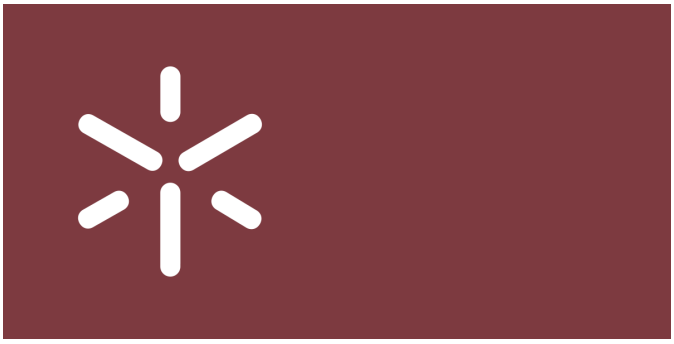

Universidade do Minho

Mestrado em Engenharia Informática

Engenharia de Segurança

Projecto de Análise - Grupo 10

OWASP Top 10 and OWASP Proactive Controls

A74806 - João Amorim
PG52687 - João Rodrigues
PG54708 - Sérgio Ribeiro



06 de Março de 2024

Índice

1 Introducão **1**
1.1 Contextualização . . . 1
1.2 Objetivos . . . 1

2 OWASP Top 10 **3**
2.1 Overview e Importãncia . . . 3
2.2 Top 10 . . . 4
2.2.1 A1:2021 - Broken Access Control . . . 4
2.2.2 A2:2021 - Cryptographic Failures . . . 5
2.2.3 A3:2021 - Injection . . . 6
2.2.4 A4:2021 - Insecure Design . . . 8
2.2.5 A5:2021 - Security Misconfiguration . . . 9
2.2.6 A6:2021 - Vulnerable and Outdated Components . . . 11
2.2.7 A7:2021 - Identification and Authentication Failures . . . 12
2.2.8 A8:2021 - Software and Data Integrity Failures . . . 14
2.2.9 A9:2021 - Security Logging and Monitoring Failures . . . 16
2.2.10 A10:2021 - Server-Side Request Forgery (SSRF) . . . 17
2.3 Casos de Estudo . . . 19
2.3.1 Equifax (2017) - A4 Insecure Design . . . 19
2.3.2 Capital One (2019) - A1/10 Broken Access Control e SSRF . . . 19
2.3.3 Ransomware WannaCry (2017) - A6 Vulnerable and Outdated Components . . . 20

3 OWASP Proactive Controls **21**
3.1 Overview e Importãncia . . . 21
3.2 Proactive Controls . . . 21
3.2.1 C1: Define Security Requirements . . . 21
3.2.2 C2: Leverage Security Frameworks and Libraries . . . 22
3.2.3 C3: Secure Database Access . . . 23
3.2.4 C4: Encode and Escape Data . . . 24
3.2.5 C5: Validate All Inputs . . . 26
3.2.6 C6: Implement Digital Identity . . . 27
3.2.7 C7: Enforce Access Controls . . . 28
3.2.8 C8: Protect Data Everywhere . . . 29
3.2.9 C9: Implement Security Logging and Monitoring . . . 30
3.2.10 C10: Handle All Errors and Exceptions . . . 31
3.3 Casos de Estudo . . . 32
3.3.1 C1 - 2014 Target Data Breach . . . 32

3.3.2 C6 - 2014 Target Data Breach 32
3.3.3 C7: 2018 ABC Bank Cyber Attack 33

4 Observações e Reflexões sobre Segurança de Software 34
4.1 Fusão dos Conceitos 34
4.2 Observações Recorrentes 34
4.3 Lições Aprendidas e Próximos Passos 35

5 Conclusão 36
5.1 Conclusões e Considerações Finais 36
5.2 Trabalho Futuro 36

Bibliografia 37

1 Introducão

1.1 Contextualização

No cenário atual de segurança cibernética, onde ameaças evoluem constantemente, a importância de proteger aplicações web é primordial. O Open Web Application Security Project (OWASP), uma referência na área, fornece dois recursos cruciais para orientar as organizações: o OWASP Top 10 e os OWASP Proactive Controls. O OWASP Top 10 identifica as dez principais vulnerabilidades enfrentadas por aplicações web, destacando as áreas mais críticas que requerem atenção imediata para prevenir ataques e proteger dados sensíveis.

Complementarmente, os OWASP Proactive Controls incentivam a adoção de medidas de segurança desde o início do desenvolvimento de software, focando na prevenção proativa de falhas de segurança. Juntos, estes recursos oferecem uma base sólida para que organizações desenvolvam aplicações mais seguras, promovendo práticas robustas de segurança cibernética e protegendo a confiança dos utilizadores.

1.2 Objetivos

Este relatório tem como objetivo proporcionar um olhar detalhado sobre as práticas essenciais delineadas nos OWASP Top 10 e OWASP Proactive Controls, destacando a sua importância na fortificação da segurança das aplicações web. Destina-se a aumentar a consciência por parte das organizações sobre as vulnerabilidades predominantes que ameaçam a integridade das suas aplicações, sublinhando a importância crítica de adotar uma abordagem proativa à segurança desde a conceção do desenvolvimento de software. Através da apresentação de orientações práticas, o relatório motiva a integração de práticas de segurança robustas em todas as etapas do ciclo de vida do software, com o objetivo de mitigar os riscos identificados e incutir uma cultura organizacional que dê prioridade à segurança como um elemento fundamental na criação de software.

Além de contribuir para a conformidade com regulamentações de proteção de dados, este relatório visa reforçar a confiança dos utilizadores nas aplicações fornecidas pelas organizações e enfatizar a necessidade de uma revisão e atualização contínuas das práticas de segurança, adaptando-as à evolução do panorama de ameaças cibernéticas. Este foco não só auxilia na proteção contra vulnerabilidades conhecidas mas também prepara as organizações para enfrentar novos desafios de segurança de forma eficaz. Em última análise, busca não apenas educar as equipas de desenvolvimento e segurança sobre as práticas recomendadas mas também estabelecer as fundações para o desenvolvimento e manutenção de aplicações web seguras, resilientes e confiáveis, essenciais no cenário de

segurança cibernética em constante mudança.

2 OWASP Top 10

2.1 Overview e Importãncia

O OWASP Top 10 representa um padrão de excelência reconhecido mundialmente para a identificação dos riscos de segurança mais significativos para as aplicações web. Criado pela Open Web Application Security Project (OWASP), uma entidade internacional empenhada em elevar os padrões de segurança de software, este documento é essencial tanto para desenvolvedores quanto para especialistas em segurança e gestores de TI. Através de revisões periódicas, o OWASP Top 10 adapta-se ao cenário de ameaças em constante evolução, fornecendo uma perspetiva atualizada sobre as falhas de segurança mais importantes e sobre a maneiro como estas podem ser abordadas.

A elaboração do OWASP Top 10 baseia-se na análise de dados provenientes de uma série de fontes, incluindo relatórios de incidentes de segurança, vulnerabilidades publicadas e investigação especializada, beneficiando ainda das contribuições da comunidade global de segurança. Este processo garante que a lista reflete as vulnerabilidades mais relevantes e potencialmente danosas, proporcionando um guia confiável para a identificação e mitigação de riscos.

Além de destacar as principais vulnerabilidades, o OWASP Top 10 serve um propósito maior, o de promover uma cultura de desenvolvimento de software que dê prioridade à segurança. Ao educar os criadores de software sobre os riscos mais comuns e as suas implicações, a lista incentiva à implementação de práticas de programação seguras, a realização de revisões de segurança regulares e a uma resposta ágil às ameaças emergentes. Este foco educacional visa não apenas mitigar as vulnerabilidades específicas, mas também fomentar uma abordagem de segurança integrada e bem informada ao longo do ciclo de vida do desenvolvimento de software.

Ao adotar as recomendações do OWASP Top 10, as organizações podem fortalecer significativamente as suas defesas contra ataques cibernéticos, proteger dados valiosos e manter a confiança dos seus utilizadores. À medida que o cenário de ameaças continua a evoluir, o OWASP Top 10 permanece como um recurso vital, orientando o desenvolvimento de aplicações web mais seguras e resilientes num mundo digital cada vez mais complexo e interconectado.

2.2 Top 10

2.2.1 A1:2021 - Broken Access Control

Descrição

Broken Access Control é uma vulnerabilidade que permite aos utilizadores contornarem as restrições de segurança para realizar ações além das suas permissões autorizadas. Isto pode resultar na divulgação não autorizada de informações, na modificação ou destruição de dados, ou na execução de funções empresariais fora dos limites permitidos. As falhas comuns incluem permitir acesso indiscriminado, manipulação de parâmetros de URL, acesso a contas de outros utilizadores, falta de controlo de acesso em APIs, elevação de privilégios não autorizada e manipulação de metadados, como tokens de controlo de acesso. Em resumo, a quebra de controlo de acesso é a exploração de vulnerabilidades que permitem aos utilizadores agir além das suas permissões pretendidas..

Causas

- **Implementação inadequada ou inexistente de controlos de acesso** - Por exemplo, falhas na validação das permissões de acesso ou na aplicação das regras de controlo de acesso.
- **Falta de monitorização e auditoria adequadas** - A ausência de monitorização eficaz das atividades de acesso e de auditorias regulares pode dificultar a deteção atempada de comportamentos suspeitos ou acessos não autorizados.

