SEGURANÇA DE APLICAÇÕES WEB.

PREVENÇÕES E MÉTODOS DE ATAQUES

Introdução

- A dependência de sistemas online exige que medidas de proteção sejam implementadas de forma rigorosa para mitigar riscos e evitar ataques cibernéticos.

- A negligência nesse aspecto pode resultar em prejuízos financeiros, danos à reputação e comprometimento da privacidade dos usuários.

História

Final da década de 1960

- Criação da ARPANET pelo Departamento de Defesa dos EUA, precursor da Internet.

Década de 1980

- Surgimento de vírus e malwares, levando ao desenvolvimento dos primeiros antivírus.

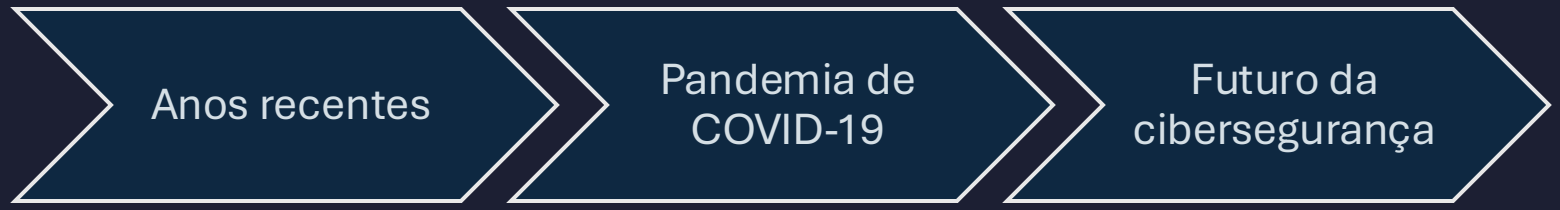
Década de 1990

- Explosão da popularidade da Internet e aumento do cibercrime; surgem firewalls e sistemas de detecção de intrusão.

Início dos anos 2000

- Foco na proteção de dados pessoais com o crescimento do comércio eletrônico e redes sociais; surgem regulamentações como GDPR, LGPD e CCPA.

História



- Adoção da Internet das Coisas (IoT) e aumento da sofisticação dos ataques cibernéticos.
- Aceleração do trabalho remoto e do comércio online, tornando a cibersegurança ainda mais crítica.
- Evolução contínua para enfrentar novas ameaças e proteger informações sensíveis.

Métodos e Prevenções: MITM

MITM (Man in the Middle)

- Interceptação de comunicação entre dois pontos.

Prevenção

- Criptografia SSL/TLS, autenticação de dois fatores (2FA).

Métodos e Prevenções: Falsificação de DNS

Falsificação de DNS

- Redirecionamento de tráfego para sites fraudulentos.

Prevenção

- Uso de DNS seguro (DNSSEC), configuração adequada de servidores DNS.

Métodos e Prevenções: Phishing

Phishing

- Engano de usuários para coletar dados sensíveis.

Prevenção

- Treinamento de usuários, filtros de e-mail, autenticação multi-fatores.

Métodos e Prevenções: Ransomware

Ransomware

- Criptografar arquivos e exigir resgate.

Prevenção

- Backup regular, software antivírus atualizado, não clicar em links suspeitos.

Métodos e Prevenções: Cavalo de Troia

Cavalo de Troia (Trojan)

- Software malicioso disfarçado de programa legítimo.

Prevenção

- Monitoramento de sistemas, antivírus, não baixar software de fontes não confiáveis.

Métodos e Prevenções: DDoS

DoS ou DDoS (Distributed Denial of Service)

- Sobrecarga de servidores com tráfego excessivo.

Prevenção

- Uso de firewalls, balanceadores de carga, redes de distribuição de conteúdo (CDN).

Métodos e Prevenções: Injeção de SQL

Injeção de SQL (SQL Injection)

- Injeção de código malicioso em bancos de dados.

Prevenção

- Validação e higienização de entradas, uso de consultas preparadas.

Métodos e Prevenções: XSS

XSS(Cross site scripting)

- Explora vulnerabilidades para injetar scripts maliciosos no navegador de usuários

Prevenção

- Validação de entrada no servidor e no cliente

Ferramentas e Frameworks

Linguagens

- Python
- Go
- JavaScript/TypeScript
- C#
- PHP
- Ruby
- Java
- Rust

Ferramentas e Frameworks

OWASP ZAP (Código aberto, testes de penetração)

- Testes automáticos de segurança e análise de vulnerabilidades.
- Interceptação de requisições HTTP para análise de tráfego.
- Identificação de falhas como SQL Injection, XSS e CSRF.

