Revista Científica UNAR (ISSN 1982-4920), Araras (SP), v.23, n.1, p.61-80, 2023.
DOI: 10.18762/1982-4920.20230005

ESTUDO DE CASO DE BOAS PRÁTICAS DE DESENVOLVIMENTO SEGURO EM APLICAÇÕES WEB

Lucas Gibelli Claro, Raul Fanti Tellaroli, Wdson de Oliveira

RESUMO

Este aborda boas práticas de desenvolvimento seguro de aplicações web, visando torná-las mais seguras, mitigando riscos provenientes das vulnerabilidades dessas aplicações. São apresentadas algumas boas práticas relevantes para fortalecer a segurança dessas aplicações, considerando um guia renomado como o da OWASP, para prevenir as vulnerabilidades mais comuns nesse tipo de aplicação, e com isso eliminar algumas das vulnerabilidades utilizadas em ataques cibernéticos. Considerando o aumento de crimes digitais e vazamentos de dados, o desenvolvimento seguro de aplicações web se tornou essencial para proteger informações sensíveis e garantir a confiança dos usuários. Nesse sentido, o trabalho destacou a necessidade de incorporar práticas de segurança, como autenticação, autorização, criptografia e validação de entrada, desde a concepção do projeto até a sua manutenção. Ao utilizar a OWASP como referência, o estudo se apoia em diretrizes e ferramentas amplamente reconhecidas para identificar e corrigir vulnerabilidades em aplicações web. Isso possibilita aos desenvolvedores adotar uma abordagem proativa, mitigando riscos cibernéticos e evitando futuros incidentes de segurança desde o primeiro contato com o desenvolvimento das aplicações.

Palavras chaves: desenvolvimento seguro, aplicações web, vulnerabilidades

ABSTRACT

This paper examines best practices for the secure development of web applications, with the aim of enhancing their security by mitigating risks associated with vulnerabilities inherent to such applications. It outlines several pertinent best practices to strengthen application security, referencing a widely acknowledged guide such as OWASP, to prevent the most prevalent vulnerabilities in this type of application, thereby addressing weaknesses commonly exploited in cyberattacks. Given the increasing prevalence of digital crimes and data breaches, secure web application development has become critical to safeguarding sensitive information and ensuring user trust. In this regard, the paper highlights the necessity of incorporating security measures—including authentication, authorization, encryption, and input validation—from the initial design phase through to ongoing maintenance. By leveraging OWASP as a benchmark, the study draws upon recognized guidelines and tools to identify and remediate vulnerabilities in web applications. This approach empowers developers to adopt a proactive stance, effectively mitigating cyber risks and preventing potential security incidents from the outset of the development lifecycle.

Keywords: Secure Development; Web Applications; Vulnerabilities.

1. INTRODUÇÃO

Nunca foi tão discutido a Segurança da Informação como nos dias de hoje. E não é para menos, visto as inúmeras ameaças digitais que atacam milhares de pessoas e empresas diariamente. E, para garantir uma boa discussão em torno desse assunto, é necessário levar em consideração todos os seus pilares: confidencialidade, integridade, disponibilidade, autenticidade e irretratabilidade.

Os sites e aplicações web são um dos recursos mais acessados hoje em dia. Tais como lojas virtuais, *marketplaces*, redes sociais como Facebook e Twitter, todos esses gêneros de sites são acessados diariamente e em grande escala, por pessoas leigas e também por profissionais da área, o que pode colaborar com o sucesso de um possível ataque de Engenharia Social, por exemplo, já que as pessoas leigas podem ser facilmente enganadas.

Dentro desses sites trafegam um grande número de informações, e, muitas delas, são dados sensíveis, tais como: senhas, CPFs, e-mails, credenciais de acesso, dados bancários, dados de cartões de crédito, etc.

Sendo assim, surge a necessidade de proteger essas informações, pois, caso elas caiam em mãos erradas, o proprietário da informação pode ter diversos prejuízos financeiros e pessoais, e se a empresa proprietária do site ou aplicação tenha negligenciado cuidar da segurança dessas informações, ou, de alguma forma facilitado com que algum agente malicioso tivesse acesso à essa informação, a mesma poderia ter problemas sérios relacionados à justiça como, por exemplo, receber um processo judicial em virtude de vazamento de dados.

Com isso, surge a necessidade de proteger essas informações que são trafegadas e, uma das etapas de proteção partem do desenvolvimento seguro dessas plataformas. É muito importante que os desenvolvedores das aplicações web e sites cuidem da segurança deles durante as etapas de desenvolvimento. Para ajudar nessa tarefa, existem algumas boas práticas de desenvolvimento seguro a serem seguidas e, dentre elas, a que será abordado nesse documento: as metodologias da OWASP (*Open Web Application Security Project*).

2. REFERÊNCIAL TEÓRICO

Este capitulo tem como principal objetivo contextualizar os conceitos tratados no artigo em questão.

2.1 SEGURANÇA DA INFORMAÇÃO

De acordo com o artigo "Segurança da Informação" publicado pela DocuSign em dezembro de 2022, a segurança da informação consiste em uma série de ações estrategicamente

adotadas para controlar e prevenir riscos relacionados ao roubo, danos e perdas de dados, dispositivos, servidores, sistemas, redes e sites.