Consequências

- **Divulgação não autorizada de informações sensíveis** - Quando há falhas no controle de acesso, os usuários podem acessar informações ou funcionalidades que não deveriam estar disponíveis para eles.
- **Manipulação ou alteração não autorizada de dados** - Capacidade dos usuários de modificar ou manipular dados de forma não autorizada. Isso pode incluir a modificação indevida de registros, alteração de configurações ou parâmetros importantes, ou até mesmo a exclusão acidental ou maliciosa de dados críticos.

Exemplos de Ataques

- **URL Manipulation** - Neste tipo de ataque, um utilizador mal-intencionado pode manipular os parâmetros de uma URL para tentar aceder a recursos ou funcionalidades que normalmente não teria permissão.
- **Privilege Escalation** - Neste tipo de ataque, um utilizador com permissões limitadas tenta aumentar o seu nível de acesso de forma não autorizada.

Prevenção e/ou Mitigação

- **Política de Negação por Defeito** - Exceto para recursos públicos, adotar a política de negação por defeito.
- **Reutilização de Mecanismos de Controlo de Acesso** - Implementar mecanismos de controlo de acesso uma vez e reutilize-os em toda a aplicação, incluindo minimizar o uso de Cross-Origin Resource Sharing (CORS).
- **Aplicação de Propriedade de Registos nos Controlos de Acesso** - Os controlos de acesso devem aplicar a propriedade dos registos em vez de aceitar que o utilizador possa criar, ler, atualizar ou apagar qualquer registo..
- **Desativação de Listagem de Diretórios no Servidor Web** - Desative a listagem de diretórios no servidor web e certificar de que os metadados dos ficheiros (por exemplo, .git) e os ficheiros de cópia de segurança não estão presentes nos diretórios web.
- **Limitação da Taxa de Acesso a APIs e Controladores** - Limitar a taxa de acesso às APIs e controladores para minimizar os danos causados por ferramentas de ataque automatizadas.

2.2.2 A2:2021 - Cryptographic Failures

Descrição

Cryptographic Failures, anteriormente conhecido como Sensitive Data Exposure na lista do OWASP Top 10 de 2017, consiste na proteção inadequada de informações confidenciais devido ao uso incorreto ou ausência de criptografia. Este problema abrange desde a falta de criptografia de dados pessoais, financeiros, ou sensíveis durante a transmissão entre cliente e servidor, até ao armazenamento inseguro destes dados em bases de dados ou sistemas de armazenamento.

A questão central é a falha em assegurar que os dados sensíveis sejam inacessíveis e indecifráveis para indivíduos não autorizados.

Causas

- **Criptografia Fraca ou Inexistente** - Uso de algoritmos de criptografia obsoletos, fracos, ou a ausência total de criptografia para proteger dados sensíveis durante o armazenamento ou transmissão dos mesmos.
- **Configuração Inadequada** - Configuração incorreta dos mecanismos de criptografia, como a utilização de chaves padrão, armazenamento inseguro de chaves, ou falta de rotação de chaves.
- **Exposição de Dados Sensíveis** - Falha ao aplicar criptografia a todos os dados sensíveis, deixando informações como detalhes de cartão de crédito, informações pessoais, entre outros.

Consequências

- **Violação de Dados** - A exposição ou acesso não autorizado a dados sensíveis pode levar a violações de dados, resultando em danos financeiros e de reputação para a organização.
- **Perda de Confiança** - Os clientes perdem confiança nas organizações que falham em proteger os seus dados, podendo resultar em perda de negócio.
- **Sanções Legais** - Violações de regulamentos de proteção de dados podem resultar em sanções legais ou multas significativas.

Exemplos de Ataques

- **Intercepção de Comunicações** - Atacantes podem intercetar comunicações não criptografadas para roubar dados sensíveis transmitidos entre o cliente e o servidor.
- **Acesso a Dados Armazenados** - Acesso não autorizado a bases de dados ou arquivos de armazenamento que contenham dados sensíveis não adequadamente protegidos.

Prevenção e/ou Mitigação

- **Utilizar Criptografia Forte** - Implementar algoritmos de criptografia fortes e atualizados para proteger dados sensíveis durante a transmissão e armazenamento.
- **Gestão Adequada de Chaves** - Assegurar uma gestão segura das chaves criptográficas, incluindo a sua proteção, armazenamento seguro e rotação periódica.
- **Minimização de Dados** - Armazenar apenas os dados necessários e por um período limitado de tempo, aplicando criptografia a todos os dados sensíveis.
- **Camada de Segurança de Transporte (TLS)** - Usar TLS para proteger todas as transmissões de dados na internet, assegurando que todos os dados transmitidos entre o cliente e o servidor são encriptados.
- **Auditorias e Testes de Segurança** - Realizar auditorias de segurança regulares e testes de penetração para identificar e corrigir vulnerabilidades criptográficas.

2.2.3 A3:2021 - Injection

Descrição

A Injection, que passou a englobar a categoria Cross-Site Scripting (XSS), categoria esta que pertencia à lista da OWASP Top 10 de 2017, uma das falhas mais perigosas e mais comuns encontradas em aplicações web. Esta permite que um atacante injete código malicioso numa determinada aplicação, aproveitando-se de falhas nas validações ou no tratamento inadequado dos dados de entrada. Essa exploração pode levar a uma série

de ataques, incluindo a execução de comandos arbitrários nos servidores das aplicações e ao acesso não autorizado a dados sensíveis.

Causas

- **Falha na implementação de controle de acesso adequado** - Ocorre quando não são aplicados controlos de acesso adequados, permitindo que utilizadores não autorizados acedam a áreas restritas do sistema.
- **Utilização de bibliotecas ou frameworks desatualizados e vulneráveis** - Ocorre quando a aplicação utiliza bibliotecas ou frameworks desatualizadas e que contêm vulnerabilidades conhecidas, as quais podem ser exploradas por atacantes.
- **Falta de criptografia adequada** - Esta causa ocorre quando os dados sensíveis não são devidamente cifrados durante o armazenamento ou a transmissão, facilitando o acesso não autorizado por parte de invasores.

Consequências

- **Degradação da integridade dos dados** - Corrupção ou adulteração das informações armazenadas pela aplicação, comprometendo a sua integridade e fiabilidade.
- **Interrupção ou indisponibilidade dos serviços** - Interrupção ou indisponibilidade dos serviços oferecidos, causando impacto operacional e financeiro para a organização.
- ??Provalvemente será geral, apresentar no fim?? **Exposição da reputação e credibilidade** - Exposição pública de falhas de segurança e resultando na perda de confiança dos clientes e parceiros comerciais, afetando negativamente a reputação e credibilidade da organização.

Exemplos de Ataques

- **Injeção de SQL** - Exploração de falhas nas validações de entrada para inserir comandos SQL maliciosos em consultas de bases de dados, permitindo que um atacante manipule ou comprometa os dados armazenados.
- **Injeção de Comandos** - Aproveitamento de vulnerabilidades em inputs de utilizador para injetar comandos maliciosos no sistema operativo hospedeiro, podendo resultar em execução de comandos arbitrários.
- **Injeção de Código** - Exploração de vulnerabilidades em inputs de utilizador para inserir código malicioso, como scripts ou código de programação, que é executado no contexto da aplicação, permitindo ao atacante realizar ações não autorizadas.

Prevenção e/ou Mitigação

- **Utilização de Consultas Parametrizadas** - Implementação de consultas SQL parametrizadas nas bases de dados para separar os dados da estrutura da consulta, prevenindo eficazmente ataques de injeções SQL.
- **Validação de Dados de Entrada** - Verificação rigorosa de todos os dados de entrada recebidos pela aplicação, garantindo que apenas dados válidos e seguros sejam processados.
- **Utilização de Listas de Permissões** - Restrição do acesso a recursos sensíveis apenas para utilizadores autorizados, utilizando listas de permissões para garantir que apenas operações legítimas sejam executadas.
- **Utilização de Frameworks Seguros** - Adoção de frameworks e bibliotecas de segurança reconhecidas, que implementam medidas de proteção contra vulnerabilidades de injeção.
- **Implementação de Testes de Segurança** - Realização de testes de segurança regulares, incluindo testes de penetração e análise estática de código, para identificar e corrigir vulnerabilidades.

2.2.4 A4:2021 - Insecure Design

Descrição

Insecure Design, uma nova categoria no OWASP Top 10 de 2021, refere-se a falhas de segurança que decorrem de erros ou omissões no design da aplicação. Ao contrário de vulnerabilidades que podem ser corrigidas com patches ou atualizações de código, o design inseguro é intrínseco à arquitetura da aplicação e requer uma abordagem de segurança proativa desde o início do desenvolvimento. Esta categoria abrange desde a falta de consideração por princípios de segurança durante a fase de design, até a ausência de controlos de segurança adequados que previnam ataques ou minimizem o impacto de falhas de segurança.

Causas

- **Ausência de Requisitos de Segurança** - Falta de definição clara de requisitos de segurança no início do projeto, o que leva a designs que não consideram adequadamente as ameaças e riscos de segurança.
- **Falhas na Modelagem de Ameaças** - Não realização de uma modelagem de ameaças eficaz ou a sua completa omissão durante o processo de design, resultando numa falta de compreensão sobre potenciais vetores de ataque.
- **Designs que não Seguem Princípios de Segurança** - Ignorância ou desconsideração de princípios de design de segurança fundamentais, como o princípio do menor privilégio, segregação de deveres, e defesa em profundidade.

