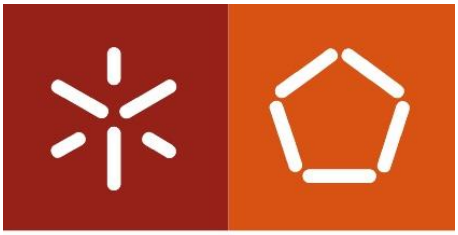

Universidade do Minho

Departamento de Informática

Mestrado em Engenharia Informática

Criptografia e Segurança de Informação

Engenharia de Segurança
Ficha Prática 8

Grupo Nº 3

Ariana Lousada (PG47034) Luís Carneiro (PG46541)
Rui Cardoso (PG42849)

10 de maio de 2022

1 Secure Software Development Lifecycle (S-SDLC)

1.1 Pergunta P1.1

1.1.1 Em que função de negócio, prática de segurança e actividade do SAMM deve ser levada em linha de conta o regulamento europeu RGPD?

O regulamento europeu RGPD deve ser levado em conta na função de negócio *Policy & Compliance*, na prática de segurança *Governance* e na atividade de *Create compliance gates for projects*, ou seja, na definição de datas no *lifecycle* de um projeto onde o mesmo não pode passar até ser auditado e estar em conformidade.

1.1.2 Em que nível de maturidade dessa prática de segurança tem de estar a empresa, para levar em conta o regulamento europeu RGPD nos seus projetos? Justifique.

No nível 3, tal como se pode verificar na seguinte imagem.

Figura 1: Harvard University SSL Server Test Score

1.2 Pergunta P1.2

No seu projeto de desenvolvimento 1 (PD1, no âmbito da avaliação prática 2) utiliza certamente componentes, bibliotecas ou APIs open source.

1.2.1 Quais são as que utiliza, que versão, e que licenciamento é que têm?

Para o primeiro projeto de desenvolvimento, utilizaram-se sobretudo as *libraries* `java.security`(versão 1.1) e `javax.crypto`(versão 1.4) para desenvolver operações para a segurança do programa. A licensa pode ser consultada através da seguinte hiperligação: https://download.oracle.com/otndocs/jcp/java_se-7-mrel-spec/license.html.

1.2.2 Face ao licenciamento que têm, que restrições/permissões impõem sobre a utilização das mesmas no seu código?

De acordo com o licenciamento, todos os componentes destas *libraries*/*packages* podem ser utilizados para desenvolvimento de software em qualquer sistema operativo.

O licenciamento apenas proíbe ações como sublicenciamento. Para além disto, qualquer código produzido com declarações que comecem por "java", "javax", "com.sun"ou "com.oracle" deve ter testado pelo próprio produtor. A empresa Oracle não se responsabiliza por nenhum dano a insfraestruturas que utilizem os seus produtos.

1.2.3 Que boas práticas considera importantes para a utilização de código open source no seu programa?

Apesar da utilização de código *open source* trazer muitas vantagens às empresas, como diminuição de custos e a disponibilidade de várias versões de várias ferramentas. Contudo, também apresenta riscos, como o acesso excessivo, falta de verificção e falta de suporte.

O acesso excessivo é inevitável, visto que o código está disponível ao público. Isto pode levar ao aumento de manipulações ao código impróprias com o intuito de inserir falhas de segurança.

A falta de verificação também por vezes pode acontecer, caso os criadores do código não testem propriamente as suas ferramentas. Isto pode levar a vulnerabilidades nas infraestruturas criadas com este tipo de código.

Para além disto, também existem casos de falta de suporte, o que pode levar a uma indisponibilidade de atualizações ou até de *security patches*. Caso sejam encontradas vulnerabilidades, um atacante pode explorá-las de modo a obter acesso à empresa através do produto.

Tendo em conta os riscos, algumas boas práticas para a utilização de código *open source* seriam:

- Verificar propriamente o código *open source* que vai ser utilizado, verificando licensas e realizando vários testes para verificar a qualidade do próprio código.
- Utilizar um *abstraction layer*(para interação entre o código *open source* e o código do produto em desenvolvimento) para remover dependências.
- Utilizar código de uma "comunidade ativa- quando o código provém de uma comunidade ativa, existem correções e atualizações frequentemente.
- Utilizar automação - como por exemplo um robot que controle as dependências entre os vários componentes e que atualize de modo a garantir a utilização das versões mais recentes do código.
- Para produtos mais complexos, verificar compatibilidades entre componentes *open source*.

2 Referências

- Software Assurance Maturity Model (SAMM) Versão 1.5, Projeto OWASP
- Java™ Platform, Standard Edition 7 API Specification : https://docs.oracle.com/javase/7/docs/api/overview-summary.html
- Oracle Legal Notices : https://docs.oracle.com/javase/7/docs/legal/cpyr.html
- Security considerations when using open source software : https://cyber.gc.ca/en/guidance/security-considerations-when-using-open-source-software-itsap10059
- Best Practices for Using Open Source Code : https://www.linuxfoundation.org/blog/best-practices-using-open-source-code/
- 5 Best Practices for Utilizing Open Source Software : https://www.beningo.com/5-best-practices-for-utilizing-open-source-software/
- 6 Best Practices for Using Open Source Software Safely : https://www.darkreading.com/application-security/6-best-practices-for-using-open-source-software-safely?slide=4
- Java(TM) Platform, Standard Edition Runtime Environment Version 7 : https://www.oracle.com/java/technologies/javase/jre-7-readme.html