

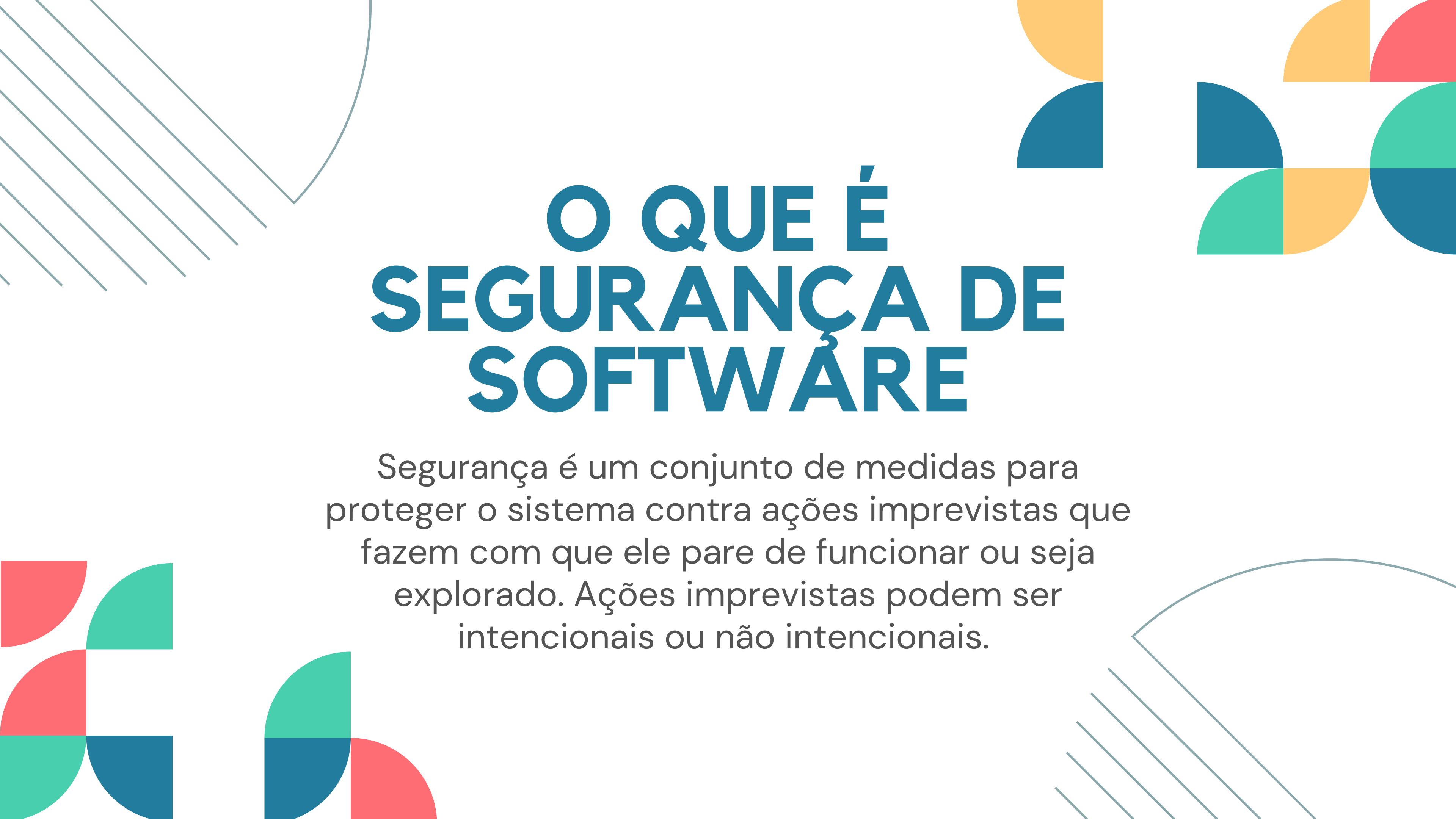
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO RN
Campus Natal-Central
Curso: Tecnologia em Análise e Desenvolvimento de Sistemas
Disciplina: Teste de Software
Professor: Plácido A. Souza Neto

TESTES DE SEGURANÇA

Ana Célia, Felipe Xavier e Pedro Maure

Natal/RN, 13/08/2024

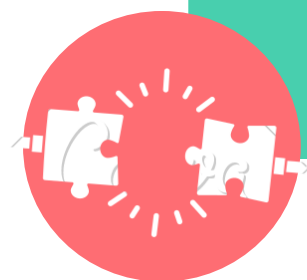




O QUE É SEGURANÇA DE SOFTWARE

Segurança é um conjunto de medidas para proteger o sistema contra ações imprevistas que fazem com que ele pare de funcionar ou seja explorado. Ações imprevistas podem ser intencionais ou não intencionais.