Consequências

- **Vulnerabilidades de Segurança Sistemáticas** - Vulnerabilidades que são difíceis de corrigir sem uma revisão completa ou redesign da aplicação, podendo permanecer como pontos fracos durante todo o ciclo de vida da aplicação.
- **Ataques de Maior Impacto** - Falhas de design permitem ataques que podem comprometer componentes críticos do sistema, resultando em violações de dados mais significativas ou comprometimento total do sistema.
- **Custos de Mitigação Elevados** - Corrigir falhas de design inseguro muitas vezes requer mudanças de arquitetura significativas, o que pode ser muito mais custoso e demorado do que corrigir vulnerabilidades específicas de implementação.

Exemplos de Ataques

- **Exploração de Falhas de Autorização** - Atacantes exploram falhas de design que não implementam adequadamente controlos de acesso, permitindo-lhes aceder a dados ou executar ações não autorizadas.
- **Elevação de Privilégios** - Atacantes tiram vantagem de designs que não separam adequadamente os privilégios de utilizador, permitindo-lhes elevar os seus privilégios dentro do sistema.

Prevenção e/ou Mitigação

- **Adotar uma Abordagem de Segurança desde o Design** - Integrar a segurança no processo de design, estabelecendo requisitos de segurança claros e considerando a segurança como um aspecto central do design da aplicação.
- **Realizar Modelagem de Ameaças** - Utilizar a modelagem de ameaças para identificar e avaliar potenciais vulnerabilidades de segurança desde o início, orientando o design da aplicação para mitigar essas ameaças.
- **Seguir Princípios de Design de Segurança** - Aplicar princípios de design de segurança, como segregação de deveres, defesa em profundidade, e o princípio do menor privilégio, para construir sistemas mais seguros.
- **Revisão e Testes de Segurança Contínuos** - Realizar revisões de segurança regulares e testes de penetração para identificar e corrigir falhas de segurança no design, mesmo após a aplicação estar em produção.

2.2.5 A5:2021 - Security Misconfiguration

Descrição

Security Misconfiguration, uma categoria já existente em 2016, passou a englobar a categoria XML External Entities (XXE), sendo que define situações em que uma aplicação, base de dados, servidor web, ou plataformas com certas relações entre sim não

estão configuradas adequadamente no que diz respeito a factores de segurança. Este tipo de vulnerabilidade resulta do uso de configurações padrão inseguras, configurações incompletas, armazenamento inseguro de passwords e chaves de criptografia, páginas de administração desprotegidas, ou deixar serviços e funcionalidades a funcionar desnecessariamente. Existe ainda um aumento de risco substancial em situações onde existe uma correta configuração de cada componente, mas em que não são consideradas as interações e dependências dos componentes como um conjunto.

Causas

- **Uso de Configurações Padrão** - Sistemas e aplicações são frequentemente instalados com configurações padrão que podem ser inseguras, como passwords padrão ou funcionalidades desnecessárias ativadas.
- **Configuração Incompleta** - Falta de aplicação de todas as configurações de segurança necessárias, muitas vezes devido a uma falta de conhecimento ou negligência.
- **Exposição de Informações Sensíveis** - Informações sensíveis, como mensagens de erro detalhadas, que são expostas aos utilizadores, o que pode fornecer métodos de ataque valiosos para um atacante.
- **Serviços Desnecessários Ativos** - Manutenção de serviços, funcionalidades, ou páginas de administração acessíveis quando não são necessários para a operação da aplicação.

Consequências

- **Acessos Não Autorizados** - Configurações incorretas podem permitir acessos não autorizados a sistemas e dados sensíveis.
- **Violações de Dados** - A exposição de informações sensíveis ou a exploração de serviços indevidamente configurados pode levar a violações de dados.
- **Ataques e Fraudes** - A configuração de segurança inadequada pode ser explorada para a realização de ataques e fraudes, comprometendo tanto a integridade quanto a disponibilidade dos sistemas.

Exemplos de Ataques

- **Exploração de Páginas de Administração Não Protegidas** - Acesso a interfaces de administração que não foram devidamente protegidas ou escondidas.
- **Ataques Baseados em Mensagens de Erro** - Utilização de informações obtidas de mensagens de erro detalhadas para realizar ataques mais sofisticados.

Prevenção e/ou Mitigação

- **Revisão e Auditoria de Configurações** - Realizar revisões periódicas e auditorias das configurações de todos os componentes do sistema para garantir que estão configurados de acordo com as melhores práticas de segurança.
- **Aplicar o Princípio do Privilégio Mínimo** - Garantir que apenas as funcionalidades, serviços, e permissões necessárias estão habilitadas e acessíveis.
- **Desativar Serviços Desnecessários** - Desativar serviços e funcionalidades que não são necessários para a operação da aplicação, reduzindo a superfície de ataque.
- **Formação e Consciencialização** - Assegurar que as equipas de desenvolvimento, operações e segurança estão cientes das melhores práticas de configuração de segurança e compreendem a importância da segurança na configuração.

2.2.6 A6:2021 - Vulnerable and Outdated Components

Descrição

Vulnerable and Outdated Componentsrefere-se ao uso de bibliotecas, frameworks, e outros módulos de software que contêm vulnerabilidades conhecidas ou que estão desatualizadas. Este problema surge quando os desenvolvedores integram componentes de terceiros sem verificar a sua segurança ou sem manter esses componentes atualizados com as últimas versões ou patches de segurança. A dependência de componentes vulneráveis pode expor aplicações a uma variedade de ataques, dado que os atacantes podem explorar essas vulnerabilidades conhecidas para comprometer a aplicação ou o sistema subjacente.

Causas

- **Falta de Conhecimento sobre Componentes** - Desconhecimento sobre as vulnerabilidades presentes nos componentes de terceiros utilizados numa aplicação.
- **Ausência de Gestão de Atualizações e Patches** - Falha em manter os componentes atualizados com as últimas versões ou patches de segurança disponibilizados pelos fornecedores.
- **Dependência de Componentes Descontinuados** - Uso de componentes que já não são suportados ou mantidos pelos seus criadores, deixando vulnerabilidades sem correção.

Consequências

- **Comprometimento da Segurança da Aplicação** - Vulnerabilidades em componentes de terceiros podem ser exploradas para realizar ataques, como injeção de código malicioso ou elevação de privilégios.

- **Violações de Dados** - Atacantes podem explorar componentes vulneráveis para aceder a dados sensíveis armazenados pela aplicação.
- **Perda de Confiança dos Utilizadores** - Incidentes de segurança resultantes da exploração de componentes vulneráveis podem levar à perda de confiança dos utilizadores na aplicação e na organização.

Exemplos de Ataques

Exploração de Vulnerabilidades Conhecidas: Atacantes exploram vulnerabilidades conhecidas em bibliotecas ou frameworks para comprometer aplicações ou dados.

Ataques de Cadeia: Comprometimento de componentes de software na cadeia de fornecimento, permitindo a introdução de código malicioso em aplicações através de componentes de terceiros.

- **Exploração de Vulnerabilidades Conhecidas** - Atacantes exploram vulnerabilidades conhecidas em bibliotecas ou frameworks para comprometer aplicações ou dados.
- **Ataques de Cadeia de Suprimentos** -

Prevenção e/ou Mitigação

- **Inventário de Componentes** - Manter um inventário atualizado de todos os componentes de terceiros utilizados, incluindo as suas versões e dependências.
- **Monitorização e Avaliação de Vulnerabilidades** - Utilizar ferramentas de monitorização de vulnerabilidades para identificar componentes desatualizados ou vulneráveis e avaliar o risco associado.
- **Política de Atualização Regular** - Implementar uma política de gestão de patches e atualizações para garantir que os componentes sejam mantidos atualizados com as últimas versões de segurança.
- **Substituição de Componentes Descontinuados** - Procurar alternativas para componentes que foram descontinuados ou que não recebem mais atualizações de segurança.
- **Revisão de Segurança durante a Seleção de Componentes** - Avaliar a segurança de componentes de terceiros antes de integrá-los na aplicação, considerando o seu histórico de vulnerabilidades e o compromisso do fornecedor com a segurança.

2.2.7 A7:2021 - Identification and Authentication Failures

Descrição

Identification and Authentication Failures, anteriormente conhecido como Broken Authentication na lista do OWASP Top 10 de 2017, diz respeito a vulnerabilidades no processo

de verificação da identidade dos utilizadores que permite aos atacantes assumir a identidade de outros utilizadores ou comprometer mecanismos de autenticação. Estas falhas surgem quando sistemas não implementam adequadamente medidas robustas de autenticação, como autenticação multifator, ou quando não gerem de forma segura sessões e credenciais de utilizador. As falhas podem variar desde a exposição de dados de autenticação sensíveis, como passwords, até à implementação inadequada de sessões de utilizador que permitam sequestro de sessão.

Causas

- **Políticas de Password Fracas** - Utilização de passwords fracas, previsíveis ou padrões de reutilização de passwords que facilitam ataques de força bruta ou adivinhação.
- **Armazenamento Inseguro de Credenciais** - Credenciais de utilizador armazenadas sem proteção adequada, como hashing e salting, que podem ser expostas através de vulnerabilidades ou ataques.
- **Falhas na Implementação de Sessões** - Gestão inadequada de sessões, incluindo a falta de expiração de sessões após o logout ou após um período de inatividade, facilitando o sequestro de sessão.
- **Ausência de Autenticação Multifator** - Falta de implementação de MFA, deixando os sistemas vulneráveis se uma password for exposta.

Consequências

- **Acesso Não Autorizado** - Atacantes podem ganhar acesso a contas de utilizadores, incluindo contas com privilégios elevados, comprometendo a segurança dos dados e funcionalidades do sistema.
- **Execução de Ações Maliciosas** - Uma vez autenticados como um utilizador legítimo, os atacantes podem realizar ações maliciosas, como roubo de dados, alteração de informações ou propagação de malwares.
- **Perda de Confiança e Reputação** - Incidentes resultantes de falhas de autenticação podem levar à perda de confiança dos clientes e danos significativos na reputação de uma organização.

