #03.1, #04.1)
#05.5	(cf. #03)
#05.7	(cf. #03)
#05.9	(cf. #03)
#05.11	(cf. #03)
#06.1	(cf. #03.1, #04.1)
#06.3	(cf. #03.2)
#06.4	(cf. #06.1 e #06.3)
#06.5	(cf. #05)
#06.7	(cf. #03)
#06.10	(cf. #06.1 e #06.3)
#06.11	(cf. #05)
#07.1	(cf. #03.1)
#07.2	(cf. #03.1)
#07.3	(cf. #03.2)
#07.4	(cf. #03.1)
#07.5	(cf. #03.2)
#08.1	(cf. #05 e #06)
#09.1	(cf. #02)
#10.1	(cf. #03.2)
#10.2	Uma vez que a solução de segurança "armazenamento seguro" encapsula *security patterns* para o armazenamento seguro de informação (ver secção 4.1.3 e 4.2.2), e que armazenamento seguro de registos de atividade implica que os mesmos não possam ser adulterados (ver secção 3.2.3 para mais detalhe sobre armazenamento seguro), podemos então considerar que a PMD garante que os registos de atividade são armazenados em modo leitura e com garantias de integridade.

#10.3	(cf. #10.2)
#10.4	(cf. #03.2)
#10.5	(cf. #03.2)
#11.1	Uma vez que a solução de segurança "Data Loss Prevention" encapsula *security patterns* para a filtragem de informação (ver secção 4.2.2), tanto ao nível de rede como da aplicação, podemos então considerar que a PMD garante a proteção da informação em função da sua relevância e criticidade.
#12.1	Sendo a PMD uma metodologia de segurança que considera as possíveis ameaças ao sistema desde as fases iniciais da sua conceção (ver secção 4.2.3 – 1.1.1.) podemos então considerar que a PMD garante a proteção da camada de FE aos ataques mais comuns.
#12.3	(cf. #03, #04, #07 e #10)
#12.4	Durante a fase de Deployment (ver secção 4.2.3 – 4.) o sistema é colocado em produção e como tal é configurado para garantir a sua correta operação no que diz respeito às funcionalidades de segurança. Uma vez que a segregação entre camadas acessíveis aos utilizadores (FE) e as restantes é um controlo essencial para garantir a segurança dos dados (ver secção 4.1 para mais detalhe sobre uma arquitetura de alto nível), podemos então considerar que a PMD garante a segregação entre as camadas aplicacional (e de bases de dados) e o ambiente com acesso exterior (FE).
#12.5	(cf. #03, #04, #07 e #10)
#12.6	(cf. #12.4)
#12.7	(cf. #03, #04, #07 e #10)
#12.8	(cf. #01.11 e #01.12)
#13.1	Durante a fase de Deployment (ver secção 4.2.3 – 4.) o sistema é colocado em produção e como tal é configurado para garantir a sua correta operação no que diz respeito às funcionalidades de segurança. Sendo assim, podemos então concluir que a PMD garante a correta configuração das zonas DNS e sistemas de correio eletrónico.
#14.1	Sendo a PMD uma metodologia de segurança que considera as possíveis ameaças ao sistema desde as fases iniciais da sua conceção (ver secção 4.2.3 – 1.1.1) podemos então considerar que a PMD garante a as propriedades de redundância, resiliência e disponibilidade.
#14.2	Uma vez que a solução de segurança "armazenamento seguro" encapsula *security patterns* para o armazenamento seguro de informação (ver secção 4.1.3 e 4.2.2), e que armazenamento seguro de informação implica que existam back-ups da informação com os mesmos controlos de segurança, podemos então considerar que a PMD garante que a informação armazenada é alvo de backups.
#14.3	Uma vez que a solução de segurança "armazenamento seguro" encapsula *security patterns* para o armazenamento seguro de informação (ver secção 4.1.3 e 4.2.2), e que armazenamento seguro de informação, neste caso backups, implica que os mesmos não possam ser comprometidos pelas mesmas condições que o sistema em produção, podemos então considerar que a PMD garante que a informação armazenada em backups não está exposta aos mesmos riscos exteriores da localização original.
#15.1	Sendo a PMD uma metodologia de segurança que considera as possíveis ameaças ao sistema desde as fases iniciais da sua conceção (ver secção 4.2.3 – 1.1.), e que o tratamento de dados pessoais é por si só um risco ao qual está associado um conjunto de funcionalidades que devem existir, podemos então concluir que a PMD garante que os sistemas estão capacitados para classificar, priorizar, pesquisar, editar e apagar os dados pessoais (ver secção 3.2.2).

#15.2	(cf. #15.1)
#16.1	Sendo a PMD uma metodologia de segurança que considera o tratamento de dados pessoais desde as fases iniciais de conceção de um sistema, e que a interoperabilidade de dados pessoais é um requisito regulamentar para o tratamento dos mesmos (ver secção 2.4.4 para mais detalhe sobre o direito à portabilidade), podemos então considerar que a PMD garante a utilização de formatos digitais compatíveis, sendo que o regulamento referido é uma especificidade dentro da Administração Pública.
#17.1	Sendo a PMD uma metodologia de segurança que considera o tratamento de dados pessoais desde as fases iniciais de conceção de um sistema e que um dos *inputs* para a definição de requisitos de segurança são as políticas de segurança (ver secção 4.2.3 – 1.1.), podemos então considerar que a PMD garante o alinhamento com as políticas de segurança existentes.
#17.2	Sendo a PMD uma metodologia de segurança que identifica os ativos de informação e casos de uso de um sistema desde as fases iniciais da sua conceção (ver secção 4.2.3 – 1.1.) com o objetivo de identificar a informação ou partes de um sistema que devem ser protegidos, podemos então considerar que a PMD garante que é possível identificar inequivocamente o estado da informação em todo o seu ciclo de vida.

Capítulo 6

Conclusões e Trabalho Futuro

Neste capítulo é apresentado um resumo do trabalho desenvolvido e as principais conclusões, assim como as possibilidades de trabalho futuro que poderão contribuir para a melhoria da solução proposta.

6.1 Trabalho realizado e conclusões

O trabalho descrito neste documento foi desenvolvido no âmbito do Projeto em Engenharia Informática, disciplina de conclusão do Mestrado em Engenharia Informática da FCUL. Mais precisamente, foi desenvolvido durante o estágio numa empresa de acolhimento, a EY. O objetivo do trabalho assentou sobre a criação de uma *framework* capaz de ajudar as empresas que desenvolvem software a aumentar o nível de segurança das suas aplicações.

A primeira fase foi dedicada à investigação sobre as metodologias de segurança atualmente existentes para o desenvolvimento de software seguro. Foi também um dos focos da pesquisa inicial o conceito *Privacy by Design* e as implicações que o mesmo tem no desenvolvimento de software. A partir da pesquisa realizada, foi possível identificar as propriedades que devem ser garantidas num sistema que preserva a privacidade dos seus utilizadores, assim como os componentes de segurança essenciais que devem ser parte integrante desse sistema. Para cada um destes componentes é apresentado um estado da arte de um ponto de vista da privacidade. Uma vez definidos os requisitos a garantir num sistema seguro, foi possível criar uma proposta de uma arquitetura segura que determina quais os componentes que devem existir num determinado sistema. Foi também desenvolvida uma proposta de metodologia de segurança (PMD) a ser integrada no Ciclo de Vida de Desenvolvimento de Software. A arquitetura proposta em conjunto com a PMD constituem a *framework* correspondente ao objetivo do trabalho.

O trabalho desenvolvido tem como base o conhecimento adquirido ao longo de todo o período de estágio. Foi possível identificar que para garantir a segurança dos sistemas de software, e nomeadamente para garantir a privacidade dos seus utilizadores, é necessário reunir um conjunto de condições essenciais que nem sempre dependem do código. A combinação entre pessoas, processos e tecnologias é a chave para atingir este objetivo. Para começar, é necessário que todos aqueles que estão envolvidos no desenvolvimento de software tenham conhecimento prático nas matérias de segurança, desde princípios de engenharia seguros até às boas práticas de desenvolvimento. É necessário garantir também que existe uma abordagem estruturada para integrar a segurança em todas as partes do ciclo de vida de desenvolvimento de software. Atividades como revisão de arquiteturas e revisão de código garantem que o sistema é sucessivamente refinado até que atinja o nível desejado de segurança, sempre considerando as possíveis ameaças de uma perspetiva do atacante, através de atividades como modelagem de ameaças e testes de intrusão. Através da definição formal de uma metodologia, suportada por diversos processos e políticas, é possível garantir que a segurança é parte integrante de um sistema, minimizando assim o risco de possíveis ameaças.

O resultado deste trabalho pode ser visto como uma solução particular de uma metodologia de segurança, com o seu próprio conjunto de funcionalidades benéficas. A metodologia proposta fornece diretrizes sobre quais as medidas de segurança que devem ser consideradas, sem que sejam sobrecarregados os processos de desenvolvimento utilizados. A metodologia promove a utilização de

repositórios de informação de segurança, relacionando soluções, ameaças, vulnerabilidades e políticas de segurança e encorajando a incorporação das melhores práticas nos processos de desenvolvimento.