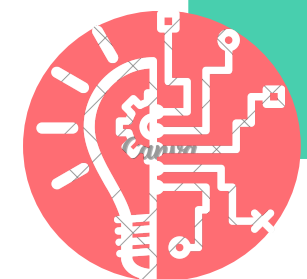
POR QUE É NECESSÁRIO INVESTIR EM SEGURANÇA?



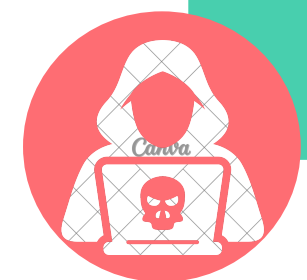
Sistemas cada vez mais conectados e interdependentes



Dados de usuários circulando e armazenados em sistemas



Sistemas críticos, aplicações financeiras, investimentos, bancos



Aumento de casos de invasões

O QUE GARANTIR COM TESTES DE SEGURANÇA?

CONFIDENCIALIDADE

INTEGRIDADE

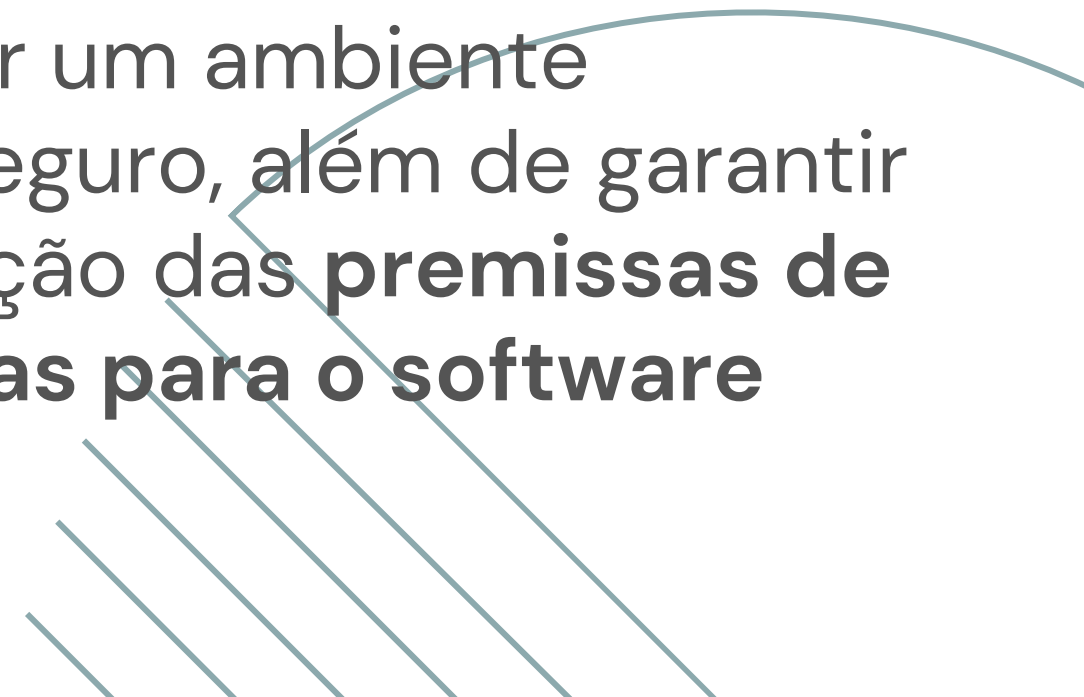
DISPONIBILIDADE

AUTENTICIDADE

NÃO – REPÚDIO

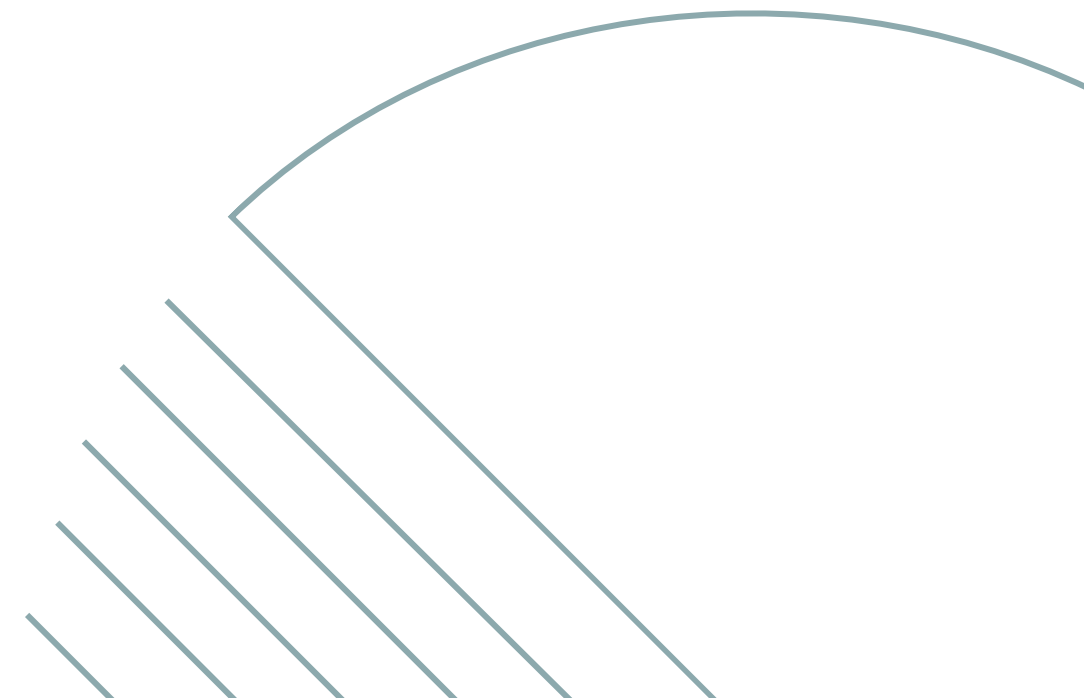
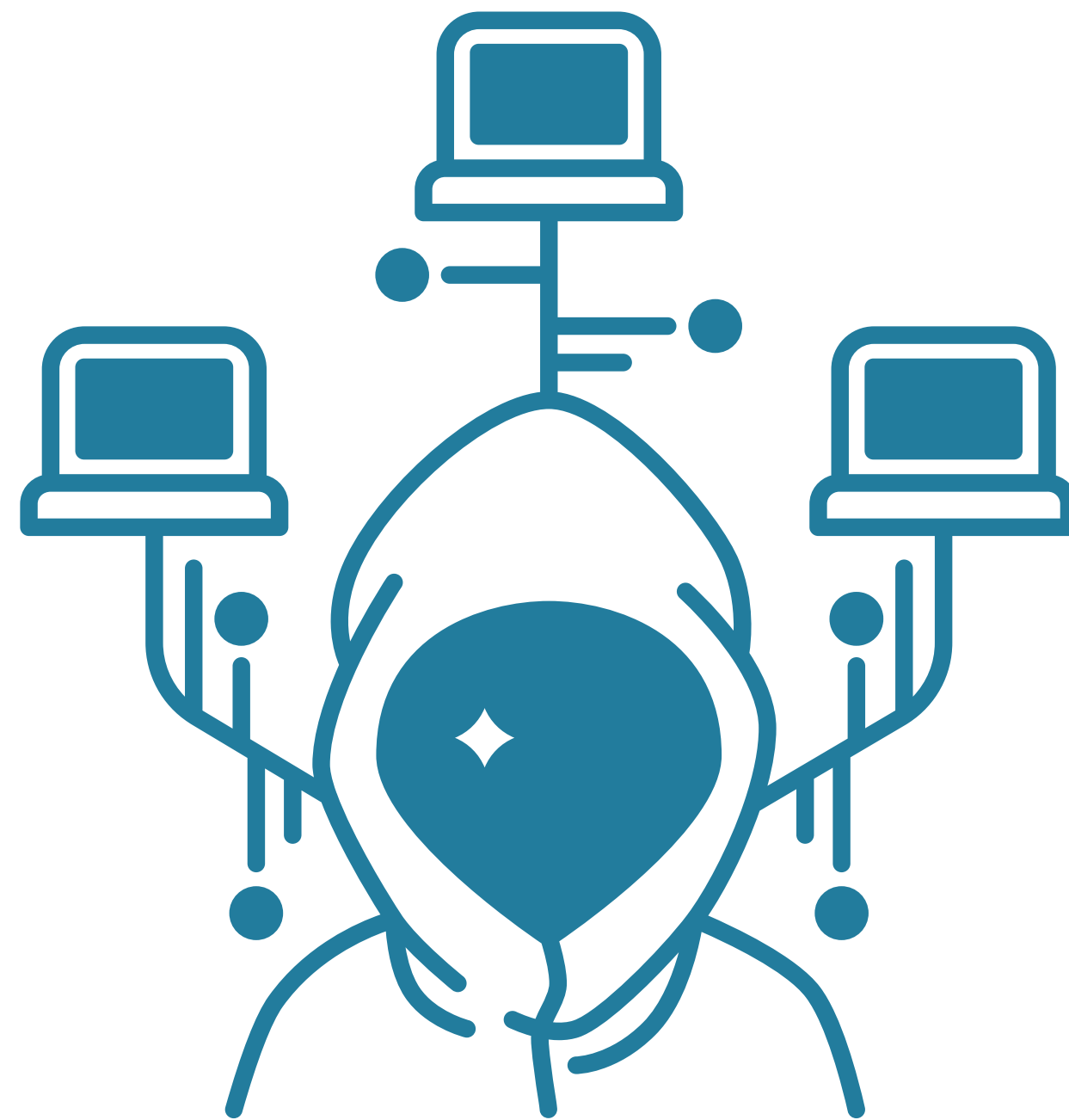
PONTOS IMPORTANTES PARA TESTES DE SEGURANÇA

