INTRODUÇÃO A SEGURANÇA DA APLICAÇÃO

JOAS ANTONIO

Whoami

- Joas Antonio dos Santos 19y;
- Asperger and TDAH;
- Red Team Love;
- CEH Master, OSWP, eMAPT, eJPT;
- Mitre Contributor;
- Hacking is NOT Crime Advocate;
- OWASP Member;
- Instructor and Mentor;
- Criador do grupo Red Team Brasil;

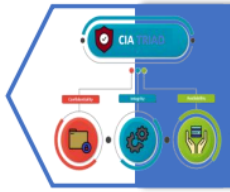
CONCEITOS

O que é Segurança da Informação?

A **proteção das informações** e seus **elementos críticos**, incluindo sistemas e hardware que usam, armazenam e transmitem essas informações.

Ferramentas necessárias: política, conscientização, treinamento, educação, tecnologia.

ACADI-TI



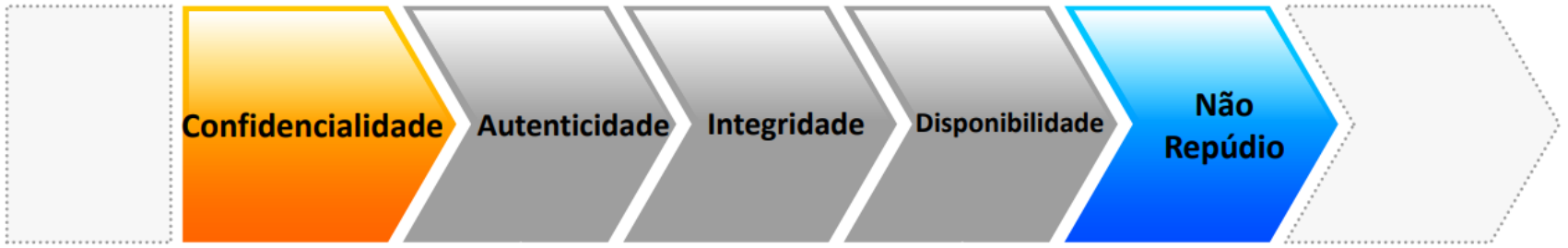
O triângulo **C.I.A** era padrão com base na **confidencialidade**, **integridade** e **disponibilidade**.

Elementos de Segurança da Inforrmação

Confidencialidade é "garantir que informações estejam acessíveis somente àqueles autorizados a terem acesso"

Integridade é "garantir que as informações sejam precisas, completas, confiáveis e em sua forma original"

Não repúdio é "garantir que uma parte de um contrato ou uma comunicação não possa negar a autenticidade de sua assinatura em um documento"



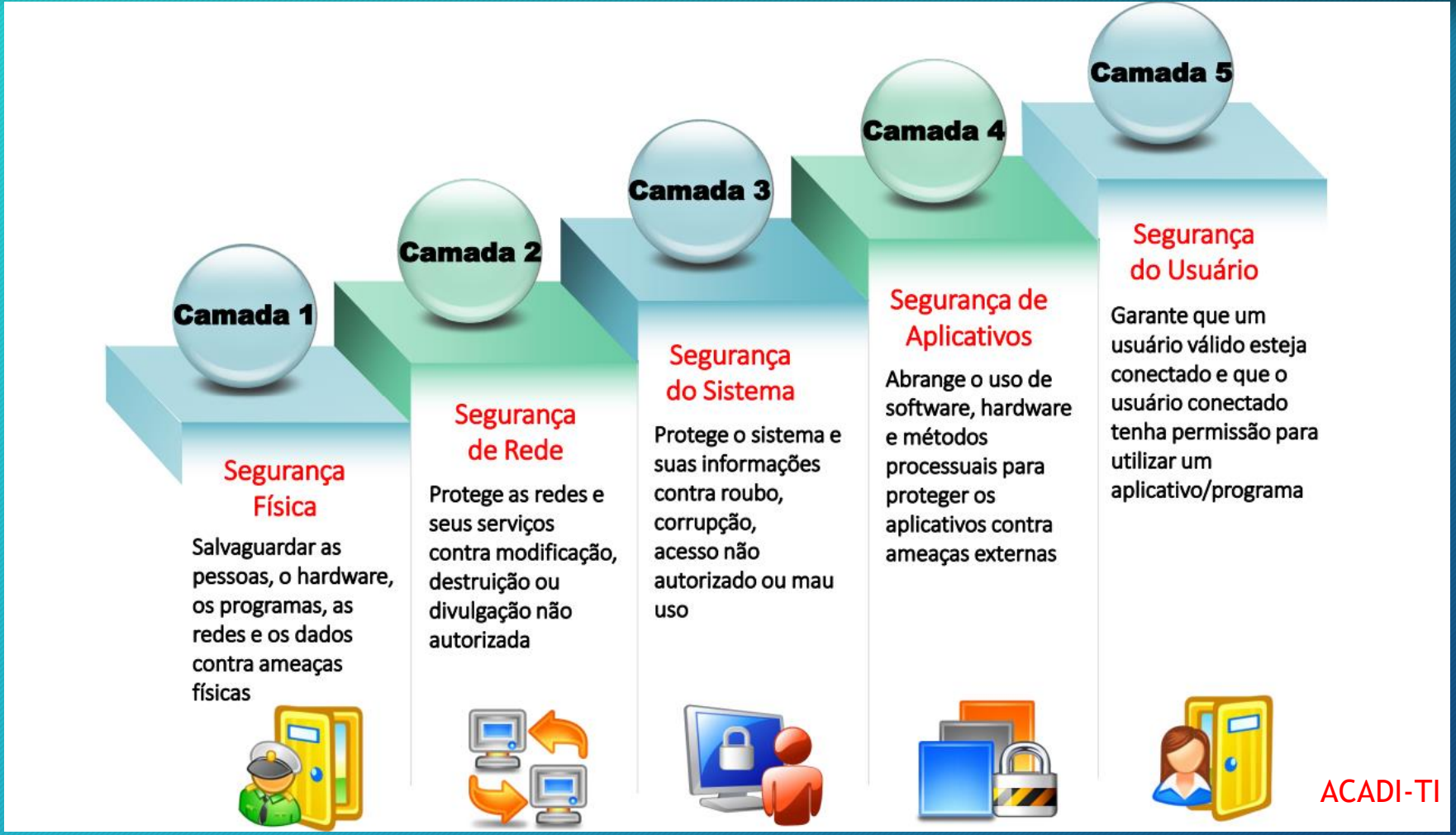
Autenticidade é "a identificação e a garantia da origem da informação"