Exemplos de Ataques

- **Ataques de Força Bruta** - Atacantes tentam incessantemente adivinhar passwords de utilizadores através de tentativas repetidas.
- **Phishing** - Enganar os utilizadores para que revelem as suas credenciais de autenticação através de websites falsificados ou mensagens enganadoras.

- **Sequestro de Sessão** - Exploração de falhas na gestão de sessões para assumir o controlo da sessão de um utilizador legítimo.

Prevenção e/ou Mitigação

- **Implementar Políticas de Passwords Fortes** - Exigir a utilização de passwords complexas e únicas, e promover a mudança regular das mesmas.
- **Utilizar Mecanismos Seguros de Armazenamento de Credenciais** - Empregar técnicas de hashing robustas, juntamente com salting e stretching, para proteger as passwords armazenadas.
- **Gestão Segura de Sessões** - Implementar medidas como timeouts de sessão, renovação de tokens de sessão após o login, e a invalidação de sessões no logout.
- **Autenticação Multifator** - Implementar MFA para adicionar uma camada adicional de segurança, reduzindo o risco, mesmo se as credenciais de um utilizador forem comprometidas.
- **Educação e Formação dos Utilizadores:** - Promover a consciencialização sobre segurança entre os utilizadores, ensinando-os a identificar tentativas de phishing e a utilizar práticas seguras de autenticação.

2.2.8 A8:2021 - Software and Data Integrity Failures

Descrição

A categoria Software and Data Integrity Failures diz respeito a situações onde a integridade do software ou dos dados não é verificada ou garantida, permitindo que dados não confiáveis ou maliciosos sejam processados. Estas falhas podem surgir em várias formas, desde a falta de validação da integridade de software ou atualizações até à inclusão não intencional de software malicioso durante o processo de desenvolvimento. Também inclui o comprometimento da integridade dos dados através de manipulação, injeção ou outra forma de adulteração, resultando em ações não autorizadas ou não intencionais dentro do sistema.

Causas

- **Falta de Validação de Software** - Não verificar a origem ou integridade de componentes de terceiros ou de atualizações, permitindo a inclusão de software malicioso ou comprometido.
- **Armazenamento e Transmissão de Dados Inseguros** - Dados sensíveis ou críticos que são armazenados ou transmitidos sem encriptação adequada, permitindo a sua manipulação ou interceção.

- **Controlos Insuficientes de Acesso a Dados** - Permissões de acesso a dados mal configuradas ou demasiado permissivas, que permitem a manipulação de dados por utilizadores não autorizados.

Consequências

- **Execução de Código Malicioso** - A execução de software comprometido pode levar a ataques de malware, perda de dados ou controlo não autorizado sobre sistemas.
- **Corrupção de Dados** - A integridade dos dados pode ser comprometida, resultando em perda de precisão, alterações não autorizadas ou destruição de dados.
- **Perda de Confiança** - Falhas na integridade podem levar a uma perda significativa de confiança dos stakeholders, incluindo clientes e parceiros.

Exemplos de Ataques

- **Ataque Man-in-the-Middle** - Interceção e manipulação de dados durante a transmissão entre duas partes.
- **Injeção de Código** - Inclusão de código malicioso em software através de componentes comprometidos ou manipulação de dados.
- **Supply Chain Attack** - Comprometimento de componentes de software na cadeia, permitindo a introdução de vulnerabilidades ou malware em produtos finais.

Prevenção e/ou Mitigação

- **Verificação de Assinaturas Digitais** - Utilizar assinaturas digitais para validar a origem e a integridade do software e das atualizações.
- **Encriptação de Dados** - Proteger dados em repouso e em trânsito utilizando encriptação forte para garantir a sua confidencialidade e integridade.
- **Políticas de Acesso a Dados** - Implementar controlos de acesso rigorosos para dados sensíveis, assegurando que apenas utilizadores autorizados podem ler ou modificar esses dados.
- **Revisão e Atualização de Dependências** - Manter um inventário das dependências de software e atualizá-las regularmente para garantir que todas as componentes são provenientes de fontes confiáveis e estão livres de vulnerabilidades conhecidas.
- **Validação de Entrada de Dados** - Implementar uma validação rigorosa de todos os dados recebidos, para evitar a injeção ou manipulação de dados maliciosos.

2.2.9 A9:2021 - Security Logging and Monitoring Failures

Descrição

Security Logging and Monitoring Failures refere-se à insuficiência ou inadequação dos processos de registo e monitorização em sistemas e aplicações, o que impede a detecção atempada de atividades suspeitas ou maliciosas. Estas falhas podem resultar da ausência de registos de eventos de segurança relevantes, da falta de análise ou revisão dos registos existentes, ou de sistemas de alerta ineficazes. A incapacidade de identificar e responder rapidamente a incidentes de segurança aumenta significativamente o risco de danos causados por ataques cibernéticos.

Causas

- **Registos Insuficientes** - Falta de registo detalhado para eventos críticos de segurança, o que dificulta a análise e a resposta a incidentes.
- **Monitorização Ineficaz** - Falta de sistemas ou processos eficazes para monitorizar continuamente a atividade do sistema em busca de sinais de comprometimento ou ataques em curso.
- **Configuração Incorreta de Alertas** - Configurações de alertas que não são otimizadas, resultando em demasiados falsos positivos ou a falta de alertas para atividades maliciosas reais.
- **Falta de Análise de Registos** - Ausência de análise proativa dos registos gerados, impedindo a detecção de padrões suspeitos ou anómalos de comportamento.

Consequências

- **Deteção Tardia de Ataques** - A falta de monitorização eficaz pode resultar na deteção tardia de ataques, permitindo que os atacantes permaneçam não detectados por longos períodos.
- **Resposta a Incidentes Comprometida** - A incapacidade de responder rapidamente e eficazmente a incidentes de segurança pode resultar em danos significativos e na propagação de ataques.
- **Perda de Dados e Danos à Reputação** - Incidentes de segurança não detectados ou mal geridos podem levar à perda de dados sensíveis e danos significativos na reputação da organização.

Exemplos de Ataques

- **Movimento Lateral Não Detectado** - Atacantes movendo-se livremente através de uma rede sem serem detectados devido à falta de monitorização adequada.
- **Exfiltração de Dados** - Dados sensíveis sendo exfiltrados sem deteção devido à ausência de alertas configurados para tal atividade.

Prevenção e/ou Mitigação

- **Implementação de Registos Abrangentes** - Garantir que todos os eventos de segurança críticos sejam registados de forma abrangente, incluindo tentativas de acesso falhadas, alterações de configuração e transações suspeitas.
- **Monitorização Contínua** - Utilizar ferramentas de monitorização de segurança para analisar continuamente o tráfego de rede e a atividade do sistema em busca de comportamentos suspeitos.
- **Configuração Apropriada de Alertas** - Calibrar sistemas de alerta para minimizar falsos positivos e garantir a geração de alertas para atividades potencialmente maliciosas.
- **Análise Regular de Registos** - Realizar análises regulares e proativas dos registos para identificar padrões anómalos ou suspeitos que possam indicar uma ameaça de segurança.
- **Resposta Automatizada a Incidentes** - Implementar procedimentos de resposta automatizada para incidentes detectados, a fim de mitigar rapidamente potenciais ameaças.

2.2.10 A10:2021 - Server-Side Request Forgery (SSRF)

Descrição

Server-Side Request Forgery é uma vulnerabilidade de segurança onde um atacante induz o servidor a realizar pedidos para um domínio ou recurso externo indesejado e potencialmente malicioso. Essencialmente, SSRF permite que um atacante force o servidor a fazer pedidos em seu nome, explorando a confiança que outros sistemas têm no servidor vulnerável. Isso pode resultar em ações não autorizadas, como aceder serviços internos da rede que são normalmente inacessíveis do exterior, manipular dados sensíveis ou realizar ataques a outros sistemas.

Causas

- **Controlos Insuficientes sobre URLs Externos** - Falta de validação ou sanitização adequada dos URLs aos quais o servidor faz pedidos, permitindo a inclusão de URLs maliciosos.
- **Permissões Excessivas do Servidor** - O servidor tem permissões excessivas que, quando exploradas através de SSRF, permitem a realização de ações mal-intencionadas dentro da infraestrutura interna.
- **Falta de Separação entre o Servidor e Recursos Internos** - A ausência de uma separação clara e segura entre o servidor vulnerável e recursos internos críticos facilita o acesso não autorizado através de SSRF.

Consequências

- **Acesso Não Autorizado a Serviços Internos** - Atacantes podem aceder e interagir com serviços internos da rede, como bases de dados ou sistemas de gestão, que deveriam estar inacessíveis.
- **Exfiltração de Dados Sensíveis** - Possibilidade de exfiltrar dados sensíveis do servidor ou da rede interna, comprometendo a confidencialidade das informações.
- **Execução de Ações Maliciosas** - Atacantes podem realizar ações maliciosas, como enviar pedidos falsificados a outros sistemas, potencialmente levando a ataques de maior escala.

Exemplos de Ataques

- **Bypass de Firewalls** - Utilização de SSRF para fazer pedidos a serviços internos que estão protegidos por firewalls e, portanto, normalmente inacessíveis do exterior.
- **Exploração de Serviços Vulneráveis** - Ataque a serviços internos vulneráveis através do servidor comprometido, explorando vulnerabilidades como injeção de código.