- Realizar uma busca de **vulnerabilidades** no software
- Assegurar se todos os mecanismos de proteção embutidos na aplicação de fato a protegerão de acessos indevidos.
- Verificar se o software se comporta adequadamente mediante as mais diversas **tentativas ilegais/ indesejadas de acesso**, visando possíveis vulnerabilidades.
- Possibilitar alcançar um ambiente operacional mais seguro, além de garantir uma correta aplicação das **premissas de segurança definidas para o software**



SOBRE O ETHICAL HACKING

Técnicas de acesso a pontos vulneráveis do sistema





ABORDAGENS PARA TESTES DE SEGURANÇA

FUZZING

ferramentas automáticas para fornecer entradas inválidas ou inesperadas para um programa ou função em um programa

ESCALÁVEL

Qualquer linguagem

INJEÇÃO DE FALHAS

introduzir propositalmente falhas ou condições adversas em um sistema para avaliar sua capacidade de gerenciar e se recuperar dessas situações

VERSÁTIL

Host ou Rede





OWASP

Open Web Application
Security Project

A Fundação OWASP® trabalha para melhorar a segurança do software através dos seus projectos de software de fonte aberta liderados pela comunidade, centenas de capítulos em todo o mundo, dezenas de milhares de membros e através da organização de conferências locais e globais.

Fonte: OWASP, 2024 (<https://owasp.org/about/>)

TOP 10 RISCOS DE SEGURANÇA DE APLICAÇÕES WEB



2017

2021

A01:2017-Injection

A02:2017-Broken Authentication

A03:2017-Sensitive Data Exposure

A04:2017-XML External Entities (XXE)

A05:2017-Broken Access Control

A06:2017-Security Misconfiguration

A07:2017-Cross-Site Scripting (XSS)

A08:2017-Insecure Deserialization

A09:2017-Using Components with Known Vulnerabilities

A10:2017-Insufficient Logging & Monitoring

A01:2021-Broken Access Control

A02:2021-Cryptographic Failures

A03:2021-Injection

(New) A04:2021-Insecure Design

A05:2021-Security Misconfiguration

A06:2021-Vulnerable and Outdated Components

A07:2021-Identification and Authentication Failures

(New) A08:2021-Software and Data Integrity Failures

A09:2021-Security Logging and Monitoring Failures*

(New) A10:2021-Server-Side Request Forgery (SSRF)*

* From the Survey

TOP 10 RISCOS DE SEGURANÇA DE APLICAÇÕES WEB



A01: Controle de Acesso Quebrado

Exemplo: Um usuário comum pode acessar a página de administração de um site. Por exemplo, se um site tem uma URL para /admin e um usuário não autenticado pode acessar essa URL sem qualquer verificação de privilégios, isso é uma falha de controle de acesso.