Disponibilidade é "garantir que a informação esteja acessível a pessoas autorizadas quando necessário sem demora"

ACADI-TI

TRIANGULO DA SEGURANÇA DA INFORMAÇÃO

CAMADAS DA SEGURANÇA



CONCEITOS DE APPSEC

- Uma aplicação é segura quando garante **confidencialidade**, **integridade** e **disponibilidade** de seus recursos restritos
- Recurso restrito é qualquer **objeto**, **dado**, **recurso** ou **função** de uma aplicação projetada para ser acessado apenas por usuários **autorizados**

SEGURANÇA DE APLICAÇÃO

UNICIV
Pós-Graduação EAD em TI

NECESSIDADE DE SEGURANÇA NA APLICAÇÃO

As organizações estão cada vez mais usando **Aplicações Web** para fornecer funções de negócios de alto valor a seus clientes, como vendas em tempo real, transações, gerenciamento de inventário em vários fornecedores, incluindo comércio eletrônico B-B e B-C, fluxo de trabalho e gerenciamento da cadeia de suprimentos, etc.

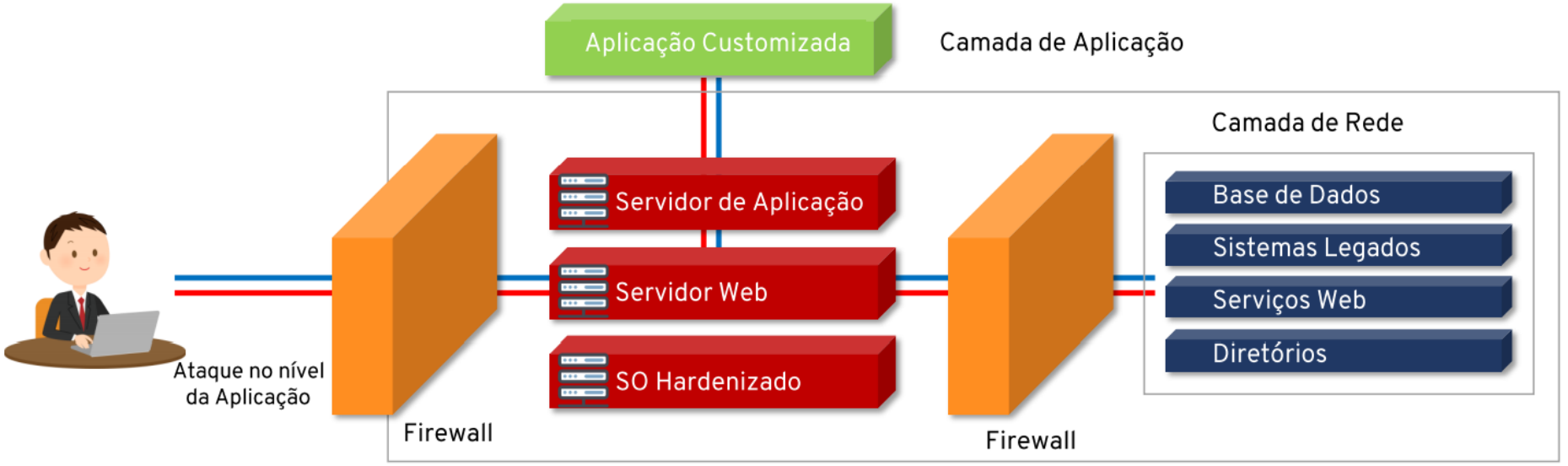
Os invasores exploram vulnerabilidades nas aplicações para **lançar vários ataques** e **obter acesso não autorizado a recursos**.

Um ataque bem sucedido no nível da aplicação pode resultar em:

1. Perda Financeira
2. Afeta a Continuidade dos Negócios
3. Encerramento dos Negócios
4. Divulgação de Informações Comerciais
5. Danifica a Reputação
6. Transações Fraudulentas

NECESSIDADE DE SEGURANÇA NA APLICAÇÃO

- É muito comum ouvir que controles de segurança de perímetro, como sistemas de firewall e IDS, podem proteger sua aplicação, mas não é verdade, pois esses controles **não são eficazes** para defender ataques à camada de aplicação
- Isso ocorre porque as portas 80 e 443 geralmente são abertas em dispositivos de perímetro para tráfego legítimo da Web, que os invasores podem usar para **explorar as vulnerabilidades no nível da aplicação** e entrar na rede



SQL INJECTION

- Os ataques de SQL Injection usam uma **série de consultas SQL maliciosas** para manipular diretamente o banco de dados
- Um invasor pode usar um aplicativo da Web vulnerável para **ignorar medidas normais de segurança** e obter acesso direto a dados valiosos
- Os ataques de SQL Injection geralmente podem ser **executados na barra de endereços**, nos campos da aplicação e através de consultas e pesquisas
- Esse ataque é possível apenas quando a aplicação executa **instruções SQL dinâmicas** e armazena procedimentos com **argumentos** baseados na inserção do usuário

ACADI-TI

SQL INJECTION

WHAT IS SQL INJECTION?

A SQL query is one way an application talks to the database.

SQL injection occurs when an application fails to sanitize untrusted data (such as data in web form fields) in a database query.