6.2 Trabalho futuro

Relativamente ao trabalho futuro a ser desenvolvido, o primeiro passo deverá ser a aplicação da metodologia a um projeto real de desenvolvimento de software. O trabalho desenvolvido assume que uma organização que desenvolva software já possuí uma metodologia de desenvolvimento à qual poderá ser integrada a PMD. Através de um caso de estudo completo da aplicação da metodologia, poderão ser recolhidas métricas de segurança que permitam avaliar a aplicabilidade da mesma no mundo real. Recolher evidências suficientes e credíveis de que um sistema é seguro é um dos maiores desafios no desenho e gestão de sistemas de informação. Para obter tais evidências é necessário recolher informação técnica acerca da segurança do sistema durante as diversas fases do seu ciclo de vida. No entanto, seria interessante recolher métricas relativas a:

- **Exatidão em termos de segurança** – garantia de que os mecanismos de segurança foram implementados corretamente e que o sistema, os seus componentes, interfaces e dados alcançam os requisitos de segurança;
- **Eficácia em termos de segurança** – garantia de que os requisitos de segurança identificados são alcançados e que as expectativas em termos de resiliência são satisfeitas, ao mesmo tempo que o sistema não se comporta de qualquer outra forma que não a pretendida;
- **Eficiência em termos de segurança** – garantia de que os objetivos em termos de segurança foram alcançados, de acordo com as restrições em termos de recursos, tempo e custos.

Para além da recolha de métricas, deverá também ser realizado um questionário às diversas partes interessadas para avaliar a metodologia em termos de utilidade e usabilidade. Como tal, foi preparado um questionário para ser respondido por profissionais que utilizem a metodologia (o questionário pode ser consultado no **Anexo H**). Inicialmente a ideia era realizar uma apresentação oral da metodologia a profissionais da EY, experientes na área de segurança, para no seguimento da apresentação serem recolhidas as respostas. Apesar dos comentários recebidos no seguimento da apresentação poderem ser considerados bastante positivos, chegou-se à conclusão que para obter resultados suficientemente válidos para os questionários seria necessário recolher as respostas no contexto da aplicação real da metodologia. Uma vez que na EY não existem programadores, nem são desenvolvidos projetos desta natureza, não foi possível por enquanto obter os dados necessários, ficando então para trabalho futuro.

A partir da recolha de métricas de segurança e dos questionários supracitados, a metodologia poderá ser refinada a partir das lições aprendidas com os resultados obtidos.

Um outro passo que seria benéfico para o trabalho realizado é a revisão das taxonomias existentes, tanto a nível das ameaças como soluções de segurança, e nomeadamente aquelas relativas a questões de privacidade. Deste modo será também possível refinar a metodologia e os artefactos que a suportam, tornando-a o mais aproximada possível da realidade dos dias de hoje.

Bibliografia

[1] IEEE 830:1998, IEEE Recommended Practice for Software Requirements Specifications

[2] ASQ, 2018. Software Engineering Standards for Quality. http://asq509.org/ht/a/GetDocumentAction/i/130678. Acedido em 25/11/2018.

[3] ISO/IEC 12207:2008, Software life cycle processes

[4] KUMAR, R.; PANDEY, S. K.; AHSON, S. I. Security in Coding Phase of SDLC. In: 2007 Third International Conference on Wireless Communication and Sensor Networks. IEEE, 2007. p. 118-120.

[5] OWASP. 2018. What Is OWASP? What Are The OWASP Top 10?. https://www.cloudflare.com/learning/security/threats/owasp-top-10/. Acedido em 25/11/2018.

[6] OWASP. 2018. OWASP Top 10 2017. https://upload.wikimedia.org/wikipedia/mediawiki/archive/e/e9/20180111214627%21OWASP_Top-10_2017_-_Presentation.pdf. Acedido em 25/11/2018.

[7] GEER, David. Are companies actually using secure development life cycles?. Computer, 2010, 43.6: 12-16.

[8] Microsoft, 2018. Microsoft Secure Development Lifecycle (SDL). https://www.microsoft.com/en-us/sdl. Acedido em 25/11/2018.

[9] HOWARD, Michael; LIPNER, Steve. The security development lifecycle. Redmond: Microsoft Press, 2006.

[10] FONSECA, José; VIEIRA, Marco. A survey on secure software development lifecycles. In: Software Development Techniques for Constructive Information Systems Design. IGI Global, 2013. p. 57-73.

[11] E. Bodden, "State of the Systems Security," 2018 IEEE/ACM 40th International Conference on Software Engineering: Companion (ICSE-Companion), Gothenburg, 2018, pp. 550-551.

[12] ROMANOU, Anna. The necessity of the implementation of Privacy by Design in sectors where data protection concerns arise. Computer law & security review, 2018, 34.1: 99-110.

[13] ISO/IEC 27001:2013, Information technology - Security techniques - Information security management systems - Requirements

[14] ISO/IEC 27002:2013, Information technology - Security techniques - Code of practice for information security controls

[15] ISO/IEC 29100:2011, Information technology - Security techniques - Privacy framework

[16] OWASP, 2014. OWASP Top 10 Privacy Risks. https://www.owasp.org/images/3/32/Top_10_Risks.png. Acedido em 25/1/2019.

[17] DANEZIS, George, et al. Privacy and data protection by design-from policy to engineering. arXiv preprint arXiv:1501.03726, 2015.

[18] ROST, Martin; PFITZMANN, Andreas. Datenschutz-schutzziele—revisited. Datenschutz und Datensicherheit-DuD, 2009, 33.6: 353-358.

[19] HANSEN, Marit. Top 10 mistakes in system design from a privacy perspective and privacy protection goals. In: IFIP PrimeLife International Summer School on Privacy and Identity Management for Life. Springer, Berlin, Heidelberg, 2011. p. 14-31.

[20] PROBST, Thomas. Generische Schutzmaßnahmen für Datenschutz-Schutzziele. Datenschutz und Datensicherheit-DuD, 2012, 36.6: 439-444.

[21] ROST, Martin. Standardisierte Datenschutzmodellierung. Datenschutz und Datensicherheit-DuD, 2012, 36.6: 433-438.
[22] BOCK, Kirsten; MEISSNER, Sebastian. Datenschutz-Schutzziele im Recht. Datenschutz und Datensicherheit-DuD, 2012, 36.6: 425-431.
[23] BUSCHMANN, Frank. PatternOriented-Software-Architecture-A-System-of-Patterns-Volume-1. Ashish Raut, 1996.
[24] GAMMA, Erich. Design patterns: elements of reusable object-oriented software. Pearson Education India, 1995.
[25] HOEPMAN, Jaap-Henk. Privacy design strategies. In: IFIP International Information Security Conference. Springer, Berlin, Heidelberg, 2014. p. 446-459.
[26] GÜRSES, Seda; TRONCOSO, Carmela; DIAZ, Claudia. Engineering privacy by design. Computers, Privacy & Data Protection, 2011, 14.3: 25.
[27] PFITZMANN, Andreas; HANSEN, Marit. A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management. 2010.
[28] BORKING, John. Der identity protector. Datenschutz und Datensicherheit, 1996, 20.11: 654-658.
[29] VAN BLARKOM, G. W.; BORKING, John J.; OLK, JG Eddy. Handbook of privacy and privacy-enhancing technologies. Privacy Incorporated Software Agent (PISA) Consortium, The Hague, 2003, 198.
[30] AIELLO, William, et al. Just fast keying: Key agreement in a hostile internet. 2004.
[31] CHADWICK, David W. Federated identity management. In: Foundations of security analysis and design V. Springer, Berlin, Heidelberg, 2009. p. 96-120.
[32] Tom Fawcett and Foster J Provost. Combining data mining and machine learning for effective user profiling. In KDD, pages 8–13, 1996.
[33] RESCORLA, E. The Transport Layer Security (TLS) Protocol Version 1.3 (August 2018). RFC 8446.
[34] DIERKS, Tim. The transport layer security (TLS) protocol version 1.2. 2008.
[35] BARRETT, Daniel J., et al. SSH, the Secure Shell: the definitive guide. " O'Reilly Media, Inc.", 2001.
[36] DORASWAMY, Naganand; HARKINS, Dan. IPSec: the new security standard for the Internet, intranets, and virtual private networks. Prentice Hall Professional, 2003.
[37] PrivateVPN: The world’smost-trusted VPN provider. https://privatevpn.com/. Acedido em 10/02/2019.
[38] Official Website | The Best VPN Service Provider in 2019. https://nordvpn.com/ Acedido em 10/02/2019.
[39] ZIMMERMANN, Philip R.; ZIMMERMANN, Philip R. The official PGP user's guide. Cambridge: MIT press, 1995.
[40] RAMSDELL, Blake. S/MIME version 3 message specification. 1999.
[41] BORISOV, Nikita; GOLDBERG, Ian; BREWER, Eric. Off-the-record communication, or, why not to use PGP. In: Proceedings of the 2004 ACM workshop on Privacy in the electronic society. ACM, 2004. p. 77-84.
[42] Status – Private, Secure Communication. https://status.im/. Acedido em 10/02/2019.
[43] ADAMANT Messeger, Crypto Wallet & Exchange. https://adamant.im/. Acedido em 10/02/2019.
[44] CHAUM, David. Untraceable electronic mail, return addresses and digital pseudonyms. In: Secure electronic voting. Springer, Boston, MA, 2003. p. 211-219.