Prevenção e/ou Mitigação

- **Validação Estrita de URLs** - Implementar uma validação rigorosa dos URLs de entrada para garantir que apenas pedidos a recursos confiáveis e pretendidos sejam permitidos.
- **Lista de Permissões de Recursos** - Utilizar uma lista de permissões (whitelist) para restringir os recursos aos quais o servidor pode fazer pedidos, bloqueando assim o acesso a URLs internos ou maliciosos.
- **Separação de Redes** - Assegurar uma separação eficaz entre o servidor e os recursos internos críticos, limitando a capacidade de um atacante explorar SSRF para aceder a serviços internos.
- **Restrições de Permissão do Servidor** - Limitar as permissões do servidor para reduzir o impacto potencial de uma vulnerabilidade SSRF, assegurando que o servidor não possa realizar ações além das necessárias para a sua função.

2.3 Casos de Estudo

2.3.1 Equifax (2017) - A4 Insecure Design

Contexto e Vulnerabilidade Explorada

O ataque à Equifax, uma das maiores agências de crédito nos EUA, foi devastador devido à exploração de uma vulnerabilidade no Apache Struts, uma framework utilizada para desenvolver aplicações web em Java. A vulnerabilidade específica, CVE-2017-5638, permitia a execução remota de código devido a um tratamento inadequado de erros durante o upload de ficheiros. Este tipo de vulnerabilidade está relacionada com um design inseguro (A4 Insecure Design), onde a falta de validações adequadas e práticas de segurança robustas no desenvolvimento e manutenção de software podem levar a falhas críticas.

Consequências do Ataque

Os atacantes conseguiram aceder aos dados pessoais de aproximadamente 147 milhões de pessoas, incluindo nomes, números de Segurança Social, datas de nascimento, endereços e, em alguns casos, números de carta de condução e informações de cartões de crédito. A magnitude do ataque e a sensibilidade dos dados expostos tiveram um impacto significativo tanto para os indivíduos afetados como para a reputação e estado financeiro da Equifax.

2.3.2 Capital One (2019) - A1/10 Broken Access Control e SSRF

Contexto e Vulnerabilidade Explorada

O ataque ao Capital One, um dos maiores bancos e provedores de cartões de crédito dos EUA, foi resultado da exploração de uma configuração incorreta de permissões num servidor web. A atacante, uma ex-engenheira de software, conseguiu explorar uma vulnerabilidade de Server-Side Request Forgery (SSRF - A10) numa aplicação web da Capital One configurada na AWS (Amazon Web Services). Esta falha permitiu que a atacante enviasse pedidos ao servidor da AWS para obter credenciais IAM (Identity and Access Management), que posteriormente usou para aceder a dados armazenados num serviço S3 da AWS. Este incidente é um exemplo de Broken Access Control (A1), demonstrando como permissões de acesso inadequadas ou mal configuradas podem levar a violações de dados significativas.

Consequências do Ataque

Informações pessoais de mais de 100 milhões de clientes nos EUA e cerca de 6 milhões no Canadá foram expostas, incluindo nomes, endereços, pontuações de crédito, limites de crédito, saldos e histórico de pagamentos. A violação destacou a importância crítica de uma gestão de configuração e permissões de segurança robusta em ambientes cloud.

2.3.3 Ransomware WannaCry (2017) - A6 Vulnerable and Outdated Components

Contexto e Vulnerabilidade Explorada

O WannaCry foi um ataque de ransomware que se aproveitou de uma vulnerabilidade nos sistemas operacionais Windows, conhecida como EternalBlue. Esta vulnerabilidade tinha sido identificada e um patch de segurança tinha sido disponibilizado pela Microsoft dois meses antes do ataque. No entanto, muitos sistemas não tinham sido atualizados e permaneciam vulneráveis. Este ataque explorou a entrada detalhada em Vulnerable and Outdated Components (A6), onde a falha em atualizar software crítico com patches de segurança conhecidos permite que atacantes explorem vulnerabilidades para lançar ataques devastadores.

Consequências do Ataque

WannaCry afetou mais de 200.000 computadores em 150 países, criptografando ficheiros e exigindo um resgate para a sua recuperação. Organizações em várias indústrias, incluindo saúde, finanças e governo, foram significativamente impactadas, com serviços críticos interrompidos e perdas financeiras substanciais. Este ataque sublinhou a importância de manter sistemas e componentes atualizados, bem como a necessidade de práticas de segurança proativas para proteger contra ameaças cibernéticas.

3 OWASP Proactive Controls

3.1 Overview e Importãncia

O OWASP Proactive Controls destaca-se como um guia essencial para programadores e engenheiros de software, focando na implementação proativa de medidas de segurança no início do ciclo de vida do desenvolvimento de aplicações. Criado também pela OWASP, este pretende, tal como o OWASP Top 10, educar e capacitar os profissionais de tecnologia com práticas recomendadas para a construção de aplicações web seguras.

Em contraste com o OWASP Top 10, que identifica as principais vulnerabilidades após a aplicação estar em produção, o OWASP Proactive Controls enfatiza a integração da segurança como um elemento essencial do desenvolvimento de software. Este foco preventivo inclui uma variedade de controlos de segurança cruciais, desde a validação de entradas e saídas até a gestão segura de autenticações e sessões, sublinhando a importância de considerar a segurança em todas as fases do desenvolvimento.

A conceção dos Proactive Controls baseia-se na vasta experiência da comunidade de segurança e nas melhores práticas da indústria, oferecendo aos programadores uma lista prioritária de medidas de segurança que devem ser implementadas para minimizar os riscos. Este guia prático é desenhado para ser acessível e aplicável, fornecendo as ferramentas necessárias para antecipar e neutralizar potenciais ameaças de segurança antes que se materializem. Adotando estas práticas proativas, as organizações podem significativamente reduzir a incidência de vulnerabilidades de segurança nas suas aplicações, facilitando a conformidade com regulamentos de proteção de dados e aumentando a confiança dos utilizadores nas suas soluções tecnológicas.

3.2 Proactive Controls

3.2.1 C1: Define Security Requirements

Descrição

Define Security Requirements enfatiza a importância de estabelecer requisitos de segurança claros e específicos no início do ciclo de vida de desenvolvimento de software. Este controlo proativo envolve a identificação e documentação de todas as necessidades de segurança que uma aplicação deve cumprir, baseando-se em padrões de segurança reconhecidos, regulamentações de conformidade e melhores práticas da indústria. Ao definir estes requisitos antecipadamente, as equipas de desenvolvimento podem assegurar que a segurança é uma prioridade integrada.

Implementação

- **Identificação de Requisitos** - Começa com a análise de riscos para identificar potenciais ameaças e vulnerabilidades específicas ao contexto da aplicação. Esta análise deve considerar a natureza dos dados processados, funcionalidades da aplicação, e interações com outros sistemas.
- **Utilização de Padrões e Frameworks** - Adotar padrões de segurança reconhecidos e frameworks de desenvolvimento que promovem práticas seguras pode facilitar a definição de requisitos de segurança robustos.
- **Colaboração Interdisciplinar** - Envolver stakeholders de diversas áreas, incluindo segurança da informação, conformidade legal, e operações, no processo de definição de requisitos para garantir que todos os aspetos de segurança sejam abrangidos.

Benefícios

- **Redução de Vulnerabilidades** - Ao definir e incorporar requisitos de segurança desde o início, as aplicações são desenvolvidas com uma fundação mais segura, reduzindo a probabilidade de vulnerabilidades críticas.
- **Conformidade Assegurada** - A identificação precoce e a integração de requisitos de conformidade relevantes (como GDPR, HIPAA, entre outros) no processo de desenvolvimento ajudam a assegurar que as aplicações estejam em conformidade com as regulamentações aplicáveis, evitando potenciais penalidades legais e danos à reputação.
- **Eficiência de Custo** - Integrar a segurança no início do ciclo de desenvolvimento é mais eficiente em termos de custos do que tentar corrigir problemas de segurança depois de a aplicação estar em produção, uma vez que as correções tardias são geralmente mais complexas e dispendiosas.
- **Confiança dos Utilizadores** - Aplicações desenvolvidas com considerações de segurança integradas desde o início tendem a ser mais resistentes a ataques, o que pode aumentar a confiança e a satisfação dos utilizadores.

3.2.2 C2: Leverage Security Frameworks and Libraries

Descrição

Leverage Security Frameworks and Libraries enfatiza a importância de utilizar bibliotecas e frameworks que já tenham sido testadas e aprovadas pela comunidade de segurança. Estas ferramentas oferecem funções e módulos pré-construídos para lidar com aspetos comuns da segurança, como autenticação, controlo de acesso, e proteção contra ataques como injeções SQL e cross-site scripting. Ao aproveitar estes recursos, as equipas podem evitar erros comuns de implementação e concentrarem-se no desenvolvimento

das funcionalidades específicas da aplicação, sabendo que a base de segurança é sólida e fiável.

Implementação

- **Escolha de Ferramentas Confiáveis** - Selecionar bibliotecas e frameworks com uma forte reputação de segurança, suportados por uma comunidade ativa e com um histórico de resposta rápida a vulnerabilidades de segurança descobertas.
- **Manutenção e Atualização** - Manter estas ferramentas atualizadas é crucial, uma vez que novas vulnerabilidades são descobertas regularmente. Deve-se estabelecer um processo para monitorizar atualizações de segurança e aplicá-las de forma oportuna.
- **Personalização Cuidadosa** - Embora estas ferramentas ofereçam boas práticas de segurança, podem necessitar de configuração ou personalização para atender aos requisitos específicos da aplicação. É importante garantir que estas personalizações não comprometam a segurança incorporada.