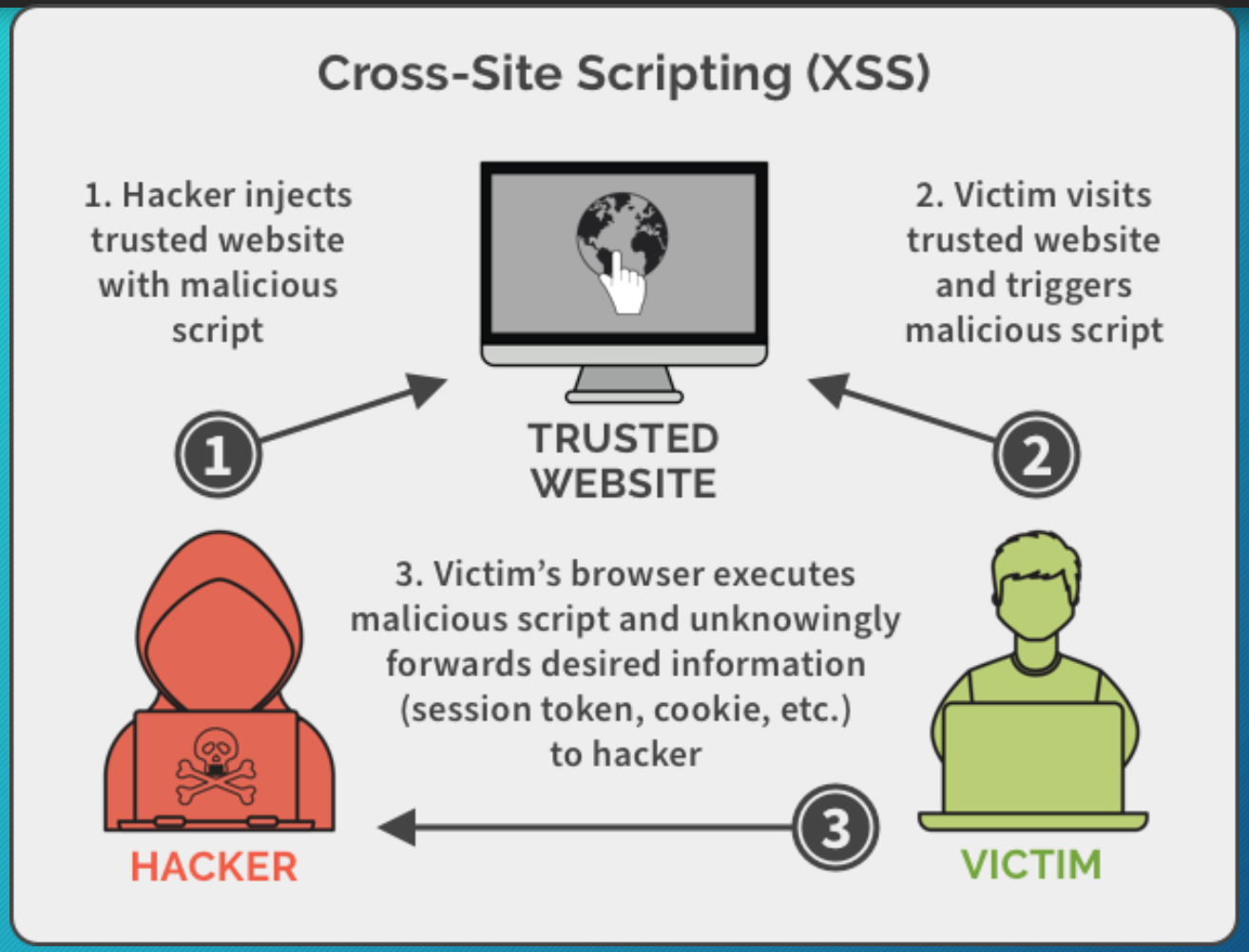
An attacker can use specially-crafted SQL commands to trick the application into asking the database to execute unexpected commands.

XSS – Cross Site Scripting

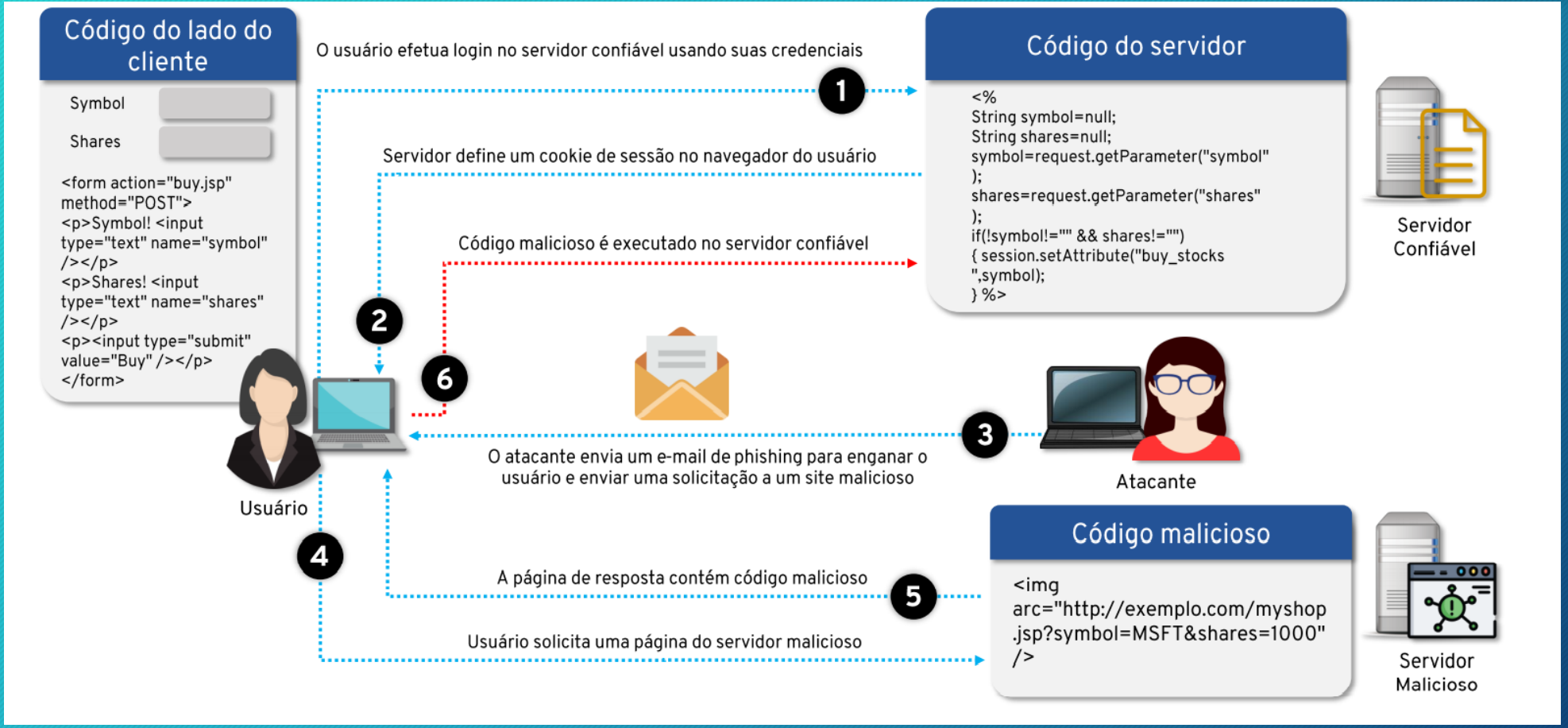
- Os ataques de Cross-site Scripting ('XSS' ou 'CSS') **exploram vulnerabilidades em páginas da web geradas dinamicamente**, o que permite que atacantes mal-intencionados injetem script do lado do cliente em páginas da web visualizadas por outros usuários
- Ocorre quando **dados de entrada inválidos são incluídos** no conteúdo dinâmico enviado para um navegador da web do usuário para renderização
- Os invasores injetam JavaScript, VBScript, ActiveX, HTML ou Flash malicioso para execução no sistema da vítima, **ocultando-o em solicitações legítimas**