[45] WOODWARD, Andrew. BitLocker-the end of digital forensics?. In: Australian Digital Forensics Conference. 2006. p. 38.
[46] Endpoint Encryption Powered by PGP Technology | Symantec. https://www.symantec.com/products/endpoint-encryption. Acedido em 10/02/2019.
[47] BELLARE, Mihir, et al. Format-preserving encryption. In: International Workshop on Selected Areas in Cryptography. Springer, Berlin, Heidelberg, 2009. p. 295-312.
[48] ANDERSON, Ross; NEEDHAM, Roger; SHAMIR, Adi. The steganographic file system. In: International Workshop on Information Hiding. Springer, Berlin, Heidelberg, 1998. p. 73-82.
[49] BROZ, Milan; MATYAS, Vashek. The TrueCrypt On-Disk Format--An Independent View. IEEE Security & Privacy, 2014, 12.3: 74-77.
[50] AOL Proudly Releases Massive Amounts of Private Data | TechCrunch. https://techcrunch.com/2006/08/06/aol-proudly-releases-massive-amounts-of-user-search-data/. Acedido em 10/02/2019.
[51] MCQUAIDE, B. Identity and Access Management, Transforming E-security into a Catalyst for Competitive Advantage. Information Systems Journal, 2003, 4.
[52] HANSEN, Marit, et al. Privacy-enhancing identity management. Information security technical report, 2004, 9.1: 35-44.
[53] Shibboleth Consortium – Privacy Preserving Identity Management. https://www.shibboleth.net/. Acedido em 10/02/2019.
[54] CHAUM, David. Security without identification: Transaction systems to make big brother obsolete. Communications of the ACM, 1985, 28.10: 1030-1044.
[55] FERRAIOLO, David; CUGINI, Janet; KUHN, D. Richard. Role-based access control (RBAC): Features and motivations. In: Proceedings of 11th annual computer security application conference. 1995. p. 241-48.
[56] LIU, Simon; KUHN, Rick. Data loss prevention. IT professional, 2010, 12.2: 10-13.
[57] The 18 biggest data breaches of the 21st century | CSO Online. https://www.csoonline.com/article/2130877/the-biggest-data-breaches-of-the-21st-century.html. Acedido em 10/02/2019.
[58] GDPR: Europe Counts 65,000 Data Breach Notifications So Far. https://www.bankinfosecurity.com/gdpr-europe-counts-65000-data-breach-notifications-so-far-a-12489. Acedido em 10/02/2019.
[59] TAHBOUB, Radwan; SALEH, Yousef. Data leakage/loss prevention systems (DLP). In: 2014 World Congress on Computer Applications and Information Systems (WCCAIS). IEEE, 2014. p. 1-6.
[60] Frost & Sullivan. World Data Leakage Prevention Market. Technical Report ND34D-74, Frost & Sullivan, United States. 2008.
[61] Joshi, Hardik, and Bhadresh K. Patel. "Towards application classification with vulnerability signatures for IDS/IPS." Proceedings of the First International Conference on Security of Internet of Things. ACM, 2012.
[62] Salem, Malek Ben, Shlomo Hershkop, and Salvatore J. Stolfo. "A survey of insider attack detection research." Insider Attack and Cyber Security. Springer, Boston, MA, 2008. 69-90.
[63] What is security information and event management (SIEM)?. https://searchsecurity.techtarget.com/definition/security-information-and-event-management-SIEM. Acedido em 10/02/2019.
[64] Drum, R. (2006). IDS and IPS placement for network protection. CISSP (March 26, 2006).
[65] Davis, Noopur, et al. "Processes for producing secure software." IEEE Security & Privacy 2.3 (2004): 18-25.

[66] Uzunov, Anton V., Katrina Falkner, and Eduardo B. Fernandez. "A comprehensive pattern-oriented approach to engineering security methodologies." Information and Software Technology 57 (2015): 217-247.
[67] Fernandez, Eduardo B., and Maria M. Larrondo-Petrie. "A methodology to develop secure systems using patterns." Information Security and Ethics: Concepts, Methodologies, Tools, and Applications. IGI Global, 2008. 654-670.
[68] Gonzalez-Perez, Cesar, and Brian Henderson-Sellers. Metamodelling for software engineering. Wiley Publishing, 2008.
[69] Kolfschoten, Gwendolyn L., Robert O. Briggs, and Stephan Lukosch. "Modifiers: Increasing richness and nuance of design pattern languages." Transactions on pattern languages of programming II. Springer, Berlin, Heidelberg, 2011. 62-78.
[70] Sangwan, Raghvinder, et al. "Integrating a software architecture-centric method into object-oriented analysis and design." Journal of Systems and Software 81.5 (2008): 727-746.
[71] Fernandez, Eduardo B., Nobukazu Yoshioka, and Hironori Washizaki. "Modeling misuse patterns." 2009 International Conference on Availability, Reliability and Security. IEEE, 2009.
[72] Uzunov, Anton V., Katrina Falkner, and Eduardo B. Fernandez. "Decomposing distributed software architectures for the determination and incorporation of security and other non-functional requirements." 2013 22nd Australian Software Engineering Conference. IEEE, 2013.
[73] Uzunov, Anton V., and Eduardo B. Fernandez. "An extensible pattern-based library and taxonomy of security threats for distributed systems." Computer Standards & Interfaces 36.4 (2014): 734-747.
[74] Uzunov, Anton V., Eduardo B. Fernandez, and Katrina Falkner. "ASE: A comprehensive pattern-driven security methodology for distributed systems." Computer Standards & Interfaces 41 (2015): 112-137.
[75] Firesmith, Donald. "Engineering security requirements." Journal of object technology 2.1 (2003): 53-68.
[76] Swiderski, Frank, and Window Snyder. Threat modeling. Microsoft Press, 2004.
[77] Opdahl, Andreas L., and Guttorm Sindre. "Experimental comparison of attack trees and misuse cases for security threat identification." Information and Software Technology 51.5 (2009): 916-932.
[78] Tøndel, Inger Anne, Jostein Jensen, and Lillian Røstad. "Combining misuse cases with attack trees and security activity models." 2010 International Conference on Availability, Reliability and Security. IEEE, 2010.
[79] Dowd, Mark, John McDonald, and Justin Schuh. The art of software security assessment: Identifying and preventing software vulnerabilities. Pearson Education, 2006.
[80] Sourya Joyee De, Daniel Le Métayer. "Privacy Risk Analysis". Synthesis Lectures on Information Security, Privacy, and Trust (2016): 1-133.
[81] A Guide to SIEM and Log Management Solutions | BlackStratus https://www.blackstratus.com/siem-log-management-solutions/. Acedido em 10/02/2019.
[82] GDPR Logging Requirements: All you need to know to comply – LogSentinel. https://logsentinel.com/gdpr-logging-requirements/. Acedido em 10/02/2019.
[83] Chandra, Pravir. "Software assurance maturity model." A guide to building security into software development v1. 0, OWASP Project (2008).
[84] Software Security Assessment Report | BSIMM. https://www.bsimm.com/download.html. Acedido em 15/07/2019.
[85] Potter, B. (2009). Microsoft SDL threat modelling tool. *Network Security*, *2009*(1), 15-18.

[86] Regulamento (UE) 2016/679 do Parlamento Europeu e do Conselho, de 27 de abril de 2016, relativo à proteção das pessoas singulares no que diz respeito ao tratamento de dados pessoais e à livre circulação desses dados e que revoga a Diretiva 95/46/CE (Regulamento Geral sobre a Proteção de Dados). https://protecao-dados.pt/wp-content/uploads/2017/07/Regulamento-Geral-Prote%C3%A7%C3%A3o-Dados.pdf. Acedido em 15/07/2019.
[87] Resolução de Conselho de Ministros nº 41/2018. https://dre.pt/application/conteudo/114937034. Acedido em 15/07/2019.
[88] Brooke, John. "SUS-A quick and dirty usability scale." Usability evaluation in industry 189.194 (1996): 4-7.

Anexo A – ISO/IEC 27002:2013 – Cláusula 14

14. Aquisição, desenvolvimento e manutenção de sistemas
14.1 Requisitos de Segurança para os sistemas de informação **Objetivo:** garantir que a segurança da informação é parte integrante de todo o ciclo de vida dos sistemas de informação, incluindo também os requisitos para sistemas de informação que fornecem serviços sobre as redes públicas.
14.1.1 Especificação e análise de requisitos de segurança da informação Os requisitos relacionados com segurança da informação devem ser incluídos nos requisitos para novos sistemas de informação ou melhorias dos sistemas de informação existentes.
14.1.2 Proteger serviços aplicacionais nas redes públicas As informações envolvidas nos serviços de aplicação que transitam sobre redes públicas devem ser protegidas de atividades fraudulentas, disputas contratuais e divulgação e modificações não autorizadas.
14.1.3 Proteger transações de serviços aplicacionais A informação envolvida em transações deve ser protegida para prevenir transmissões incompletas, erros de encaminhamento, alteração não autorizada da mensagem, divulgação não autorizada, duplicação ou reapresentação da mensagem não autorizada.
14.2 Segurança nos processos de desenvolvimento e suporte **Objetivo:** garantir que a segurança da informação é considerada e implementada no desenvolvimento do ciclo de vida dos sistemas de informação.
14.2.1 Política de desenvolvimento seguro As regras para o desenvolvimento de sistemas de software devem ser estabelecidas e aplicadas ao desenvolvimento realizado dentro da organização.
14.2.2 Procedimentos para controlo de alterações aos sistemas As mudanças em sistemas durante o seu ciclo de vida de desenvolvimento devem ser controladas utilizando procedimentos formais de controlo de alterações.
14.2.3 Revisão técnica de aplicações após alterações na plataforma de produção Quando existirem modificações em plataformas operacionais, convém que as aplicações críticas de negócio sejam analisadas e testadas para assegurar que não ocorreu nenhum impacto adverso nas operações da organização ou na segurança da informação.
14.2.4 Restrições sobre alterações em pacotes de software As modificações em pacotes de software devem ser desencorajadas e estar limitadas às mudanças necessárias, e todas as mudanças devem ser estritamente controladas.
14.2.5 Princípios de engenharia de sistemas seguros Devem ser estabelecidos, documentados, mantidos e aplicados princípios para projetar sistemas seguros qualquer implementação de sistemas de informação.
14.2.6 Ambiente de desenvolvimento seguro As organizações devem estabelecer e proteger adequadamente os diferentes ambientes de desenvolvimento ao longo de todo o ciclo de vida de desenvolvimento de um sistema.