Benefícios

- **Redução de Erros de Segurança** - Ao aproveitar o trabalho de especialistas em segurança incorporado nestas ferramentas, reduz-se a probabilidade de erros de segurança comuns e vulnerabilidades na aplicação.
- **Eficiência no Desenvolvimento** - Utilizar bibliotecas e frameworks de segurança permite que as equipas de desenvolvimento se concentrem nas funcionalidades específicas da aplicação, melhorando a eficiência e reduzindo o tempo de desenvolvimento.
- **Conformidade Melhorada** - Muitas destas ferramentas são desenhadas para ajudar as aplicações a cumprir com padrões de segurança e regulamentações, facilitando a conformidade.
- **Segurança Reforçada** - A implementação de componentes de segurança testados e aprovados aumenta a robustez da segurança da aplicação, oferecendo proteção contra uma ampla gama de ataques conhecidos.

3.2.3 C3: Secure Database Access

Descrição

Secure Database Access sublinha a necessidade de implementar medidas rigorosas de segurança na interação entre as aplicações e as suas bases de dados. Este controlo proativo tem como objetivo prevenir a exposição de dados sensíveis e proteger contra ataques que visam comprometer ou extrair informações das bases de dados, como injeções SQL. A adoção de práticas como a autenticação forte, a encriptação de dados em

trânsito e em repouso, e o princípio do mínimo privilégio para acessos à base de dados são essenciais para mitigar estes riscos.

Implementação

- **Utilização de Consultas Parametrizadas** - Evitar injeções SQL utilizando consultas parametrizadas ou preparadas, que separam claramente o código da consulta dos dados fornecidos pelo utilizador.
- **Autenticação e Autorização** - Implementar mecanismos de autenticação robustos e assegurar que apenas utilizadores ou sistemas autorizados possam aceder à base de dados. Aplicar o princípio do mínimo privilégio, concedendo apenas as permissões necessárias.
- **Encriptação** - Proteger dados sensíveis utilizando encriptação em trânsito (durante a transmissão de dados entre a aplicação e a base de dados) e em repouso (dados armazenados na base de dados).
- **Auditoria e Monitorização** - Configurar logs detalhados para ações críticas e monitorizar a atividade na base de dados para detetar e responder a atividades suspeitas ou maliciosas.

Benefícios

- **Proteção Contra Ataques Comuns** - A implementação de acessos seguros à base de dados protege contra várias ameaças, incluindo ataques de injeções SQL, um dos vetores de ataque mais comuns e perigosos.
- **Confidencialidade e Integridade dos Dados** - As medidas de segurança ajudam a garantir que os dados sensíveis sejam acessados apenas por entidades autorizadas e que a integridade dos dados seja mantida, prevenindo alterações não autorizadas.
- **Conformidade Regulamentar** - Muitas regulamentações de proteção de dados exigem a implementação de controlos de segurança rigorosos para o armazenamento e processamento de informações sensíveis. Práticas seguras de acesso à base de dados são fundamentais para atender a estas exigências.
- **Confiança dos Utilizadores** - Uma gestão eficaz da segurança das bases de dados pode reforçar a confiança dos utilizadores e dos clientes na segurança das suas informações pessoais e confidenciais.

3.2.4 C4: Encode and Escape Data

Descrição

A prática de Encode and Escape Data é crucial para prevenir vulnerabilidades de segurança, como Cross-Site Scripting e injeções SQL, que podem ocorrer quando a aplicação

trata de forma inadequada dados não confiáveis. A codificação transforma dados recebidos num formato seguro para apresentação, enquanto o Data Escape envolve tratar dados de forma que sejam interpretados de maneira apropriada pelo destino, seja uma base de dados, um navegador ou outro sistema. Esta abordagem garante que os dados inseridos por utilizadores ou recebidos de fontes externas não sejam utilizados de forma maliciosa.

Implementação

- **Identificar Contextos de Saída** - Determinar os contextos em que os dados serão utilizados (e.g., HTML, JavaScript, SQL) e aplicar métodos de codificação ou escaping específicos para cada contexto, garantindo que os dados sejam tratados de forma segura.
- **Utilizar Bibliotecas Confiáveis** - Recorrer a bibliotecas e frameworks que ofereçam funções de codificação e escaping testadas e comprovadas, reduzindo a possibilidade de erros de implementação.
- **Codificação na Apresentação de Dados** - Quando dados são apresentados ao utilizador, aplicar codificação para prevenir ataques XSS, garantindo que scripts inseridos não sejam executados no navegador do utilizador.
- **Escaping de Dados para Consultas SQL** - Utilizar consultas parametrizadas, que automaticamente tratam o escaping necessário, para proteger contra injeções SQL.

Benefícios

- **Mitigação de Ataques de Injeção** - A codificação e o escaping eficazes de dados são linhas de defesa essenciais contra ataques de injeção, protegendo a aplicação e a base de dados de execução de código malicioso.
- **Proteção de Informações Sensíveis** - Evita que dados sensíveis sejam expostos ou manipulados de forma indevida, mantendo a integridade e a confidencialidade das informações.
- **Conformidade Regulamentar** - Ajuda a cumprir com regulamentos de proteção de dados, que frequentemente exigem medidas para proteger dados contra acessos e manipulações não autorizadas.
- **Confiança dos Utilizadores** - A proteção eficaz contra ataques comuns reforça a confiança dos utilizadores na segurança das aplicações web, promovendo uma experiência online mais segura.

3.2.5 C5: Validate All Inputs

Descrição

Validate All Inputs é uma prática crítica de segurança que consiste em verificar a correção, integridade e segurança de todos os dados recebidos pela aplicação antes de os processar ou armazenar. Esta validação aplica-se a dados provenientes de diversas fontes, incluindo utilizadores, sistemas externos ou qualquer outra entrada de dados. O objetivo é assegurar que apenas dados apropriados e esperados sejam aceites pela aplicação, protegendo-a contra uma variedade de ataques, como injeções SQL, XSS e outros tipos de explorações que se aproveitam de dados não validados.

Implementação

- **Definir Critérios de Validação** - Estabelecer regras claras sobre o que constitui dados válidos para cada tipo de entrada, considerando o seu tipo, formato, tamanho e intervalo de valores permitidos.
- **Implementar Validação no Lado do Servidor** - Embora a validação no lado do cliente possa melhorar a experiência do utilizador, a validação crucial de segurança deve ocorrer no lado do servidor para evitar que seja contornada.
- **Utilizar Listas de Permissão** - Preferir abordagens baseadas em listas de permissão (whitelisting), onde apenas dados que satisfazem critérios estritos são aceites, em vez de listas de exclusão (blacklisting), que tentam bloquear dados conhecidos como perigosos.
- **Sanitizar Dados** - Além da validação, aplicar sanitização aos dados de entrada pode remover ou substituir caracteres perigosos, reduzindo o risco de injeção e outras explorações.

Benefícios

- **Prevenção de Ataques Comuns** - A validação eficaz de todos os inputs protege contra ataques de injeção e outros métodos de exploração que dependem de dados de entrada maliciosos ou inesperados.
- **Melhoria da Integridade dos Dados** - Assegura que apenas dados precisos e apropriados sejam processados e armazenados pela aplicação, mantendo a qualidade e a integridade dos dados.
- **Redução de Erros de Aplicação** - Ao assegurar que os dados de entrada estejam dentro das expectativas, a validação pode ajudar a reduzir erros de aplicação e melhorar a estabilidade e a fiabilidade.
- **Conformidade com Normas de Segurança** - Contribui para o cumprimento de várias normas e regulamentos de segurança da informação, que frequentemente exigem a validação rigorosa de dados inseridos nos sistemas.

3.2.6 C6: Implement Digital Identity

Descrição

Implement Digital Identity refere-se à criação, gestão e validação seguras de identidades digitais de utilizadores, sistemas ou dispositivos que interagem com aplicações. Este controlo proativo abrange desde a autenticação (verificação da identidade) até à autorização (definição de acessos baseada na identidade verificada), passando pela gestão de sessões. Uma gestão eficaz da identidade digital é fundamental para assegurar que apenas utilizadores autorizados possam aceder a recursos e realizar ações dentro de uma aplicação, protegendo contra acessos não autorizados e outros ataques relacionados com a usurpação de identidade.

Implementação

- **Fortalecimento da Autenticação** - Utilizar mecanismos de autenticação robustos, como autenticação multifator (MFA), para adicionar uma camada adicional de segurança, tornando mais difícil para os atacantes comprometerem contas de utilizador.
- **Gestão de Sessões Segura** - Assegurar que as sessões de utilizador sejam geridas de forma segura, implementando timeouts de sessão, renovação de identificadores de sessão após o login e mecanismos seguros de logout.
- **Políticas de Senhas Seguras** - Definir e impor políticas de senhas fortes, incentivando os utilizadores a criar credenciais difíceis de adivinhar ou comprometer.
- **Proteção de Dados de Autenticação** - Garantir que dados sensíveis utilizados na autenticação, como senhas e tokens de sessão, sejam armazenados e transmitidos de forma segura, utilizando técnicas como hashing e encriptação.