- Execução de script malicioso
- Redirecionando para um servidor malicioso
- Explorando privilégios de usuário
- Anúncios em IFRAMES e pop-ups ocultos
- Manipulação de dados
- Session hijacking
- Quebra de senha por brute force
- Roubo de dados
- Análise da intranet
- Keylogging e monitoramento remoto

XSS – Cross Site Scripting



CSRF – Cross Site Request Forgery



RAZÕES PARA FALHAS EM SEGURANÇA DE APLICAÇÃO E CODIFICAÇÃO SEGURA

Nenhuma **orientação adequada** foi fornecida às partes interessadas relevantes nas diferentes fases do desenvolvimento do projeto

Falha ao **reunir** os requisitos de segurança da aplicação na **fase inicial**

Aplicação inadequada dos **princípios de segurança** na fase de design

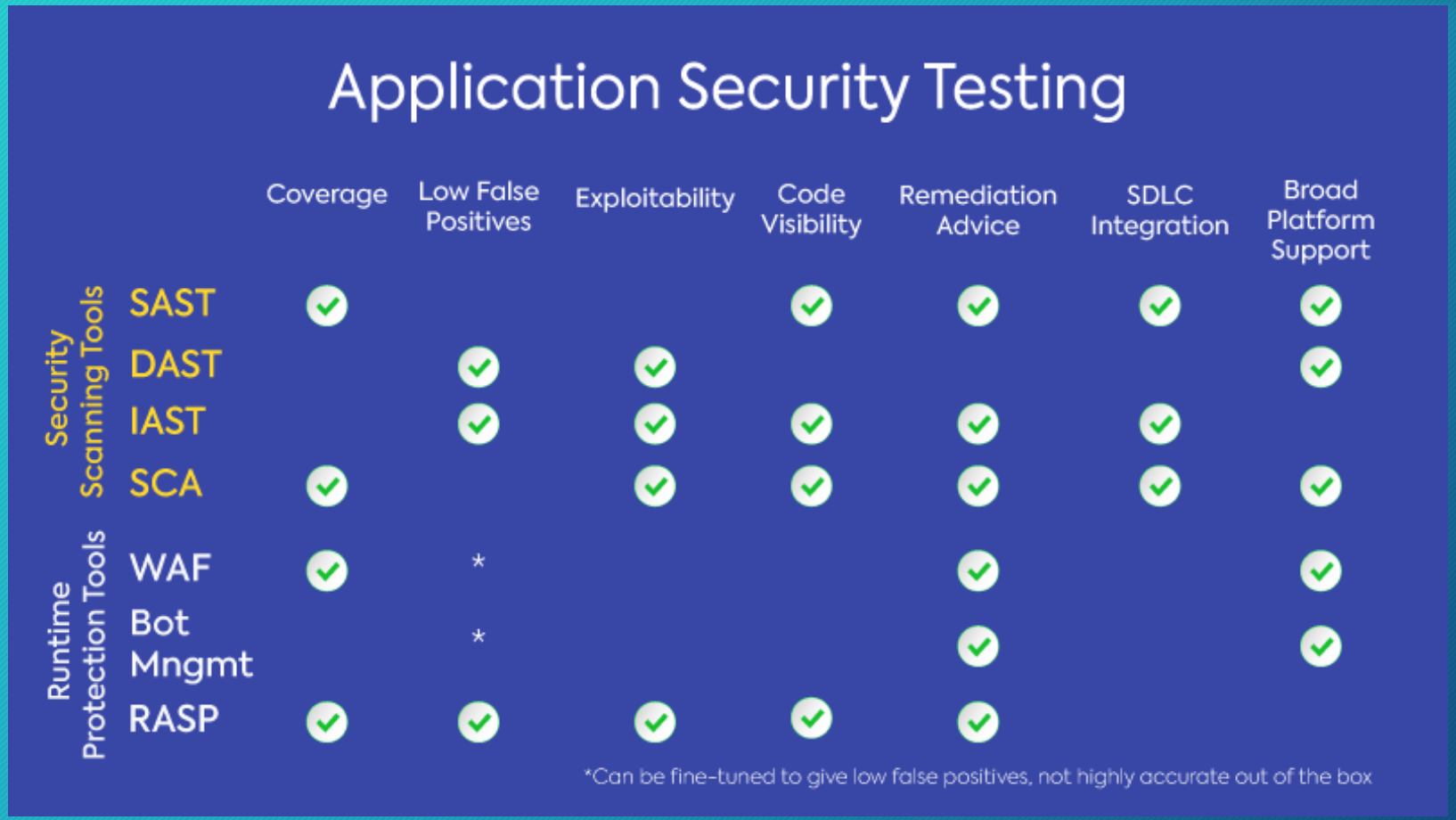
Técnicas de codificação inseguras dão espaço a várias vulnerabilidades

Falta de **teste** de segurança na **fase de teste**

Negligência de segurança na **fase de implantação**

- Validação Inadequada de Entrada
- Armazenamento Criptográfico Inseguro
- Proteção Insuficiente da Camada de Transporte
- Referências Inseguras e Diretas a Objetos
- Quebra de Autenticação e Gerenciamento de Sessões
- Tratamento Inadequado de Erros
- Redirecionamentos e Encaminhamentos não Validados
- Falha ao Restringir o Acesso à URL