14.2.7 Desenvolvimento subcontratado A organização deve supervisionar e monitorizar as atividades subcontratadas de desenvolvimento de sistemas.
14.2.8 Testes de segurança de sistemas Os testes de funcionalidades de segurança devem ser realizados durante o desenvolvimento de sistemas.
14.2.9 Testes de aceitação de sistemas Devem ser estabelecidos programas de testes de aceitação e respetivos critérios para novos sistemas de informação, atualizações e novas versões.
14.3 Dados para teste **Objetivo:** assegurar a proteção dos dados utilizados para testes.
14.3.1 Proteção dos dados de teste Os dados utilizados para testes devem ser cuidadosamente selecionados, protegidos e controlados.

Adaptado a partir de [14].

Anexo B – OWASP Top 10 2017

OWASP Top 10	
1. Injeção de código	Ataques de injeção de código consistem em enviar dados não confiáveis para serem interpretados, através de um formulário ou outro método de submissão de dados para uma aplicação web. Entre todos os tipos de injeção, o ataque mais comum é a injeção de uma *query* SQL num formulário onde supostamente é inserido texto. Se não forem tomadas medidas de segurança o código SQL poderá ser executado resultando em fuga de informação ou comprometimento do sistema. Este tipo de ataque pode ser prevenido validando ou sanitizando os dados submetidos ou até acrescentando controlos na própria base de dados.
2. Quebra de autenticação	Vulnerabilidades nos sistemas de autenticação poderão levar um atacante a ganhar acesso a contas de utilizadores ou até mesmo contas com privilégios de administração. Por exemplo, um atacante poderá utilizar uma lista contendo milhares de combinações de utilizadores e palavras-chave. Algumas das estratégias utilizadas para mitigar este tipo de vulnerabilidade passam por utilizam autenticação com 2 fatores (2FA) ou então limitar o número de tentativas de autenticação sucessivas.
3. Exposição de dados sensíveis	No caso das aplicações web não protegerem devidamente dos dados sensíveis nelas contidos, como por exemplo palavras-chave ou dados financeiros, um atacante poderá ganhar acesso a esses dados e deste modo utiliza-los para fins maliciosos. Um método para obter informação sensível é conhecido como *"man-in-the-middle*". A exposição de dados sensíveis pode ser minimizada ao cifrar os dados assim como desativar funcionalidades de *caching* para informação sensível. Para além disso, deverão ser tidos cuidados para que não seja guardada informação sensível desnecessária.
4. Entidades externas de XML	Este é um tipo de ataque que afeta aplicações que analisem dados XML, uma vez que estes dados podem fazer referência a entidades externas, como por exemplo uma unidade de armazenamento, o que poderá levar a que dados sejam enviados para entidades externas não autorizadas. A melhor forma de prevenir este tipo de ataque é utilizando outros formatos de dados menos complexos, como JSON, ou no mínimo desativando o uso de entidades externas numa aplicação que utilize XML.
5. Quebra de controlo de acesso	Um sistema de controlo de acesso define quem pode aceder a uma determinada informação ou funcionalidade. Quando existe uma quebra neste controlo de acesso, um atacante poderá realizar tarefas como se fosse um utilizador privilegiado. Por exemplo, uma aplicação pode permitir alterar entre diferentes contas autenticadas alterando parte do *url*, sem que seja feita nenhuma verificação adicional. Para garantir um controlo de acessos seguro a aplicação web poderá recorrer ao uso de *tokens* de autenticação, que terão de ser apresentados cada vez que é feito um pedido privilegiado.

6. Configurações de segurança incorretas	Esta é a vulnerabilidade mais comum desta lista e é muitas vezes resultado de serem utilizadas configurações padrão ou serem mostradas mensagens de erro em excesso. Por exemplo, uma aplicação poderá revelar vulnerabilidades ao apresentar erros demasiado descritivos. Este risco pode ser mitigado ao remover do código quaisquer funcionalidades não utilizadas e garantindo que apenas são mostradas mensagens de erro genéricas.
7. *Cross-site Scripting*	Este tipo de vulnerabilidades existe quando aplicações web permitem que seja adicionado código a um caminho *url* ou a uma página que será vista por terceiros. Deste modo um atacante poderá executar um *script* no *browser* de uma vítima, utilizando o *url* de uma página que à partida seria de confiança. Para mitigar este tipo de vulnerabilidade, para além da utilização de *frameworks* de desenvolvimento mais modernas, deverão ser ignorados pedidos HTTP que não sejam de confiança, assim como validar e sanitizar conteúdos gerados pelos utilizadores.
8. Desserialização insegura	Esta é uma ameaça às aplicações web que serializam e desserializam dados com frequência. A exploração desta vulnerabilidade é o resultado da desserialização de dados provenientes de fontes não confiáveis, o que poderá levar a consequências como DoS ou execução de código remoto. Poderão ser tomadas medidas para tentar identificar este tipo de ataque, monitorizando e validando a desserialização, no entanto a única forma segura de evitar este tipo de ataque é proibindo a desserialização de dados de fontes que não sejam consideradas seguras.
9. Utilização de componentes com vulnerabilidades conhecidas	É relativamente comum que no desenvolvimento de aplicações web sejam utilizados componentes como *frameworks* ou bibliotecas. Estes componentes permitem evitar que seja feito trabalho redundante acrescentando as funcionalidades necessárias. Alguns atacantes procuram vulnerabilidades nesses componentes de forma a poderem realizar ataques. Alguns dos componentes mais comuns são utilizados por milhares de aplicações web, o que leva a que a descoberta de uma vulnerabilidade num destes componentes deixe todas estas aplicações vulneráveis. Para minimizar o risco da utilização de componentes com vulnerabilidades conhecidas deverão ser removidos todos os componentes que não sejam utilizados, assim como garantir que os componentes utilizados são provenientes de fontes confiáveis assim como estão na sua versão mais atual.
10. Registos e monitorização insuficientes	Muitas das aplicações web não tomam as medidas necessárias para evitar fugas de informação. O tempo médio que leva para que uma fuga de informação seja descoberta é por volta dos 200 dias após o acontecimento. Este intervalo de tempo permite aos atacantes causar elevados danos antes que exista algum tipo de resposta. Uma forma de mitigar este risco é introduzindo funcionalidades de registos (*logging*) e monitorização, assim como planos de resposta a incidentes que garantam a consciencialização sobre eventuais ataques à aplicação.

Adaptado a partir de [5].

Anexo C – Microsoft SDL

Microsoft SDL	
Fase de treino	
Core Security Training	Esta prática é um pré-requisito para a implementação do SDL. Deverão ser tidos em conta conceitos fundamentais para construir melhor software, incluindo desenho seguro, modelagem de ameaças, código seguro, testes de segurança e melhores práticas sobre privacidade.
Fase de levantamento de requisitos	
Estabelecer requisitos de segurança e privacidade	A definição e integração de requisitos de segurança e privacidade nas fases iniciais ajuda a identificar objetivos essenciais e entregáveis, assim como minimizar os desvios do planeamento e calendário.
Criação de critérios de qualidade	A definição de níveis mínimos de segurança e privacidade desde o início ajudam as equipas a perceber os riscos associados com os problemas de segurança, a identificar e corrigir erros de segurança durante o desenvolvimento e a aplicar as normas ao longo de todo o projeto.
Realização de avaliações de riscos de segurança e privacidade	Examinar o desenho do software com base nos custos e requisitos regulatórios ajudam a equipa a identificar qual a parte do projeto que irá precisar que seja feita modelagem de ameaças e revisões de segurança antes do seu lançamento, assim como determinar o impacto a nível da privacidade de uma determinada funcionalidade, produto ou serviço.
Fase de desenho	
Estabelecer requisitos do desenho	Considerar as preocupações de segurança e privacidade desde o início ajuda a minimizar o risco de não cumprimento do calendário previsto assim como reduzir os custos do projeto.
Análise/redução da superfície de ataque	Reduzir as oportunidades de um atacante explorar um potencial ponto fraco ou vulnerabilidade requer uma análise sobre a superfície de ataque e inclui desativar ou restringir o acesso a serviços do sistema, assim como aplicar o princípio do privilégio mínimo e aplicar camadas de defesa sempre que possível.
Modelagem de ameaças	Aplicar uma abordagem estruturada a cenários de ameaça durante a fase de desenho ajuda uma equipa a identificar vulnerabilidades de segurança de uma forma mais eficiente e menos dispendiosa, assim como determinar os riscos para essas ameaças e estabelecer ações de mitigação apropriadas.
Fase de implementação	
Utilização de ferramentas aprovadas	Publicar uma lista de ferramentas aprovadas e respetivas configurações de segurança ajudam a automatizar e reforçar as práticas de segurança de forma fácil e sem custos. Ao manter essa lista atualizada regularmente é possível utilizar sempre as versões mais atuais das ferramentas e incluir novas funcionalidades de análise e proteções de segurança.
Retirar funcionalidades inseguras	Analisando todas as funcionalidades e APIs de um projeto e banindo todas as funções consideradas inseguras é possível reduzir os potenciais riscos de segurança com custos bastante reduzidos. Ações especificas incluem a