Ferramentas e Frameworks

Burp Suite (Popular entre pentesters)

- ➢ Proxy HTTP/HTTPS para interceptação de requisições.
- ➢ Scanner automatizado para XSS, SQL Injection e outras falhas.
- ➢ Ferramentas como "Intruder" e "Repeater" para análise manual.

Ferramentas e Frameworks

Metasploit Framework (Exploração de vulnerabilidades)

- ➢ Plataforma para testes de penetração e desenvolvimento de exploits.
- ➢ Automação de ataques simulados.
- ➢ Testes de segurança em sistemas e redes.

Ferramentas e Frameworks

Spring Security (Java) (Segurança para aplicações Java)

- ➢ Autenticação e autorização (JWT, OAuth2).
- ➢ Proteção contra CSRF, XSS e injeções.
- ➢ Integração com frameworks Spring.

Ferramentas e Frameworks

Django Security (Python) (Segurança nativa no Django)

- ➢ Proteção embutida contra CSRF, XSS, SQL Injection e Clickjacking.
- ➢ Sistema de autenticação e gerenciamento seguro de senhas.
- ➢ Controle de sessões e acessos.

Ferramentas e Frameworks

ASP.NET Core Security (C#) (Segurança para aplicações .NET)

- Autenticação baseada em token (JWT), MFA e Identity Framework.
- Proteção contra CSRF, XSS e injeção de SQL.
- Criptografia de dados e proteção de senhas.

Casos Reais

Ataque à Equifax (2017)

- Exploração de falha no **Apache Struts** (CVE-2017-5638).
- Vazamento de dados sensíveis de milhões de usuários.
- Multas, ações judiciais e dano irreparável à reputação.

Casos Reais

Ataque ao Yahoo (2013-2014)

- Comprometimento de **3 bilhões de contas**.
- Falhas na **criptografia de senhas e autenticação**.
- Perda de confiança, multas e venda da empresa à Verizon.

Casos Reais

Ataque à PlayStation Network (2011)

- Invasão pelo grupo **LulzSec**.
- Roubo de dados de **77 milhões de contas**.
- PSN ficou fora do ar **por mais de 3 semanas**, causando prejuízos milionários.

Modelagem de Ameaças

O que é?	Método	Exemplo: Comércio Eletrônico
• Identificação e análise de riscos no sistema. • Utilizado em todas as etapas do desenvolvimento.	• Diagrama de fluxo de dados para mapear ativos e interações. • Identificação de ameaças com frameworks como STRIDE e OWASP Top 10. • Definição de mitigação ou aceitação de riscos.	• Atores: usuários, funcionários, atacantes. • Ativos: dados de pagamento, identidade, estoque. • Possíveis ameaças: roubo de cartões, acesso não autorizado. • Controles: autenticação, restrição de acesso, validação de dados.

Treinamento de Desenvolvedores

- Por que é essencial?
  - Previne falhas de segurança desde a fase de desenvolvimento.
  - OWASP (2023): maioria das vulnerabilidades pode ser evitada com boas práticas.
  - Verizon (2023): 90% das violações poderiam ser prevenidas.

- Principais tópicos abordados
  - Vulnerabilidades comuns: SQL Injection, XSS, CSRF.
  - Práticas seguras: validação de entrada, criptografia, menor privilégio.
  - Ferramentas essenciais: OWASP ZAP, Burp Suite, SonarQube.
  - Segurança para APIs e microsserviços: OAuth 2.0, JWT, proteção contra força bruta.

- Métodos de treinamento eficazes
  - Workshops práticos e simulações de ataques.
  - Hackathons de segurança e desafios gamificados (Secure Code Warrior).
  - Certificações: CSSLP e outras formações especializadas.

Conclusão

Por que a segurança é essencial?

- Protege dados e sistemas contra ameaças cibernéticas.
- Minimiza riscos como roubo de informações, fraudes e interrupções operacionais.
- Evita impactos financeiros, sanções legais e perda de reputação.

Compromisso contínuo com a segurança

- Segurança não é um evento único, mas um processo contínuo.
- Empresas devem integrar segurança em todas as etapas do desenvolvimento.
- Proteção de usuários e continuidade dos negócios dependem de medidas proativas e atualizações constantes.

Obrigado!

Alunos:JOSIAS, KAICK, KELLY, MARCOS, PEDRO, RAPHAEL