Benefícios

- **Prevenção de Acessos Não Autorizados** - Implementar uma gestão robusta da identidade digital ajuda a prevenir acessos não autorizados, protegendo a aplicação e os dados sensíveis contra explorações.
- **Melhoria da Conformidade Regulamentar** - Muitas regulamentações exigem a implementação de controlos rigorosos de autenticação e gestão de identidades. Uma estratégia de identidade digital bem implementada pode facilitar o cumprimento destas normas.
- **Aumento da Confiança dos Utilizadores** - Ao garantir que apenas utilizadores autorizados possam aceder a recursos e dados, as organizações podem aumentar a confiança dos seus clientes na segurança das suas aplicações.

- **Redução de Riscos de Fraude** - A autenticação forte e a gestão eficaz das identidades digitais reduzem significativamente o risco de fraudes, incluindo a usurpação de identidade e outras atividades maliciosas.

3.2.7 C7: Enforce Access Controls

Descrição

O Enforce Access Controls é crucial para assegurar que apenas utilizadores autorizados tenham acesso a recursos específicos dentro de uma aplicação. Este controlo proativo envolve a definição de políticas claras de acesso e a sua rigorosa implementação, regulando assim as interações entre utilizadores (ou sistemas) e os recursos baseados nos seus papéis, responsabilidades e necessidades de negócio. A implementação efetiva de controlos de acesso minimiza o risco de acesso indevido ou malicioso, protegendo contra potenciais ameaças internas e externas.

Implementação

- **Modelo de Controlo de Acesso** - Adotar um modelo de controlo de acesso adequado, como Controlo de Acesso Baseado em Papéis (Role-Based Access Control - RBAC) ou Controlo de Acesso Baseado em Atributos (Attribute-Based Access Control - ABAC), que melhor se adeque às necessidades da aplicação.
- **Princípio do Mínimo Privilégio** - Garantir que os utilizadores tenham apenas os privilégios estritamente necessários para realizar as suas funções, limitando o acesso a informações e funcionalidades sensíveis.
- **Gestão de Identidades e Acessos (IAM)** - Implementar soluções de IAM para gerir de forma centralizada identidades, papéis e políticas de acesso, facilitando a administração e auditoria dos controlos de acesso.
- **Autenticação e Autorização Seguras** - Assegurar que os mecanismos de autenticação e autorização sejam implementados e configurados corretamente para proteger contra tentativas de acesso não autorizado.

Benefícios

- **Proteção contra Acessos Não Autorizados** - A implementação rigorosa de controlos de acesso ajuda a prevenir o acesso indevido a recursos sensíveis, protegendo a aplicação contra possíveis explorações.
- **Redução do Risco de Danos Internos** - Limitando o acesso com base no princípio do mínimo privilégio, minimiza-se o potencial de danos causados por utilizadores internos, quer seja por erro ou malícia.
- **Aumento da Conformidade Regulamentar** - Muitos regulamentos e normas de segurança exigem controlos de acesso eficazes para proteger dados sensíveis. A sua implementação efetiva pode ajudar a cumprir com estas exigências.

- **Melhoria na Segurança Global da Aplicação** - Os controlos de acesso são uma camada fundamental de defesa que contribui significativamente para a segurança geral da aplicação, protegendo contra uma variedade de ameaças.

3.2.8 C8: Protect Data Everywhere

Descrição

Protect Data Everywhere foca na importância de salvaguardar dados sensíveis, independentemente de onde se encontrem: em trânsito, em repouso ou durante o processamento. Este controlo proativo abrange a aplicação de medidas de segurança robustas, como encriptação, mascaramento de dados e gestão de chaves, para garantir a confidencialidade, integridade e disponibilidade dos dados. A proteção abrangente de dados é crucial para prevenir a exposição de informações confidenciais, minimizando o risco de violações de dados e outros tipos de ataques cibernéticos.

Implementação

- **Encriptação de Dados em Trânsito** - Utilizar protocolos seguros, como TLS (Transport Layer Security), para proteger dados sensíveis enquanto são transmitidos através de redes inseguras.
- **Encriptação de Dados em Repouso** - Aplicar encriptação a dados armazenados, seja em discos, bases de dados ou outros meios de armazenamento, para proteger contra acessos não autorizados.
- **Gestão de Chaves de Encriptação** - Implementar uma gestão de chaves eficaz, assegurando que as chaves de encriptação sejam armazenadas de forma segura e apenas acessíveis a entidades autorizadas.
- **Mascaramento de Dados e Anonimização** - Em certos contextos, como ambientes de desenvolvimento e teste, utilizar técnicas de mascaramento de dados ou anonimização para evitar a exposição de dados reais.
- **Controlos de Acesso a Dados** - Assegurar que medidas de controlo de acesso sejam aplicadas aos dados, garantindo que apenas utilizadores autorizados possam aceder ou modificar os dados sensíveis.

Benefícios

- **Prevenção de Violações de Dados** - Proteger dados sensíveis em todos os estados minimiza significativamente o risco de exposições e violações de dados, protegendo a privacidade dos utilizadores e a reputação da organização.
- **Conformidade com Regulamentações** - A implementação eficaz de medidas de proteção de dados ajuda a cumprir com regulamentos de proteção de dados, como o GDPR e a CCPA, que exigem a proteção rigorosa das informações pessoais.

- **Fortalecimento da Confiança do Cliente** - Proteger de forma eficaz as informações dos clientes reforça a confiança na segurança das práticas de negócio da organização, contribuindo para relações de longo prazo mais fortes.
- **Redução de Riscos Financeiros e Legais** - A proteção adequada de dados pode evitar as consequências financeiras e legais associadas a violações de dados, incluindo multas, litígios e danos à marca.

3.2.9 C9: Implement Security Logging and Monitoring

Descrição

Security Logging and Monitoring é essencial para detetar, investigar e responder a incidentes de segurança de forma atempada. Este controlo proativo envolve a recolha, análise e armazenamento seguro de registos de eventos de segurança que ocorrem dentro de sistemas e aplicações. Uma implementação eficaz permite às organizações identificar padrões suspeitos ou anómalos de comportamento, fornecendo uma base crucial para ações corretivas e preventivas contra ameaças à segurança.

Implementação

- **Definição de Políticas de Registo** - Estabelecer políticas claras que determinem quais eventos devem ser registados, incluindo tentativas de acesso falhadas, alterações de configuração e transações sensíveis.
- **Armazenamento Seguro de Registos** - Garantir que os registos sejam armazenados de forma segura, protegidos contra acessos não autorizados e manipulação, e mantidos por um período adequado para fins de auditoria e conformidade.
- **Análise Contínua de Registos** - Utilizar ferramentas automatizadas de análise de registos para monitorizar e alertar sobre atividades suspeitas em tempo real, facilitando a deteção precoce de potenciais ameaças.
- **Integração com Sistemas de Resposta a Incidentes** - Assegurar que os sistemas de registo e monitorização estejam integrados com procedimentos de resposta a incidentes, permitindo uma reação rápida e eficaz a eventos de segurança identificados.

Benefícios

- **Detecção Precoce de Ameaças** - A capacidade de detetar atividades suspeitas ou maliciosas em tempo real permite às organizações responder rapidamente a ameaças, minimizando potenciais danos.
- **Análise Forense e Investigação** - Os registos de segurança fornecem informações valiosas que podem ser utilizadas para investigar a origem e o impacto de incidentes de segurança, ajudando a prevenir futuras ocorrências.

- **Melhoria da Conformidade** - Muitas regulamentações exigem a implementação de registo e monitorização de segurança. Manter registos detalhados ajuda a demonstrar conformidade com estas normas.
- **Aumento da Consciência de Segurança** - A análise de padrões de registo pode revelar lacunas na postura de segurança de uma organização, fornecendo insights valiosos para melhorias contínuas.

3.2.10 C10: Handle All Errors and Exceptions

Descrição

A gestão eficaz de erros e exceções é crucial para prevenir a exposição de informações sensíveis e para garantir a estabilidade e segurança das aplicações. "Handle All Errors and Exceptions"implica implementar uma abordagem padronizada e segura para capturar, tratar e registar erros e exceções que ocorram durante a execução da aplicação. Este controlo proativo ajuda a evitar que detalhes de erros, tais como mensagens de stack trace, informações de configuração ou dados sensíveis, sejam revelados ao utilizador final, minimizando o risco de exploração por parte de atacantes.

Implementação

- **Padronização de Mensagens de Erro** - Desenvolver mensagens de erro genéricas que não revelem detalhes específicos sobre a estrutura ou falhas de segurança da aplicação, enquanto fornecem informações suficientes para que os utilizadores possam tomar ações apropriadas.
- **Logging de Erros Detalhados** - Configurar o registo detalhado de erros e exceções de forma segura no lado do servidor, incluindo o contexto necessário para a análise e resolução de problemas, sem expor informações sensíveis.
- **Revisão e Monitorização de Registos de Erros** - Implementar procedimentos regulares de revisão dos registos de erros e configurar alertas para padrões anómalos ou eventos críticos, facilitando a deteção precoce de potenciais vulnerabilidades ou ataques.
- **Testes de Manipulação de Erros** - Incluir testes específicos para a manipulação de erros como parte do processo de desenvolvimento e QA (Quality Assurance), para assegurar que todos os erros são tratados de forma adequada.

Benefícios

- **Prevenção de Vazamentos de Informação** - Evita que informações potencialmente sensíveis ou úteis para atacantes sejam expostas através de mensagens de erro, aumentando a segurança da aplicação.

- **Melhoria da Experiência do Utilizador** - Uma gestão eficaz de erros contribui para uma experiência do utilizador mais limpa e profissional, ajudando a manter a confiança na aplicação.
- **Facilitação da Resolução de Problemas** - Registos detalhados e seguros de erros fornecem informações valiosas para a equipa de desenvolvimento, facilitando a rápida identificação e correção de problemas.
- **Reforço da Postura de Segurança** - A deteção e respostas proativas a padrões anómalos nos registos de erros podem ajudar a identificar e mitigar tentativas de exploração antes que causem danos significativos.