	utilização de compiladores mais recentes, ou ferramentas de análise de código para verificar se existem funções da lista de funções banidas, para depois poderem ser substituídas por alternativas mais seguras.
Realização de análises estáticas	Analisar o código fonte de um programa é um método escalável para revisões de segurança e ajuda a garantir que as normas de desenvolvimento seguro estão a ser cumpridas.
Fase de verificação	
Realização de análises dinâmicas	Realização de verificações em tempo de execução nas funcionalidades do software utilizando ferramentas que monitorizam o comportamento da aplicação relativo a corrupções de memória, problemas com privilégios dos utilizadores e outros problemas críticos de segurança.
Testes de Fuzzing	Induzir o programa em erro ao introduzir deliberadamente dados mal formados ou aleatórios na aplicação ajuda a revelar potenciais problemas de segurança antes do seu lançamento representando um investimento relativamente baixo.
Revisão da superfície de ataque	A revisão da superfície de ataque após o código estar completo ajuda a garantir que qualquer alteração feita no desenho ou implementação foram tidas em conta, e que quaisquer novos vetores de ataque criados como resultado das alterações foram revistos e seus riscos mitigados, incluindo a criação de modelos de ameaça.
Fase de lançamento	
Criação de um plano de resposta a incidentes	A preparação de um plano de resposta a incidentes é crucial para ajudar a endereçar novas ameaças que apareçam com o tempo. Este plano deve incluir a identificação dos contactos de emergência apropriados assim como estabelecer os planos de segurança para partes do código provenientes de outros grupos dentro da organização ou código de terceiros licenciado.
Revisão final de segurança	Deliberadamente rever todas as atividades de segurança que foram realizadas ajuda a garantir a prontidão do software a ser lançado. A revisão final de segurança por norma inclui a revisão dos modelos de ameaça, outputs das ferramentas e testes de desempenho relativos aos critérios de segurança definidos na fase de levantamento de requisitos.
Lançamento certificado e arquivo	A certificação do software após o seu lançamento ajuda a garantir que os requisitos de segurança e privacidade foram alcançados. Arquivar todos os dados pertinentes é essencial para a realização de tarefas após o lançamento e ajuda a baixar os custos a longo prazo associados com a manutenção do software.
Fase de Resposta	
Execução do plano de resposta a incidentes	Ter a capacidade de implementar o plano de resposta a incidentes estabelecido na fase de lançamento é essencial para ajudar a proteger os clientes das vulnerabilidades de segurança ou privacidade que possam surgir.

Adaptado a partir de [8].

Anexo D – Mapeamento entre camadas funcionais e taxonomia de ameaças.

Functionality decomposition layer	Relevant threat classes
User interaction	Identity attacks, Passing illegal data, Remote information inference, Repudiation, Uncontrolled operations
Data/storage management	Passing illegal data, Stored data attacks, Remote information inference, Uncontrolled operations
Resource management	Uncontrolled operations
Distribution control	Identity attacks, Passing illegal data, Remote information inference, Uncontrolled operations
Communication	Network communication attacks, Network protocol attacks, Repudiation
Addressing	Network communication attacks, Network protocol attacks, Repudiation

Obtido a partir de [73].

Anexo E – Mapeamento entre realizações técnicas abstratas e taxonomia de ameaças

Technical realization abstraction	Relevant (first level) threat patterns from Uzunov and Fernandez's base threat taxonomy [73]
User interaction	
Input ports	Identity spoofing, Advantageous identity allocation, Injection, Repudiation, Unauthorized access, Spoofing privileged processes, Exploitation of tight component coupling, Process overflow attack
Output ports	Output information disclosure, Data inference
Data/storage management	
Data structures	Injection, Corruption, Data inference, Session state poisoning, Process overflow attack
Storage abstractions	Injection, Corruption, Data inference, Unauthorized access, Resource exhaustion
Database systems	Injection, Corruption, Track erasing, Data inference, Unauthorized access, Resource exhaustion
File systems	Injection, Corruption, Data inference, Track erasing, Unauthorized access, Resource exhaustion
Resource management	
Resources	Unauthorized access, Invoking unauthorized operations, Resource exhaustion, Targeted process crashing, Covert network channels
Algorithms	Invoking unauthorized operations, Spoofing privileged processes, Exploitation of tight component coupling, Exploiting concurrency flaws, Resource exhaustion
Distribution control	
Software component interfaces	Identity spoofing, Injection, Probing, Output information disclosure, Invoking unauthorized operations, Spoofing privileged processes, Exploitation of tight component coupling, Process overflow attack, Exploiting concurrency flaws
Operations	Advantageous identity allocation, Output information disclosure, Invoking unauthorized operations, Exploitation of tight component coupling, Exploiting concurrency flaws
Execution abstractions	Identity spoofing, Injection, Invoking unauthorized operations, Spoofing privileged processes, Exploitation of tight component coupling, Process overflow attack, Exploiting concurrency flaws, Resource exhaustion, Targeted process crashing
Processes (OS)	Identity spoofing, Injection, Probing, Output information disclosure, Unauthorized access, Invoking unauthorized operations, Spoofing privileged processes, Unsafe code execution,

	Exploitation of tight component coupling, Process overflow attack, Exploiting concurrency flaws, Resource exhaustion, Targeted process crashing
Communication	
Messages	Message secrecy violation, Message integrity violation, Message authenticity violation, Injection, Repudiation, Covert network channels
Message channels	Message secrecy violation, Message integrity violation, Message authenticity violation, Traffic analysis, Session hijacking, Session state poisoning, Message flooding, Scanning, Probing, Covert network channels
Protocols	Message replay, Message re-use, Protocol field modification, Use of abnormal packet sizes, Use of abnormal packet sequencing, Use of reserved protocol packets, Protocol initial/end state exploitation, Data inference, Repudiation
Networking infrastructure (S/W)	All the threat patterns in the Network communication attacks and Network protocol attacks classes are relevant, considered this time at the lower network layers.
Addressing	
Addresses/identifiers	Identity spoofing, Advantageous identity allocation
Protocols/algorithms	Identity spoofing, Advantageous identity allocation, Route poisoning, Message replay, Message re-use, Protocol field modification, Repudiation
Routing data structures (tables)	Identity spoofing, Advantageous identity allocation, Route poisoning

Obtido a partir de [73].

Anexo F – Mapeamento entre classes de requisitos de segurança e políticas de segurança genéricas

Requirement class	Generic security policies (from [73])
Identification & authentication requirements	Identity management
Authorization & trust requirements	Authorization, Execution control
Confidentiality requirements	Secure communications, Identity management, Authorization, Storage security
Integrity requirements	Secure communications, Authorization, Execution control, Storage security
Immunity requirements	Execution control, Filtering
Exportability (code mobility) requirements	Authorization, Execution control
Privacy requirements	Secure communications, Identity management
Nonrepudiation requirements	Secure communications, Identity management, Storage security, Logging and monitoring
Security auditing requirements	Logging and monitoring
Credentials & security information requirements	Security information management
Intrusion detection requirements	Logging and monitoring, Filtering, Execution control
Availability & survivability requirements	Storage security, Logging and monitoring, Secure communications, Execution control
System maintenance security requirements	Logging and monitoring, Authorization, Secure communications

Adaptado a partir de [73].

Anexo G – Avaliação da metodologia proposta

#	Requisito Geral		#	Requisito Especificos	Racional
#01	**As aplicações cliente (exemplo, Android, IOS, WEB) devem ser desenvolvidas adotando práticas de desenvolvimento seguro.**	**FE**	**#01.1**	Seguir as boas práticas de desenvolvimento. **Exemplo:** Open Web Application Security Project (OWASP), no que respeita ao desenvolvimento de código seguro e de submissão desse código a testes de segurança.	Sendo a PMD uma metodologia de segurança a ser integrada nas fases genéricas do ciclo de desenvolvimento, e uma vez que a mesma considera para cada uma das diferentes fases as atividades de segurança necessárias (i.e., boas práticas de desenvolvimento), então podemos considerar que a PMD segue as boas práticas de desenvolvimento. Em contraste com o exemplo apresentado, a PMD tem em conta o desenvolvimento de código seguro (ver secção 4.2.3 – 3.1.1.) e a submissão desse código a testes de segurança (ver secção 4.2.3 - 3.2.).
			#01.2	Utilização de sessões seguras com protocolos de Segurança.	Uma sessão segura é um mecanismo que garante a segurança qualquer tipo de comunicação através de uma rede, seja ela pública ou privada, incluindo a internet (ver secção 3.2.2 para mais detalhe sobre comunicações seguras). Sendo a PMD uma metodologia que recorre a um conjunto de soluções de segurança, e sendo as "comunicações seguras" uma dessas soluções de segurança, podemos então considerar que a PMD garante a utilização de sessões seguras com protocolos de segurança (ver secção 4.2.2).
			#01.4	Não guardar informação pessoal no browser, memória ou disco, para além do tempo da sessão e apenas na medida do necessário.	Uma vez que a PMD garante a utilização de sessões seguras (cf. #01.2) e que para além disso considera a solução de segurança "armazenamento seguro" (ver secção 4.2.2), podemos então considerar que a PMD garante que não será guardada informação pessoal no browser (e.g., no formato de cookies) para além do tempo da sessão e apenas na medida do necessário.
		App	**#01.5**	Utilização de sessões seguras com protocolo de Segurança.	(cf. #01.2)

