
🛡️ **Guia Didático: Segurança da Informação para Iniciantes**

📍 Com indicação de **responsabilidade: Desenvolvedor / Servidor**

---

1. Content Security Policy (CSP) Ausente ou Insegura

📌 **Problema**: Scripts externos podem ser executados no site.
✅ **Solução**: Adicionar o cabeçalho Content-Security-Policy.

🛠️ **Responsável**:

- ✅ **Servidor (operações)**: configurando o cabeçalho na resposta HTTP (ex: Apache, Nginx).
- ✅ **Desenvolvedor**: definir regras adequadas para cada rota, recurso ou página.

---

2. Cookies sem HttpOnly, Secure e SameSite

📌 **Problema**: Cookies podem ser roubados ou vazados.
✅ **Solução**: Definir os atributos de segurança nos cookies.

🛠️ **Responsável**:

- ✅ **Desenvolvedor**: ao criar cookies no código (ex: PHP, JavaScript).
- ✅ **Servidor**: pode reforçar as políticas via configuração global.

---

3. Cabeçalhos HTTP que Revelam Tecnologia

📌 **Problema**: Facilita reconhecimento de tecnologias vulneráveis.
✅ **Solução**: Remover ou esconder cabeçalhos como X-Powered-By, Server.

🛠️ **Responsável**:

- ✅ **Servidor (operações)**: removendo no Apache (ServerTokens, Header unset) ou Nginx.
- ✅ **Desenvolvedor** (em APIs ou frameworks): desabilitar no framework (ex: Express.js, Laravel).

---

4. Falta de Proteção Anti-CSRF (Cross-Site Request Forgery)

📌 **Problema**: Um site externo pode fazer requisições maliciosas autenticadas.
✅ **Solução**: Usar tokens CSRF nos formulários e validação no servidor.

🛠️ **Responsável**:

- ✅ **Desenvolvedor**: gerar, incluir e validar o token no backend.
- 🚫 **Servidor**: geralmente não tem controle direto sobre isso.

---

5. Falta do Cabeçalho X-Frame-Options (Clickjacking)

📌 **Problema**: O site pode ser carregado por <iframe> malicioso.
✅ **Solução**: Adicionar cabeçalho X-Frame-Options.

🛠️ **Responsável**:

- ✅ **Servidor (operações)**: configurando cabeçalho.
- ✅ **Desenvolvedor**: pode definir o cabeçalho em frameworks (ex: Express, Laravel).

---

6. Cookies Inseguros

📌 **Problema**: Sessões podem ser sequestradas se cookies forem mal configurados.
✅ **Solução**: Criptografar o conteúdo, limitar domínio e definir expiração.

🛠️ **Responsável**:

- ✅ **Desenvolvedor**: gerar cookies de forma segura.
- ✅ **Servidor**: pode aplicar políticas padrão (ex: HTTPS obrigatório).

---

7. Vazamento de Informações nos Cabeçalhos HTTP

📌 **Problema**: Cabeçalhos como Server: Apache/2.4.18 expõem a tecnologia.
✅ **Solução**: Remover ou mascarar essas informações.

🛠️ **Responsável**:

- ✅ **Servidor (operações)**: editar configurações do Apache/Nginx.
- ✅ **Desenvolvedor**: verificar se APIs ou apps estão expondo dados extras.

---

8. Falta de Sanitização de Entradas

📌 **Problema**: Permite XSS, SQL Injection, etc.
✅ **Solução**: Validar, filtrar e escapar os dados recebidos do usuário.

🛠️ **Responsável**:

- ✅ **Desenvolvedor**: sempre! Essa é uma responsabilidade de quem escreve o código.
- 🚫 **Servidor**: não tem como proteger diretamente.

📊 **Tabela-Resumo**

Vulnerabilidade	**Correção/Responsável**
CSP Ausente ou Insegura	Desenvolvedor + Servidor
Cookies sem atributos seguros	Desenvolvedor + Servidor
Cabeçalhos que revelam tecnologia	Servidor + Desenvolvedor
Falta de proteção CSRF	Desenvolvedor
Falta do X-Frame-Options	Desenvolvedor + Servidor
Cookies inseguros	Desenvolvedor + Servidor
Vazamento de cabeçalhos	Servidor + Desenvolvedor
Falta de sanitização de entradas	Desenvolvedor