3.3 Casos de Estudo

3.3.1 C1 - 2014 Target Data Breach

Contexto e Vulnerabilidade Explorada

Em 2014, a empresa Target foi alvo de um ataque cibernético massivo que resultou no vazamento de informações de cartões de crédito e dados pessoais de milhões de clientes. O ataque explorou uma vulnerabilidade na rede de fornecedores da Target, especificamente um fornecedor de HVAC (aquecimento, ventilação e ar condicionado). Os invasores conseguiram acesso à rede da Target através das credenciais de um fornecedor terceiro e, em seguida, exploraram vulnerabilidades na infraestrutura de TI da empresa para roubar dados.

Consequências do Ataque

O ataque à Target teve consequências significativas. Milhões de números de cartões de crédito e dados pessoais de clientes foram comprometidos, resultando em perdas financeiras para os clientes afetados e danos à reputação da empresa. A Target enfrentou custos significativos relacionados à investigação do incidente, mitigação de danos, ações judiciais e compensação aos clientes afetados. Além disso, houve uma perda de confiança do público e uma queda nas vendas da empresa no curto prazo.

3.3.2 C6 - 2014 Target Data Breach

Contexto e Vulnerabilidade Explorada

Em 2019, a empresa de tecnologia XYZ, especializada em serviços de comércio eletrónico, implementou um sistema de autenticação de utilizadores para a sua plataforma online. A empresa adotou uma abordagem de autenticação de nível 1, conforme descrito nas diretrizes do NIST 800-63b, reservado para aplicações de menor risco que não contenham dados pessoais identificáveis.

No entanto, durante uma análise de segurança realizada por especialistas, foi descoberto que a implementação de autenticação da empresa estava sujeita a uma série de

vulnerabilidades. Uma das principais vulnerabilidades identificadas estava relacionada aos requisitos de senha definidos pela empresa. Embora a empresa tenha estabelecido políticas de senha, como comprimento mínimo e aceitação de caracteres ASCII, a falta de implementação adequada e o armazenamento seguro das senhas representavam um risco significativo para a segurança dos usuários.

Além disso, foi descoberto que o processo de recuperação de senha não incorporava elementos de autenticação multifator, o que deixava os usuários vulneráveis a ataques de engenharia social e comprometimento de contas.

Consequências do Ataque

Como resultado das vulnerabilidades de autenticação descobertas, a empresa XYZ enfrentou várias consequências negativas. Hackers aproveitaram as falhas de segurança para comprometer contas de utilizadores, resultando num vazamento de informações pessoais e financeiras. Isto causou danos à reputação da empresa, perda de confiança dos clientes e exposição a ações legais e regulatórias.

3.3.3 C7: 2018 ABC Bank Cyber Attack

Contexto e Vulnerabilidade Explorada

Em 2018, uma grande instituição financeira, conhecida como ABC Bank, foi alvo de um ataque cibernético que expôs informações confidenciais de milhares de clientes. Durante uma investigação subsequente, descobriu-se que o ataque explorou falhas nas políticas de controlo de acesso implementadas pelo banco.

A instituição financeira havia adotado um sistema de Controlo de Acesso baseado em funções para gerir as permissões de acesso dos utilizadores aos recursos do sistema. No entanto, análises mais aprofundadas revelaram que o sistema de controlo de acesso apresentava várias vulnerabilidades, incluindo a falta de aplicação do Princípio do Menor Privilégio.

Uma investigação mais detalhada revelou que certos utilizadores tinham acesso a recursos e dados sensíveis que não eram necessários para realizar as suas funções, o que aumentava significativamente o risco de acesso não autorizado e fuga de informações confidenciais.

Consequências do Ataque

Como resultado das vulnerabilidades de controlo de acesso exploradas, o ABC Bank enfrentou graves consequências. Milhares de clientes tiveram as suas informações pessoais e financeiras comprometidas, resultando em perdas financeiras significativas e danos à reputação da instituição financeira.

Além disso, o banco enfrentou ações regulatórias e legais, incluindo multas substanciais por violações de privacidade e segurança de dados. A confiança dos clientes no ABC Bank foi abalada, levando a uma redução significativa no número de clientes e prejudicando a imagem da instituição financeira no mercado.

4 Observações e Reflexões sobre Segurança de Software

Neste capítulo, reunimos algumas observações e reflexões obtidas ao explorar os pincípios de OWASP Top Ten e do OWASP Proactive Controls, procurando uma visão abrangente sobre as melhores práticas de segurança de informação no desenvolvimento de software, sendo que ao longo do relatório indentificámos padrões e aspetos comuns que merecem destaque.

Ao longo deste capítulo, iremos discutir não apenas as diretrizes estabelecidas pelos padrões OWASP, mas também as implicações práticas desses princípios. Investigaremos como as organizações podem adaptar e implementar essas melhores práticas em seus próprios contextos, considerando os desafios específicos enfrentados por diferentes setores e ambientes de desenvolvimento de software.

4.1 Fusão dos Conceitos

Ao examinar de perto as diretrizes estabelecidas pelo OWASP Top Ten, destacamos as ameaças emergentes e as vulnerabilidades mais comuns que podem comprometer a segurança de sistemas de software. Por outro lado, ao explorar os princípios do OWASP Proactive Controls, buscamos oferecer uma abordagem proativa para mitigar estes riscos, fornecendo *insights* valiosos sobre como implementar medidas preventivas desde as fases iniciais do ciclo de desenvolvimento de software.

4.2 Observações Recorrentes

Ao longo da análise das práticas de segurança da informação, várias observações destacaram-se consistentemente:

1. **Importância da Formação:** A capacitação das equipas de desenvolvimento emerge como um fator crucial na mitigação e prevenção de vulnerabilidades. Investir em formação contínua sobre boas práticas de segurança é essencial para fortalecer a postura defensiva das organizações.

2. **Vazamento de Dados Sensíveis:** Uma tendência preocupante é a recorrência do vazamento de dados sensíveis em ataques. Isso destaca a necessidade urgente de implementar medidas robustas de proteção de dados, como criptografia e controlo de acesso, para salvaguardar informações confidenciais contra exploração maliciosa.

3. **Abordagem Holística:** A segurança da informação não pode ser tratada como uma reflexão tardia no ciclo de desenvolvimento de software. É fundamental integrar práticas de segurança desde as fases iniciais do processo de desenvolvimento, adotando uma abordagem holística que abrange desde a concepção até à implementação e manutenção do sistema.

4.3 Lições Aprendidas e Próximos Passos

Ao analisar as interseções entre o OWASP Top Ten e o OWASP Proactive Controls, e considerando as tendências e desafios identificados ao longo deste relatório, fica claro que a segurança da informação é uma preocupação contínua e dinâmica. Para avançar, as organizações devem adotar uma postura proativa, investindo em educação, implementando práticas robustas de segurança e mantendo-se atualizadas sobre as últimas ameaças e soluções emergentes.

Este relatório não apenas oferece uma análise detalhada das melhores práticas de segurança da informação, mas também serve como um ponto de partida para aprimoramentos futuros, incentivando uma cultura de segurança resiliente e adaptável em organizações de todos os tamanhos e setores.

5 Conclusão

5.1 Conclusões e Considerações Finais

Após uma análise detalhada do OWASP Top 10, do OWASP Proactive Controls e das considerações sobre segurança de software, é inegável que a proteção contra vulnerabilidades representa uma prioridade crucial para todas as organizações.

É fundamental adotar uma abordagem proativa para identificar e mitigar potenciais vulnerabilidades em sistemas, em contraposição a uma abordagem reativa que simplesmente responde a ameaças após a sua ocorrência.

É importante ressaltar que a segurança de software vai além de aspectos puramente técnicos e abrange elementos humanos e organizacionais, incluindo conscientização, treinamento e implementação de políticas e procedimentos adequados. Somente por meio de uma abordagem holística e multidisciplinar é possível enfrentar de forma eficaz os desafios crescentes e complexos no campo da cibersegurança.

5.2 Trabalho Futuro

O campo da segurança da informação está em constante evolução, e é essencial para os profissionais e organizações permanecerem atualizados e adaptáveis para enfrentar os desafios emergentes. Algumas áreas de foco para futuras pesquisas e práticas incluem:

- **Integração de Inteligência Artificial e *Machine Learning*:**Explorar como técnicas de inteligência artificial e *machine learning* podem ser aplicadas para fortalecer as defesas cibernéticas, identificar padrões de ataque e fornecer respostas automáticas a ameaças em tempo real.
- **Desenvolvimento de Ferramentas e Estruturas de Segurança:**Investigar e desenvolver ferramentas e estruturas de segurança que possam facilitar a implementação eficaz das práticas do OWASP Top Ten e OWASP Proactive Controls, simplificando o processo para desenvolvedores e equipas de segurança.
- **Educação e Sensibilização Contínuas:**Promover programas contínuos de educação e sensibilização em segurança da informação para desenvolvedores, profissionais de TI e utilizadores finais, garantindo que todos os envolvidos estejam cientes das melhores práticas de segurança e das últimas ameaças cibernéticas.

Estas áreas representam apenas algumas das muitas oportunidades emocionantes para avançar na segurança da informação e garantir a proteção contínua dos sistemas num mundo digital em constante mudança.

Bibliografia

- https://owasp.org/www-project-top-ten/
- https://owasp.org/www-project-proactive-controls/