		BD	**#01.11**	Comunicação com camada aplicacional através de autenticação por certificado válido por período não superior a 2 anos, no caso de as camadas serem física ou logicamente distintas. **Exemplo:** padrão X.509, da ITU-T para Infraestruturas de Chaves Públicas (ICP).	Uma vez que a solução de segurança "comunicações seguras" encapsula *security patterns* que permitem que duas ou mais partes comuniquem de forma segura, e que a camada aplicacional e a base de dados são exemplos de partes comunicantes, podemos então considerar que a PMD garante a comunicação segura entre a camada aplicacional e a base de dados, sendo o padrão X.509 um exemplo de uma implementação concreta de uma solução.
			#01.12	Prever cifra de informação pessoal (recomenda-se mínimo 2048 bit) apenas se a aplicação cliente tiver camada de BD física e logicamente distinta, usando preferencialmente tecnologia que permita interoperabilidade entre sistemas.	Uma vez que a solução de segurança "armazenamento seguro" encapsula *security patterns* para o armazenamento seguro de informação (ver secção 4.2.2), e que armazenamento seguro de dados pessoais passa pela cifra dos mesmos, podemos então considerar que a PMD considera o referido (ver secção 3.2.3 para mais detalhe sobre armazenamento seguro).
#02	**Capacidade para autenticar e autorizar todos os utilizadores e dispositivos, incluindo o controlo do acesso a sistemas e aplicações.**	**FE**	**#02.1**	O processo de autenticação deve ser sempre iniciado e mantido em sessão segura.	Uma vez que a solução de segurança "gestão de identidades" (ver secção 4.2.2) encapsula *security patterns* que permitem a validação da identidade de um utilizador (i.e., autenticação - ver secção 3.2.1 para mais detalhe sobre autenticação), e que a PMD garante a utilização de sessões seguras (cf. #01.2), podemos então considerar que a PMD garante que o processo de autenticação é iniciado e mantido em sessão segura.
			#02.3	Dados pessoais de sessão excluídos das variáveis Uniform Resource Locator (URL) ou de outras variáveis visíveis ao utilizador.	Uma vez que a PMD garante a utilização de sessões seguras (cf. #01.2), e que o facto de os dados pessoais estarem guardados em variáveis visíveis ao utilizador não é considerado seguro, podemos então considerar que a PMD garante a exclusão de dados pessoais de sessão das variáveis URL ou de outras variáveis visíveis.
			#02.4	Credenciais de início de sessão transmitidos através do seu HASH, mínimo Secure Hash Algorithm- 256 (SHA-256), ou utilização de cifra ou codificação para a transmissão de dados pessoais (nome do utilizador e palavra-passe em HASH e restantes dados cifrados).	Uma vez que a PMD garante que o processo de autenticação é mantido em sessão segura (cf. #02.1), que a solução de segurança "comunicações seguras" encapsula *security patterns* que permitem a comunicação segura entre duas partes, e que a transmição de credenciais deverá ser feita através do seu HASH (mínimo SHA-256) para ser considerada segura, podemos então considerar que a PMD garante que as credenciais de início de sessão são trasmitidas através do seu HASH e que a restante informação é cifrada.

			#02.5	Sempre que aplicável, a palavra-passe deve ter no mínimo 9 caracteres (13 caracteres para utilizadores com acesso privilegiado) e ser complexa. A sua composição deverá exigir a inclusão de 3 dos 4 seguintes conjuntos de caracteres: letras minúsculas (a…z), letras maiúsculas (A…Z), números (0…9) e caracteres especiais (~! @ # $ % ^ & * () _ + | `- = \ {} []:"; '<>?,. /). Poderá, em alternativa, ser constituída por frases ou excertos de texto longo conhecidos pelo utilizador, sem caracter de "espaço".	Uma vez que a solução de segurança "gestão de identidades" (ver secção 4.2.2) encapsula security patterns que permitem a validação da identididade de um utilizador (i.e., autenticação - ver secção 3.2.1 para mais detalhe sobre autenticação), e que para este processo ser considerado seguro existem requisitos mínimos para as credenciais utilizadas, podemos então considerar que a PMD garante que as credenciais utilizadas cumprem com os requisitos mínimos de segurança, correspondendo a sua complexidade ao que é considerado atualmente como boa prática (considerando o risco associado a cada elemento que se autentica).
		App	**#02.7**	A palavra-passe dos administradores deve ter no mínimo 13 caracteres e ser complexa. Neste caso, a sua composição deverá exigir a inclusão de 3 dos 4 seguintes conjuntos de caracteres: letras minúsculas (a…z), letras maiúsculas (A…Z), números (0…9) e caracteres especiais (~! @ # $ % ^ & * () _ + | `- = \ {} []:"; '<>?,. /). Poderá, em alternativa, ser constituída por frases ou excertos de texto longo conhecidos pelo utilizador, sem caracter de "espaço".	(cf. #02.5)
			#02.8	Para todos os administradores deve-se utilizar Padrão de autenticação 2FA: **Exemplos:** - Palavra-passe + Smartcard - Palavra-passe + Biometria - Palavra-passe + certificado (por exemplo X.509, da ITU-T para ICP, válido por período não superior a 2 anos).	(cf. #02.5)
			#02.10	Comunicação com camadas FE ou BD através de sessão segura, com prévia autenticação se camadas forem física ou logicamente distintas.	(cf. #01.2 e #02.1)
			#02.13	Autenticação de elementos comunicantes garantida por validação de informação estática ao nível da rede. **Exemplos:** 1) utilização de IP fixo + *hostname* + *MacAddress* + fatores de autenticação, ou 2) Utilização de certificados	(cf. #02.10)
		BD	**#02.14**	A palavra-passe deve ter no mínimo 13 caracteres e ser complexa. Neste caso, a sua composição deverá exigir a inclusão de 3 dos 4 seguintes conjuntos de caracteres: letras minúsculas (a...z), letras maiúsculas (A...Z), números (0...9) e caracteres especiais (~! @ # $ % ^ & * () _ + | `- = \ {} []:"; '<>?,. /). Poderá, em alternativa, ser constituída por frases ou excertos de texto longo conhecidos pelo utilizador, sem caracter de "espaço".	(cf. #02.5)

#03	**Atribuição de direitos de acesso e privilégios de forma restrita e controlada**	**FE**	**#03.1**	Criação perfis com privilégios mínimos, onde cada tipo de perfil é definido em função do Tipo de Dado Pessoal a que acede e Ação que pode efetuar sobre o Dado Pessoal (Create, Read, Update, Delete - CRUD), de acordo com o princípio da necessidade de conhecer.	Uma vez que a solução de segurança "gestão de identidades" (ver secção 4.2.2) encapsula *security patterns* que permitem a atribuição de perfis de acordo com as necessidades de cada utilizador (ver secção 3.2.4 para mais detalhe sobre Role Based Access Control), e que a atribuição de privilégios para além daqueles considerados os necessários é considerado inseguro, podemos então considerar que a PMD garante a criação de perfis com pribilégios mínimos e de acordo com a necessidade de conhecer.
			#03.2	Criação de registo de acesso, alteração e remoção (logs), com informação sobre quem acedeu, de onde acedeu (IP e Porto), quando acedeu, a que dados acedeu, que ação foi efetuada sobre os mesmos (CRUD).	Uma vez que a solução de segurança "logging e monitorização" (ver secção 4.2.2) encapsula *security patterns* para o registo e monitoização de eventos em um sistema (ver secção 3.2.5 para mais detalhe sobre registos e monitorização), e que os acessos à informação presente no sistema (i.e., tratamento da informação) são um exemplo de um evento, podemos então considerar que a PMD garante o registo dos acessos, alterações e remoção de dados, incluindo quem acedeu, de onde acedeu, quando acedeu, a que dados acedeu e a ação efetuada sobre os mesmos.
		App	**#03.3**	Criação perfis com privilégios mínimos, onde cada tipo de perfil é definido em função do Tipo de Dado Pessoal a que acede e Ação que pode efetuar sobre o Dado Pessoal (CRUD), de acordo com o princípio da necessidade de conhecer.	(cf. #03.1)
			#03.4	Criação de registo de acesso, alteração e remoção (logs), com informação sobre quem acedeu, de onde acedeu (IP e Porto), quando acedeu, a que dados acedeu, que ação foi efetuada sobre os mesmos (CRUD).	(cf. #03.2)
		BD	**#03.5**	Criação perfis com privilégios mínimos, onde cada tipo de perfil é definido em função do Tipo de Dado Pessoal a que acede e Ação que pode efetuar sobre o Dado Pessoal (CRUD), de acordo com o princípio da necessidade de conhecer.	(cf. #03.1)
			#03.6	Criação de registo de acesso, alteração e remoção (logs), com informação sobre quem acedeu, de onde acedeu (IP e Porto), quando acedeu, a que dados acedeu, que ação foi efetuada sobre os mesmos (CRUD).	(cf. #03.2)
#04	**Atribuição das credenciais de acesso de forma controlada através de um processo formal de gestão do respetivo ciclo de vida.**	**FE**	**#04.1**	Processo definido de acordo com a política de “Atribuição de direitos de acesso e privilégio de forma restrita e controlada”.	Uma vez que a solução de segurança "gestão de identidades" (ver secção 4.2.2) encapsula *security patterns* que permitem a atribuição e remoção de direitos de acesso a cada utilizador, que a solução "autorização" (ver secção 4.2.2) encapsula *security patterns* que permitem a construção de modelos conceptuais de autorização, e que para garantir que a atribuição e remoção é feita de forma segura deverá existir um processo definido associado, podemos então considerar que a PMD garante a atribuição de direitos de acesso e privilégios de forma restrita e controlada.