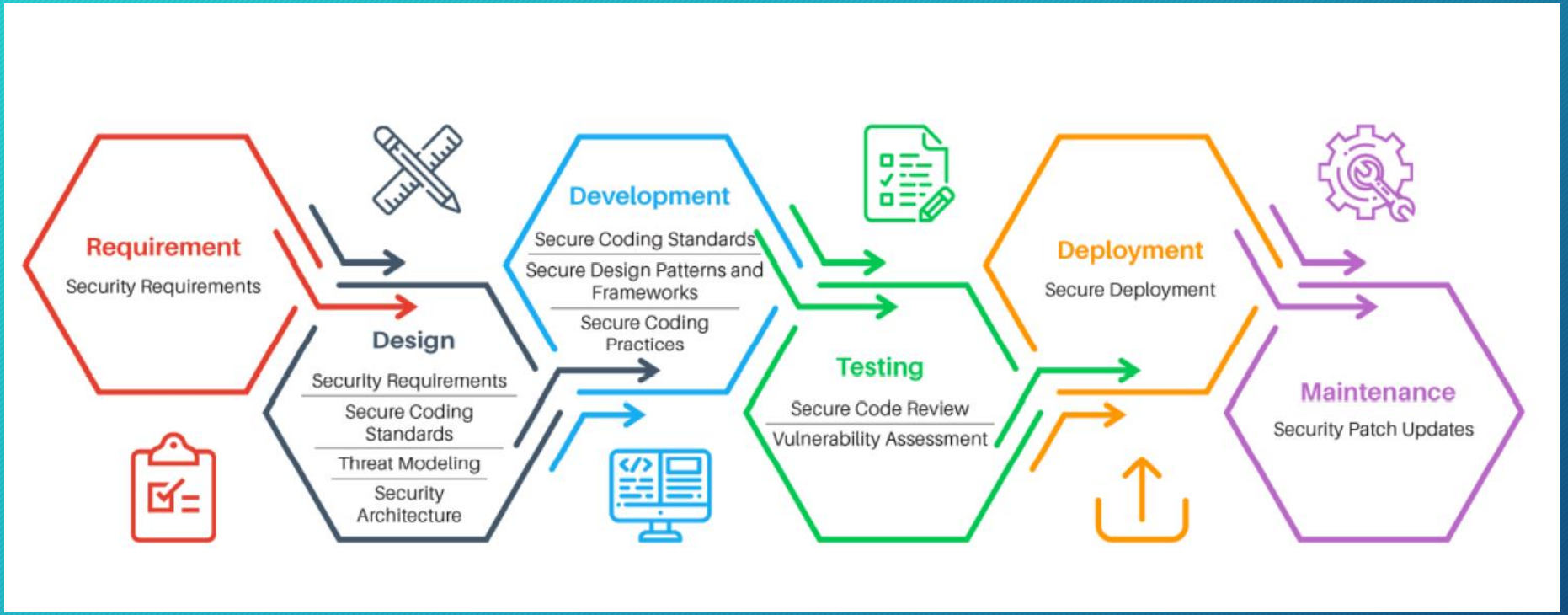
TESTES DE APLICAÇÃO WEB



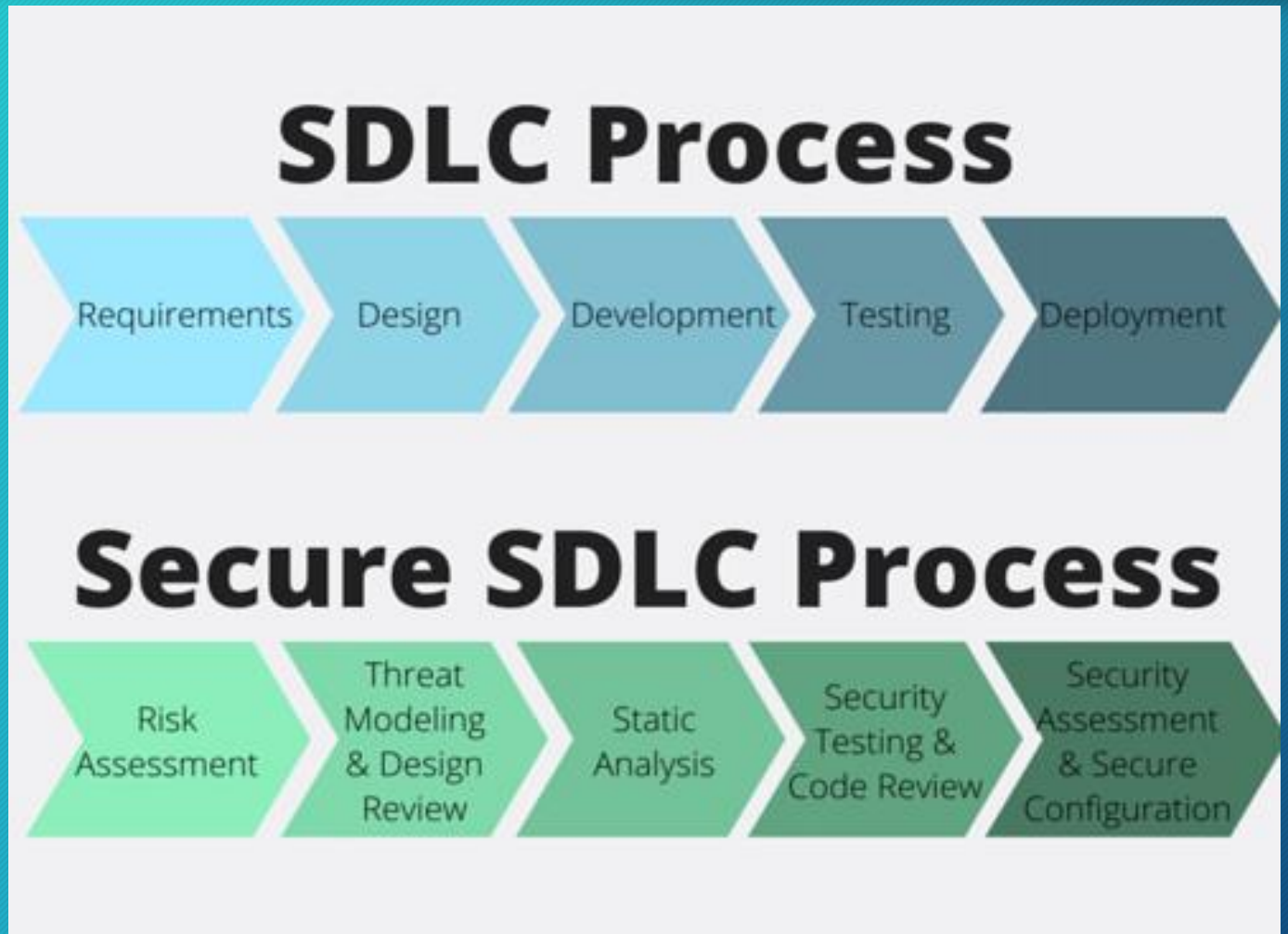
TESTES DE APLICAÇÃO WEB

- Teste Dinâmico de Segurança de Aplicação, testa as interfaces expostas, em busca de vulnerabilidades.
- SAST, que em Português é um acrônimo para Teste Estático de Segurança de Aplicação, analisa o código fonte dos sistemas. Os testes normalmente são realizados antes que o sistema esteja em produção.
- Teste Interativo de Segurança de Aplicação é basicamente a combinação dos modelos de testes estáticos e dinâmicos (SAST e DAST), e apresenta melhores resultados.
- As ferramentas SCA rastreiam os projetos de software de uma organização para detectar componentes de código aberto com vulnerabilidades conhecidas e fornecer informações de segurança detalhadas sobre as vulnerabilidades para ajudar os desenvolvedores a remediá-las rapidamente.
- Um firewall de aplicativos Web filtra, monitora e bloqueia o tráfego HTTP de e para um aplicativo ou site da Web. Um WAF é diferenciado de um firewall comum em que um WAF é capaz de filtrar o conteúdo de aplicativos web específicos, enquanto os firewalls comuns servem como um portão de segurança entre servidores.
- O gerenciamento de bots é uma estratégia que permite filtrar quais bots têm permissão para acessar seus ativos da web. Com essa estratégia, você pode permitir bots úteis, como os rastreadores do Google, enquanto bloqueia bots maliciosos ou indesejados, como os usados para ataques cibernéticos. As estratégias de gerenciamento de bots são projetadas para detectar a atividade do bot, identificar a origem do bot e determinar a natureza da atividade.
- Runtime Application Self-Protection RASP é uma tecnologia de segurança emergente que permite que as organizações parem as tentativas de hackers de comprometer aplicativos e dados corporativos. Construída em um aplicativo ou ambiente de tempo de execução de aplicativo, a tecnologia RASP é capaz de controlar a execução de aplicativos, detectar vulnerabilidades e prevenir ataques em tempo real. Uma solução RASP incorpora segurança ao aplicativo em execução, onde quer que ele resida em um servidor. Por ser baseada em servidor, a segurança RASP é capaz de detectar, bloquear e mitigar ataques imediatamente, protegendo os aplicativos enquanto são executados em tempo real, analisando o comportamento e o contexto do aplicativo. Ao usar o aplicativo para monitorar continuamente seu próprio comportamento, o RASP tem a capacidade de proteger um aplicativo contra roubo de dados.