A02: Falhas Criptográficas

Exemplo: Armazenar senhas em texto claro em um banco de dados. Por exemplo, se as senhas dos usuários são armazenadas sem qualquer tipo de criptografia ou hashing (como SHA-1 ou bcrypt), elas estão expostas a qualquer atacante que consiga acessar o banco de dados.

TOP 10 RISCOS DE SEGURANÇA DE APLICAÇÕES WEB



A03: Injeção

Exemplo: Injeção SQL. Por exemplo, se um formulário de login permite que um usuário insira uma string como ' OR '1'='1, e isso é usado diretamente em uma consulta SQL sem sanitização, isso pode permitir ao atacante acessar informações de outros usuários ou modificar dados.

A04: Design Inseguro

Exemplo: Falta de autenticação adequada para operações críticas. Por exemplo, uma aplicação permite que qualquer usuário execute ações administrativas sem realizar uma verificação de autorização adequada, como atualizar ou excluir dados sensíveis.

TOP 10 RISCOS DE SEGURANÇA DE APLICAÇÕES WEB



A05: Má Configuração de Segurança

Exemplo: Servidores web expostos com configurações padrão. Por exemplo, um servidor Apache ou Nginx que está configurado com as configurações padrão e não desativa diretivas como ServerTokens ou ServerSignature, que podem expor informações sobre a versão do servidor e suas configurações.

A06: Componentes Vulneráveis e Desatualizados

Exemplo: Uso de uma biblioteca JavaScript desatualizada com vulnerabilidades conhecidas. Por exemplo, uma aplicação web pode usar uma versão antiga do jQuery que tem uma vulnerabilidade crítica conhecida, e a falta de atualização permite que um atacante explore essa falha.

TOP 10 RISCOS DE SEGURANÇA DE APLICAÇÕES WEB



A07: Falhas na Identificação e Autenticação

Exemplo: Implementação fraca de autenticação multifatorial. Por exemplo, uma aplicação que oferece autenticação multifatorial, mas permite que um usuário bypass o segundo fator ao inserir um código incorreto múltiplas vezes sem bloqueio ou limitação.

A08: Falhas na Integridade do Software e dos Dados

Exemplo: Falta de verificação de integridade dos arquivos. Por exemplo, uma aplicação web que não valida a integridade dos arquivos de configuração ou de upload do usuário, permitindo que um atacante substitua arquivos críticos por versões maliciosas.

TOP 10 RISCOS DE SEGURANÇA DE APLICAÇÕES WEB



A09: Falhas na Registro e Monitoramento de Segurança

Exemplo: Falta de logs de eventos críticos. Por exemplo, uma aplicação que não registra eventos de login e falhas, o que impede a detecção de tentativas de acesso não autorizado e a investigação de incidentes de segurança.

A10: Falsificação de Solicitações do Lado do Servidor (SSRF)

Exemplo: Permitir que um usuário forneça uma URL para ser acessada pelo servidor. Por exemplo, uma aplicação que permite que um usuário insira um URL para a aplicação buscar dados de um serviço externo, mas sem restrições adequadas, permitindo que o atacante faça o servidor acessar serviços internos ou privados.

COMMAND INJECTION

A injeção de comandos é um ataque em que o objetivo é a execução de comandos arbitrários no sistema operativo do anfitrião através de uma aplicação vulnerável. Os ataques de injeção de comandos são possíveis quando uma aplicação passa dados não seguros fornecidos pelo utilizador (formulários, cookies, cabeçalhos HTTP, etc.) para uma shell do sistema. Neste ataque, os comandos do sistema operativo fornecidos pelo atacante são normalmente executados com os privilégios da aplicação vulnerável. Os ataques de injeção de comandos são possíveis, em grande parte, devido à insuficiente validação das entradas.



COMMAND
+
SQL INJECTION
=
?



The image features a light gray background with the word "OBRIGADA" centered in a bold, blue, sans-serif font. The corners are decorated with abstract geometric patterns. The top-left corner has a series of parallel diagonal lines in a light blue-gray color. The top-right corner features a cluster of overlapping semi-circles in yellow, red, and teal. The bottom-left corner shows a similar cluster of overlapping semi-circles in red, teal, and blue. The bottom-right corner contains a large, light blue-gray arc with several parallel diagonal lines extending from its base.

OBRIGADA