			#04.2	Atribuição de credenciais de acesso efetuada de forma a permitir a sua auditoria, sem permitir outro acesso que não o do destinatário da informação. **Exemplo:** - Envio de informação de autenticação por SMS com validade limitada (não superior a 5 minutos), com primeiro acesso a implicar sempre a redefinição da informação enviada; - Envio de informação de autenticação gerada automática e aleatoriamente, enviada por Envelope (semelhante ao do envio de dados do Cartão de Cidadão).	Uma vez que a solução de segurança "gestão de informação de segurança" encapsula *security patrterns* relacionados com a gestão da informação de segurança, e que as credenciais de acesso são um exemplo de um tipo de informação de segurança, podemos então considerar que a PMD garante a atribuição de credenciais de acesso e forma a permitir a sua auditoria, e sem permitir outro acesso que não o do destinatário da informação.
		App	**#04.3**	Processo definido de acordo com a política de "Atribuição de direitos de acesso e privilégio de forma restrita e controlada".	(cf. #04.1)
			#04.4	Atribuição de credenciais de acesso efetuada de forma a permitir a sua auditoria, sem permitir outro acesso que não o do destinatário da informação.	(cf. #04.2)
		BD	**#04.5**	Processo definido de acordo com a política de "Atribuição de direitos de acesso e privilégio de forma restrita e controlada".	(cf. #04.1)
			#04.6	Atribuição de credenciais de acesso efetuada de forma a permitir a sua auditoria, sem permitir outro acesso que não o do destinatário da informação.	(cf. #04.2)
#05	**Revisão de direitos de acesso de utilizadores em intervalos regulares.**	**FE**	**#05.1**	Processo de renovação de conta do utilizador de acordo com os mesmos requisitos de segurança da criação do mesmo, não devendo ter um ciclo de vida superior a 180 dias.	(cf. #04.1)
			#05.2	A gestão do ciclo de vida da conta do utilizador deve ter em conta a segregação das funções existentes e os privilégios de acesso que devem estar associados a essas funções, em cada momento (privilégios mínimos, onde cada tipo de conta é definido em função do Tipo de Dado Pessoal a que acede e Ação que pode efetuar sobre o Dado Pessoal (CRUD), de acordo com o princípio da necessidade de conhecer.	(cf. #03.1, #04.1)
		App	**#05.5**	Processo de gestão de validade de perfis.	(cf. #03)
			#05.7	Processo automatizado ou interoperável com sistemas responsáveis pela gestão das funções associados aos privilégios atribuídos a cada perfil. Em casos de verificação assíncrona do binómio função/privilégios, a mesma deve ocorrer com uma periodicidade, no máximo bimestral ou quando se verifique uma alteração no mapa de pessoal associado a esta função.	(cf. #03)
		BD	**#05.9**	Processo de gestão de validade de perfis.	(cf. #03)

			#05.11	Processo automatizado ou interoperável com sistemas responsáveis pela gestão das funções associados aos privilégios atribuídos a cada perfil. Em casos de verificação assíncrona do binómio função/privilégios, a mesma deve ocorrer com uma periodicidade, no máximo bimestral ou quando se verifique uma alteração no mapa de pessoal associado a esta função.	(cf. #03)
#06	**Capacidade para garantir que os utilizadores fazem uma utilização correta dos dados.**	**FE**	**#06.1**	A gestão do ciclo de vida da conta do utilizador deve ter em conta a segregação das funções existentes e os privilégios de acesso que devem estar associados a essas funções, em cada momento (privilégios mínimos, onde cada tipo de conta de utilizador é definido em função do Tipo de Dado Pessoal a que acede e Ação que pode efetuar sobre o Dado Pessoal (CRUD), de acordo com o princípio da necessidade de conhecer.	(cf. #03.1, #04.1)
			#06.3	Ação dos utilizadores sobre dados pessoais (CRUD) deve permitir a sua auditoria em registo de atividade (logs).	(cf. #03.2)
		App	**#06.4**	Para Administradores de Sistemas, Redes e Aplicacional, caso acedam a dados pessoais, aplicam-se os requisitos da camada FE.	(cf. #06.1 e #06.3)
			#06.5	Processo de gestão de validade de contas de utilizadores.	(cf. #05)
			#06.7	Processo automatizado ou interoperável com sistemas responsáveis pela gestão das funções associados aos privilégios atribuídos a cada perfil. Em casos de verificação assíncrona do binómio função/privilégios, a mesma deve ocorrer com uma periodicidade limitada.	(cf. #03)
		BD	**#06.10**	Para Administradores de Bases de Dados, Administradores de Sistemas, de Redes e Aplicacional, caso acedam a dados pessoais, aplicam-se os requisitos da camada FE.	(cf. #06.1 e #06.3)
			#06.11	Processo de gestão de validade das contas dos utilizadores.	(cf. #05)
#07	**Restrição de acesso à informação baseado no princípio necessidade de conhecer (criação de perfil).**	**FE**	**#07.1**	Associação da tipologia de dados a perfis específicos, individuais e associados à função, com privilégios mínimos, onde cada tipo de perfil é definido em função do Tipo de Dado Pessoal a que acede e Ação que pode efetuar sobre o Dado Pessoal (CRUD), de acordo com o princípio da necessidade de conhecer.	(cf. #03.1)
		App	**#07.2**	Associação da tipologia de dados a perfis específicos, individuais e associados à função, com privilégios mínimos, onde cada tipo de perfil é definido em função do Tipo de Dado Pessoal a que acede e Ação que pode efetuar sobre o Dado Pessoal (CRUD), de acordo com o princípio da necessidade de conhecer.	(cf. #03.1)
			#07.3	Processo de registo de tentativas de acesso a dados excluídos dos privilégios associados ao perfil (qualquer perfil, incluindo o dos administradores), com alarmística a partir de um determinado número de tentativas (por exemplo, 3 tentativas), a notificar ao encarregado da proteção de dados da organização.	(cf. #03.2)

		BD	**#07.4**	Associação da tipologia de dados a perfis específicos, individuais e associados à função, com privilégios mínimos, onde cada tipo de perfil é definido em função do Tipo de Dado Pessoal a que acede e Ação que pode efetuar sobre o Dado Pessoal (CRUD), de acordo com o princípio da necessidade de conhecer.	(cf. #03.1)
			#07.5	Processo de registo de tentativas de acesso a dados excluídos dos privilégios associados ao perfil (qualquer perfil, incluindo o dos administradores), com alarmística a partir de um determinado número de tentativas (por exemplo, 3 tentativas), a notificar ao encarregado da proteção de dados da organização.	(cf. #03.2)
#08	**Automatização dos processos de concessão, revisão, análise e revogação de acesso.**	**-**	**#08.1**	Aplicam-se as mesmas disposições que em "Capacidade para garantir que os utilizadores fazem uma utilização correta dos dados" e "Revisão de direitos de acesso de utilizadores em intervalos regulares".	(cf. #05 e #06)
#09	**Procedimentos seguros de início de sessão**	**-**	**#09.1**	Aplicam-se as mesmas disposições referidas em "Capacidade para autenticar e autorizar todos os utilizadores e dispositivos, incluindo o acesso controlado por um procedimento seguro de início de sessão".	(cf. #02)
#10	**Capacidade de monitorização, registo e análise de toda a atividade de acessos de modo a procurar ameaças prováveis.**	**-**	**#10.1**	Deve ser guardado registo de atividade (log) de todas as ações que um utilizador efetue sobre dados pessoais, independentemente do seu perfil e função.	(cf. #03.2)
		-	**#10.2**	Todos os registos de atividade (log) devem ser armazenados apenas em modo de leitura, devendo, com uma periodicidade máxima de 1 mês, ser englobados num único bloco de registos e assinado digitalmente (garantia de integridade).	Uma vez que a solução de segurança "armazenamento seguro" encapsula *security patterns* para o armazenamento seguro de informação (ver secção 4.2.2), e que armazenamento seguro de registos de atividade implica que os mesmos não possam ser adulterados (ver secção 3.2.3 para mais detalhe sobre armazenamento seguro), podemos então considerar que a PMD garante que os registos de atividade são armazenados em modo leitura e com garantias de integridade.
		-	**#10.3**	Deve ser guardado registo de atividade (log) de todos os acessos e tentativas falhadas de acesso, obedecendo aos requisitos anteriores.	(cf. #10.2)
		-	**#10.4**	Garantir que os registos de atividade provenientes dos diversos subsistemas (Sistemas Operativos, aplicações, browsers, Sistema de Gestão de Base de Dados - SGBD, etc.) são inequivocamente associados à sua origem.	(cf. #03.2)
		-	**#10.5**	Os registos de atividade (log) devem conter, no mínimo, o endereço de acesso (IP e Porto), Host, HASH da conta do utilizador que efetuou a ação, ação efetuada (CRUD), Tipo de Dado Pessoal onde a ação foi efetuada, data/hora/minuto/segundo (TimeStamp) da ação, alteração efetuada sobre o dado pessoal.	(cf. #03.2)