SDLC Seguro


Requirement
Security Requirements
Design
Security Requirements
Secure Coding Standards
Threat Modeling
Security Architecture
Development
Secure Coding Standards
Secure Design Patterns and Frameworks
Secure Coding Practices
Testing
Secure Code Review
Vulnerability Assessment
Deployment
Secure Deployment
Maintenance
Security Patch Updates

SDLC Seguro


SDLC Process
Requirements
Design
Development
Testing
Deployment
Secure SDLC Process
Risk Assessment
Threat Modeling & Design Review
Static Analysis
Security Testing & Code Review
Security Assessment & Secure Configuration

METODOLOGIAS E FRAMEWORKS

UNICIV
Pós-Graduação EAD em TI

OWASP-TOP 2017

A1	Injeção
A2	Quebra de Autenticação e Gerenciamento de Sessão
A3	Cross-Site Scripting (XSS)
A4	Quebra de Controle de Acesso
A5	Configurações de Segurança Incorretas
A6	Exposição de Dados Sensíveis
A7	Proteção Insuficiente Contra Ataques
A8	Cross-site Request Forgery (CSRF)
A9	Utilização de Componentes Vulneráveis
A10	APIs Desprotegidas

WASC

O Web Application Security Consortium (WASC) é um grupo internacional de especialistas, profissionais do setor e representantes organizacionais que produzem padrões de segurança para código aberto e boas práticas que são amplamente aceitas para a World Wide Web.

SAMM

O Software Assurance Maturity Model (SAMM) é um framework aberto para ajudar as organizações a formular e implementar uma estratégia de **segurança de software** adaptada aos riscos específicos enfrentados pela organização.

- O SAMM ajuda você a:

1. Avaliar as práticas de segurança de software existentes na organização.
2. Criar um programa balanceado que garanta a segurança de software em iterações bem definidas.
3. Demonstrar melhorias concretas no programa de segurança.
4. Demonstrar melhorias concretas no programa de segurança.

BSIMM

- O principal objetivo do BSIMM é permitir que a organização **analise** e **implemente os recursos de segurança** necessários para a organização, avaliando os recursos de segurança implementados com mais frequência nas outras empresas.
- O BSIMM é composto de um **framework de segurança de software** utilizado para organizar as **112 atividades** usadas para avaliar iniciativas.
- O framework consiste em **12 práticas** organizadas em quatro domínios.

Building Security In Maturity Model

Framework de Segurança de Software (SSF)			
Governança	Inteligência	SSDL Touchpoints	Implantação
Estratégia e Métricas	Modelos de Ataque	Análise de Arquitetura	Teste de Invasão
Conformidade e Política	Recursos e Design de Segurança	Revisão de Código	Ambiente de Software
Treinamento	Padrões e Requisitos	Teste de Segurança	Gerenciamento de Configuração e Vulnerabilidades

Fonte: https://www.bsimm.com/

SAMM vs BSIMM

BSIMM	OpenSAMM
Modelo descritivo	Modelo prescritivo
Possui 12 práticas de segurança que consistem em 112 atividades	Possui 12 práticas de segurança que consistem em 72 atividades
Com base nas coisas que realmente são seguidas em uma organização	Com base em algumas atividades desenvolvidas para segurança de software
Possui uma comunidade ativa que permite às organizações entender os recursos de segurança seguidos em outras organizações	Não possui atividades comunitárias ativas

THREAT MODEL

- A modelagem de ameaças é um processo de **identificação**, **análise** e **mitigação das ameaças** para a aplicação
- É uma abordagem estruturada que permite ao desenvolvedor **classificar as ameaças** com base na arquitetura e implementação da aplicação
- É realizada na **fase de design** do ciclo de vida do desenvolvimento seguro
- É um **processo iterativo** que começa na fase de design e repete por todo o ciclo de vida da aplicação até que todas as possíveis **ameaças da aplicação** sejam identificadas
- A saída da modelagem de ameaças é um modelo de ameaças que expõe todas as **ameaças** e **vulnerabilidades** possíveis em uma aplicação

STRIDE

- O modelo STRIDE categoriza as ameaças do aplicativo com base nos **objetivos** e **na finalidade do ataque**, que ajuda o desenvolvedor a **desenvolver uma estratégia de segurança**.
- Também inclui as **contramedidas** para todas as categorias de ameaças.

- Este modelo descreve as seguintes categorias de ameaças:

Propriedade desejada	Ameaças	Descrição
Autenticação	Spoofing	Acesso não autorizado a um sistema usando uma identidade falsa
Integridade	Tampering	codificar uma modificação de dados sem autorização
Não repúdio	Repudiation	Capacidade dos usuários de reivindicar que não executam algumas ações contra o aplicativo
Confidencialidade	Information Disclosure	Exposição indesejada de dados privados a usuários não autorizados
Disponibilidade	Denial of Service	Capacidade de negar um sistema ou aplicativo indisponível aos usuários legítimos
Autorização	Elevation of Privilege	Capacidade do usuário com privilégios limitados de elevar seus privilégios com um aplicativo sem autorização

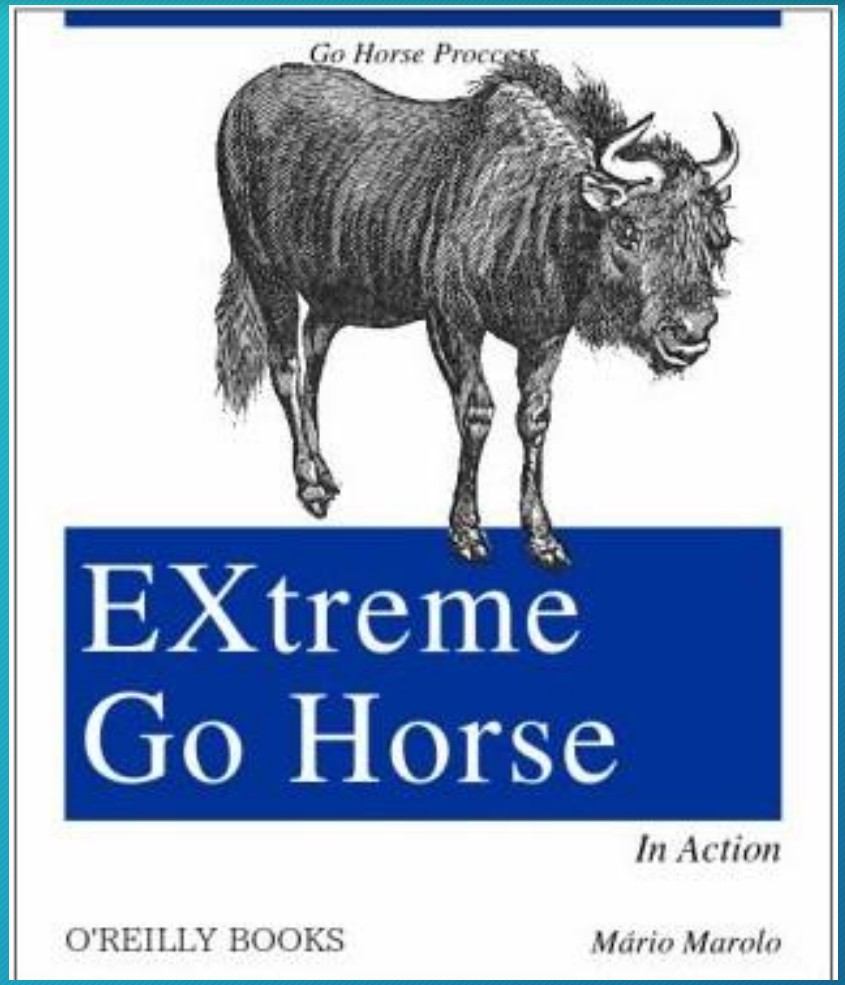
DREAD

Modelo DREAD

- DREAD significa potencial de dano, reprodutibilidade, capacidade de exploração, usuários afetados e capacidade de descoberta
- O modelo DREAD é usado para classificar as diversas ameaças à segurança na aplicação, **calculando o risco de cada ameaça**
- Os níveis de gravidade são atribuídos a diversas ameaças na aplicação, a fim de **atenuar essas ameaças conforme sua gravidade**

	Rating	Alto (3)	Médio (2)	Baixo (1)
D	Damage Potential	O invasor pode subverter o sistema de segurança; obtenha autorização de confiança completa executada como administrador; carregar conteúdo	Vazando informações confidenciais	Vazando informações triviais
R	Reproducibility	O ataque pode ser reproduzido todas as vezes e não requer uma janela de tempo	O ataque pode ser reproduzido, mas apenas com uma janela de tempo e uma situação de corrida específica	O ataque é muito difícil de reproduzir, mesmo com o conhecimento da falha de segurança
E	Exploitability	Um programador iniciante pode fazer o ataque em pouco tempo	Um programador qualificado pode fazer o ataque e repetir os passos	O ataque exige uma pessoa extremamente qualificada e um conhecimento profundo a todo momento para explorar
A	Affected Users	Todos os usuários, principais clientes com configuração padrão	Alguns usuários, configuração não padrão	Porcentagem muito pequena de usuários, recurso obscuro afeta usuários anônimos
D	Discoverability	As informações publicadas explicam o ataque. A vulnerabilidade é encontrada no recurso mais usado e é muito perceptível	A vulnerabilidade em uma parte do produto raramente usada e apenas alguns usuários devem se deparar com ela. Levaria algum tempo de reflexão para ver o uso malicioso.	O bug é obscuro e é improvável que os usuários calculem o potencial de dano