#11	**Inspeção automática dos conteúdos para procurar dados sensíveis e acessos remotos ao sistema a partir do exterior do ambiente organizacional.**	**-**	**#11.1**	Tendo em vista garantir que a entidade responsável pelo tratamento de dados deve definir e implementar mecanismos de proteção da informação em função da sua relevância e criticidade, deve ser implementado: - Deteção de ameaças na defesa perimétrica do sistema (por exemplo, regras definidas nas firewall, Intrusion Detection System - IDS, etc.); - Extensão desta proteção desejavelmente a todos os dispositivos (incluindo móveis) com acesso a dados pessoais nos sistemas corporativos; - Mecanismo de cifra ponto a ponto sempre que houver necessidade de aceder remotamente ao FE (e apenas a esta camada), como por exemplo com recurso à tecnologia Virtual Private Network (VPN).	Uma vez que a solução de segurança "Data Loss Prevention" encapsula *security patterns* para a filtragem de informação tanto ao nível de rede como da aplicação, podemos então considerar que a PMD garante a proteção da informação em função da sua relevância e criticidade.
#12	**Proteção dos dados contra modificações não autorizadas, perdas, furtos e divulgação não autorizada.**	**FE**	**#12.1**	FE desenvolvido e em produção de acordo com as melhores práticas de segurança, garantindo a proteção desta camada aos ataques mais comuns (SQLi, injeção de código, etc.).	Sendo a PMD uma metodologia de segurança que considera as possíveis ameaças ao sistema desde as fases iniciais da sua conceção (ver secção 4.2.3 – 1.1.1.) podemos então considerar que a PMD garante a proteção da camada de FE aos ataques mais comuns.
			#12.3	Aplicam-se as disposições anteriores relativas à segurança da atribuição dos acessos e segurança dos dados pessoais, dos acessos propriamente ditos e do registo da atividade efetuada sobre os dados pessoais.	(cf. #03, #04, #07 e #10)
		App	**#12.4**	Camada aplicacional segregada da rede ou ambiente com visibilidade e/ou acesso exterior.	Durante a fase de Deployment (ver secção 4.2.3 – 4.) o sistema é colocado em produção e como tal é configurado para garantir a sua correta operação no que diz respeito às funcionalidades de segurança. Uma vez que a segregação entre camadas acessíveis aos utilizadores (FE) e as restantes é um controlo essencial para garantir a segurança dos dados (ver secção 4.1 para mais detalhe sobre uma arquitetura de alto nível), podemos então considerar que a PMD garante a segregação entre as camadas aplicacional (e de bases de dados) e o ambiente com acesso exterior (FE).
			#12.5	Aplicam-se as disposições anteriores relativas à segurança da atribuição dos acessos e segurança dos dados pessoais, dos acessos propriamente ditos e do registo da atividade efetuada sobre os dados pessoais.	(cf. #03, #04, #07 e #10)
		BD	**#12.6**	Camada de BD segregada da rede ou ambiente com visibilidade/acesso exterior.	(cf. #12.4)
			#12.7	Aplicam-se as disposições anteriores relativas à segurança da atribuição dos acessos e segurança dos dados pessoais, dos acessos propriamente ditos e do registo da atividade efetuada sobre os dados pessoais.	(cf. #03, #04, #07 e #10)
			#12.8	Mascaramento, anonimização ou, sendo necessário, cifra dos dados pessoais transmitidos ou acedidos.	(cf. #01.11 e #01.12)

#13	**Capacidade para garantir a identidade correta do remetente e destinatário da transmissão dos dados pessoais.**	**-**	**#13.1**	Deve ser garantida a integridade das zonas Domain Name System (DNS) onde se encontra inserido o sistema e o ecossistema envolvente, recorrendo às boas práticas de DNSSec e de configuração de sistemas de Correio Eletrónico (por exemplo, Sender Policy Framework - SPF, DomainKeys Identified Mail – DKIM, Domain-based Message Authentication, Reporting and Conformance - DMARC, entre outros.	Durante a fase de Deployment (ver secção 4.2.3 – 4.) o sistema é colocado em produção e como tal é configurado para garantir a sua correta operação no que diz respeito às funcionalidades de segurança. Sendo assim, podemos então concluir que a PMD garante a correta configuração das zonas DNS e sistemas de correio eletrónico.
#14	**Os sistemas de armazenamento devem garantir redundância e disponibilidade, não devendo existir nenhum "single point of failure".**	**-**	**#14.1**	A arquitetura de processamento e armazenamento deve garantir as propriedades da redundância, resiliência e disponibilidade.	Sendo a PMD uma metodologia de segurança que considera as possíveis ameaças ao sistema desde as fases iniciais da sua conceção (ver secção 4.2.3 – 1.1.1.) podemos então considerar que a PMD garante a as propriedades de redundância, resiliência e disponiblilidade.
		-	**#14.2**	Devem existir dois tipos de backups (online e offsite), que devem obedecer aos mesmos requisitos de segurança definidos para os sistemas produtivos.	Uma vez que a solução de segurança "armazenamento seguro" encapsula *security patterns* para o armazenamento seguro de informação (ver secção 4.2.2), e que armazenamento seguro de informação implica que existam back-ups da informação com os mesmos controlos de segurança, podemos então considerar que a PMD garante que a informação armazenada é alvo de backups.
		-	**#14.3**	Os backups offsite devem ser guardados numa localização que não esteja exposta aos mesmos riscos exteriores da localização original, podendo ser da organização mas geograficamente distinta e/ou afastada.	Uma vez que a solução de segurança "armazenamento seguro" encapsula *security patterns* para o armazenamento seguro de informação (ver secção 4.2.2), e que armazenamento seguro de informação, neste caso backups, implica que os mesmos não possam ser comprometidos pelas mesmas condições que o sistema em produção, podemos então considerar que a PMD garante que a informação armazenada em backups não está exposta aos mesmos riscos exteriores da localização original.
#15	**As redes e sistemas de informação devem possuir as funcionalidades necessárias ao respeito pelos direitos do titular dos dados**	**-**	**#15.1**	Os sistemas devem estar capacitados para classificar, priorizar, pesquisar, editar e apagar os dados pessoais.	Sendo a PMD uma metodologia de segurança que considera as possíveis ameaças ao sistema desde as fases iniciais da sua conceção (ver secção 4.2.3 – 1.1.), e que o tratamento de dados pessoais é por si só um risco ao qual está associado um conjunto de funcionalidades que devem existir, podemos então concluir que a PMD garante que os sistemas estão capacitados para classificar, priorizar, pesquisar, editar e apagar os dados pessoais (ver secção 3.2.2).
		-	**#15.2**	Os sistemas devem possuir os controlos necessários que permitam a identificação, autenticação, acesso e validação dos dados pessoais armazenados.	(cf. #15.1)

#16	**As tecnologias de informação a implementar devem permitir a portabilidade e a exportação de dados pessoais.**	**-**	**#16.1**	Deve-se garantir a utilização de formatos digitais compatíveis, que assegurem a interoperabilidade técnica e semântica dentro da Administração Pública, na interação com o cidadão ou com a empresa e para disponibilização de conteúdos e serviços, adotando as especificações técnicas e formatos digitais definidos no Regulamento Nacional de Interoperabilidade Digital, aprovado pela Resolução do Conselho de Ministros n.º 91/2012, ou noutro que o venha a substituir	Sendo a PMD uma metodologia de segurança que considera o tratamento de dados pessoais desde as fases iniciais de conceção de um sistema, e que a interoperabilidade de dados pessoais é um requisito regulamentar para o tratamento dos mesmos (ver secção 2.4.4 para mais detalhe sobre o direito à portabilidade), podemos então considerar que a PMD garante a utilização de formatos digitais compatíveis, sendo que o regulamento referido é uma especificidade dentro da Administração Pública.
#17	**Devem ser definidas políticas que garantam a segurança dos dados pessoais, em alinhamento com a estratégia superiormente definida para a segurança do tratamento de dados pessoais.**	**-**	**#17.1**	As políticas que garantam a segurança do tratamento de dados pessoais devem abranger: - A priorização e classificação dos dados de acordo com os critérios de sensibilidade e criticidade pré-definidos; - A criação; - A modificação; - A transmissão; - A recolha (independentemente do respetivo meio ou processo); - A destruição; - O armazenamento (incluindo a retenção); - A pesquisa de dados.	Sendo a PMD uma metodologiade segurança que considera o tratamento de dados pessoais desde as fases iniciais de conceção de um sistema e que um dos *inputs* para a definição de requisitos de segurança são as políticas de segurança (ver secção 4.2.3 – 1.1.), podemos então considerar que a PMD garante o alinhamento com as políticas de segurança existentes.
		-	**#17.2**	Deve-se garantir o conhecimento, a todo o tempo, dos ativos de informação relativamente a dados pessoais, de modo a permitir identificar inequivocamente o estado da informação em todo o seu ciclo de vida.	Sendo a PMD uma metodologia de segurança que identifica os ativos de informação e casos de uso de um sistema desde as fases iniciais da sua conceção (ver secção 4.2.3 – 1.1.1.) com o objetivo de identificar a informação ou partes de um sistema que devem ser protegidos, podemos então considerar que a PMD garante que é possível identificar inequivocamente o estado da informação em todo o seu ciclo de vida.

Anexo H – Questionário de usabilidade

	Discordo totalmente				Concordo totalmente
1. Imagino-me a utilizar esta metodologia com frequência					
	1	2	3	4	5
2. Achei a metodologia desnecessariamente complexa					
	1	2	3	4	5
3. Achei a metodologia fácil de utilizar					
	1	2	3	4	5
4. Iria precisar do apoio de um técnico para recorrer a esta metodologia					
	1	2	3	4	5
5. Achei que as diversas funções da metodologia estavam bem integradas					
	1	2	3	4	5
6. Achei que existia demasiada inconsistência nesta metodologia					
	1	2	3	4	5
7. A maioria das pessoas iria aprender a utilizar a metodologia rapidamente					
	1	2	3	4	5
8. Achei a metodologia demasiado pesada para ser utilizada					
	1	2	3	4	5
9. Senti-me bastante confiante ao utilizar a metodologia					
	1	2	3	4	5
10. Teria de aprender muitos conceitos antes de poder utilizar a metodologia					
	1	2	3	4	5

Questionário desenvolvido utilizando o método proposto em [88].