GOHORSE



MEME

Vulnerabilidades por Linguagem de Programação - Veracode

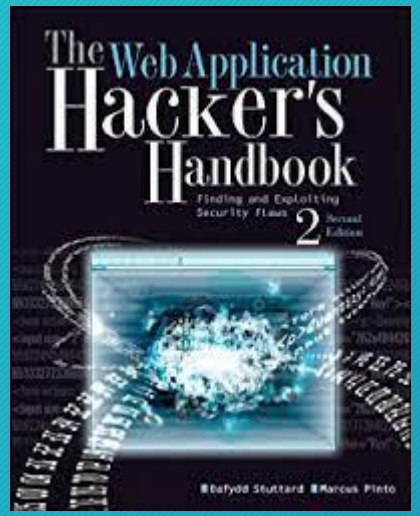
	.Net	C++	Java	JavaScript	PHP	Python
1	Information Leakage 62.8%	Error Handling 66.5%	CRLF Injection 64.4%	Cross-Site Scripting (XSS) 31.5%	Cross-Site Scripting (XSS) 74.6%	Cryptographic Issues 35.0%
2	Code Quality 53.6%	Buffer Management Errors 46.8%	Code Quality 54.3%	Credentials Management 29.6%	Cryptographic Issues 71.6%	Cross-Site Scripting (XSS) 22.2%
3	Insufficient Input Validation 48.8%	Numeric Errors 45.8%	Information Leakage 51.9%	CRLF Injection 28.4%	Directory Traversal 64.6%	Directory Traversal 20.6%
4	Cryptographic Issues 45.9%	Directory Traversal 41.9%	Cryptographic Issues 43.3%	Insufficient Input Validation 25.7%	Information Leakage 63.3%	CRLF Injection 16.4%
5	Directory Traversal 35.4%	Cryptographic Issues 40.2%	Directory Traversal 30.4%	Information Leakage 22.7%	Untrusted Initialization 61.7%	Insufficient Input Validation 8.3%
6	CRLF Injection 25.3%	Code Quality 36.6%	Credentials Management 26.5%	Cryptographic Issues 20.9%	Code Injection 48.0%	Information Leakage 8.3%
7	Cross-Site Scripting (XSS) 24.0%	Buffer Overflow 35.3%	Cross-Site Scripting (XSS) 25.2%	Authentication Issues 14.9%	Encapsulation 48.0%	Server Configuration 8.1%
8	Credentials Management 19.9%	Race Conditions 30.2%	Insufficient Input Validation 25.2%	Directory Traversal 11.5%	Command or Argument Injection 45.4%	Credentials Management 7.2%
9	SQL Injection 12.7%	Potential Backdoor 25.0%	Encapsulation 18.1%	Code Quality 7.6%	Credentials Management 44.3%	Dangerous Functions 6.9%
10	Encapsulation 12.4%	Untrusted Initialization 22.4%	API Abuse 16.2%	Authorization Issues 4.0%	Code Quality 40.3%	Authorization Issues 6.8%

CONCLUSÃO

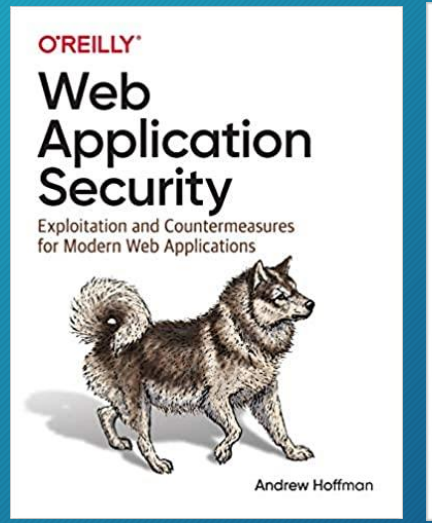
O que leva ao estado de segurança de software deste ano? É natureza ou criação? São os atributos do aplicativo que o desenvolvedor herda - é a dívida de segurança, seu tamanho - ou são as ações dos desenvolvedores - com que frequência eles fazem a varredura de segurança ou como a segurança é integrada em seus processos? E se for a “natureza”, há algo que os desenvolvedores ou profissionais de segurança podem fazer para melhorar os resultados de segurança? - Veracode

LIVROS


Alice & Bob learn
APPLICATION SECURITY
Tanya Janca
WILEY


The Web Application Hacker's Handbook
Finding and Exploiting Security Flaws
2
Dafydd Stuttard
Marcus Pinto


THE BASICS
THE BASICS OF WEB HACKING
Tools and Techniques to Attack the Web
Josh Pauli


O'REILLY
Web Application Security
Exploitation and Countermeasures for Modern Web Applications
Andrew Hoffman


O'REILLY
Segurança para Desenvolvedores Web
USANDO JAVASCRIPT, HTML E CSS
novatec
John Paul Mueller

CREDITOS

https://acaditi.com.br/case-java-certified-application-security-engineer/
https://www.veracode.com/state-of-software-security-report
https://portswigger.net/
https://www.insightassessment.com/article/insight-reports-and-analysis
https://www.csoonline.com/article/3245748/what-is-devsecops-developing-more-secure-applications.html
https://www.csoonline.com/article/3186699/what-it-takes-to-become-an-application-security-engineer.html
https://www.csoonline.com/article/3157377/open-source-software-security-challenges-persist.html
https://www.csoonline.com/article/2157785/five-new-threats-to-your-mobile-security.html
https://www.csoonline.com/article/2978858/is-poor-software-development-the-biggest-cyber-threat.html
https://www.softwaresecured.com/what-do-sast-dast-iast-and-rasp-mean-to-developers/#:~:text=IAST%20is%20designed%20to%20address,QA%20or%20even%20in%20production
João Paulo de Andrade / Filipi Pires
https://sqlwiki.netspi.com/attackQueries/readingAndWritingFiles/#mysql
https://infosecwriteups.com/sql-injection-with-load-file-and-into-outfile-c62f7d92c4e2
https://infosecwriteups.com/tagged/csrf
https://github.com/payloadbox/xss-payload-list

Contato

Meu LinkedIn

https://www.linkedin.com/in/joas-antonio-dos-santos