As práticas de proteção da informação consistem em um conjunto de processos realizados de forma sincronizada, com o objetivo de proteger ativos virtuais e físicos relacionados à informação, independentemente de como eles são editados, compartilhados, processados ou arquivados.

A segurança da informação é um campo que requer cuidados específicos e um gerenciamento de alto nível. No entanto, apesar de parecer rígida inicialmente, ela pode ser implementada por meio de etapas bem definidas, o que facilita sua aplicação.

Para obter resultados efetivos, é necessário realizar a gestão de riscos em todas essas etapas, a fim de identificar ativos, vulnerabilidades, fontes de ameaças, formas de controle e possíveis impactos das ações executadas.

Diante disso, surgiram cinco pilares fundamentais para servir de base na hora de uma empresa tomar ações voltadas para a segurança da informação. Esses cinco pilares são descritos por David Pedra em seu artigo “Segurança da informação: o que é e como criar uma política para proteção de dados”, em abril de 2023.

O primeiro pilar da segurança da informação é a confidencialidade, que se refere à restrição de acesso aos dados somente a pessoas autorizadas. Medidas como definição de níveis de acesso e controle hierárquico são implementadas para assegurar que as informações sejam acessadas apenas por indivíduos autorizados.

O segundo pilar é a integridade, que diz respeito à preservação dos dados, evitando alterações, danos ou corrupção que possam causar prejuízos à empresa. Para garantir a integridade dos dados, são adotadas ações como backups automáticos, controle de alterações em documentos e manutenção periódica de dispositivos de armazenamento.

A disponibilidade é o terceiro pilar, que envolve assegurar que os dados estejam acessíveis quando necessário. Isso significa que a segurança da informação deve garantir que usuários autorizados possam acessar os ativos de informação a qualquer momento, evitando interrupções nos processos organizacionais e atrasos nas operações. (PEDRA, 2023). [1]

O quarto pilar é a autenticidade, que visa garantir que os dados sejam legítimos e sem intervenções de pessoas não autorizadas que se passam por outras com autorização. É importante assegurar a ausência de falsificação de registros e o rastreamento adequado de todas as ações realizadas pelos usuários. (PEDRA, 2023). [1]

E por último, o quinto pilar é a legalidade, que define que todos os procedimentos relacionados à segurança da informação devem estar de acordo com a legislação vigente, incluindo a Lei Geral de Proteção de Dados Pessoais. Dessa forma, os dados protegidos pela segurança da informação devem atender aos requisitos estabelecidos pela legislação aplicável. (PEDRA, 2023). [1]

Diante disso, entende-se que a segurança da informação é uma disciplina essencial para garantir a proteção dos dados e ativos de uma organização e que ela não se trata de um objeto final, mas sim de um processo contínuo.

Ao seguir os pilares da confidencialidade, integridade, disponibilidade, autenticidade e legalidade, juntamente com uma abordagem abrangente de gestão de riscos, é possível reduzir significativamente os riscos de violação de dados, fortalecer a confiança dos stakeholders na organização e manter a empresa nas normas estabelecidas pelas leis no contexto de proteção de dados.

2.2 DESENVOLVIMENTO SEGURO

A Perallis Security (acessado em 2023) descreve o desenvolvimento seguro como sendo uma abordagem que visa garantir a segurança da informação em aplicativos e sistemas web ou desktop. Para alcançar esse objetivo, são adotadas práticas e medidas que visam prevenir vulnerabilidades e mitigar riscos cibernéticos. Além disso, essa estratégia proativa é implementada desde o início do processo de desenvolvimento, o que permite evitar reações a incidentes de segurança após sua ocorrência.

No contexto atual, em que vazamentos de dados e ataques cibernéticos são cada vez mais comuns, o desenvolvimento seguro tornou-se indispensável para proteger empresas contra ameaças. Afinal, a falta de cuidado dos desenvolvedores durante a criação de aplicativos e sistemas web tem sido uma das principais causas de incidentes de segurança. Assim, é necessário enfatizar a incorporação de medidas de segurança desde o início do processo de desenvolvimento, levando em consideração as preocupações relacionadas à segurança cibernética.

Não há dúvida de que o desenvolvimento seguro é uma abordagem que pode trazer diversos benefícios para as empresas. Ao adotar práticas e medidas de segurança, é possível reduzir os riscos de exposição cibernética, proteger seus dados e preservar sua reputação. Em outras palavras, o desenvolvimento seguro pode ser considerado um investimento que traz retornos positivos a longo prazo, garantindo a tranquilidade e confiança dos usuários e dos clientes das empresas.

Já a Nova8 em outubro de 2021, o desenvolvimento seguro de software (SSDLC) integra a segurança em todas as etapas do ciclo de vida do desenvolvimento. Diferente dos modelos tradicionais, o SSDLC aborda a segurança desde o planejamento até a manutenção do software. Essa abordagem identifica e mitiga vulnerabilidades precocemente, reduzindo custos e riscos associados a falhas de segurança. Adotar o SSDLC garante software mais seguro, envolve todas as partes interessadas, detecta falhas de design antes da implementação e minimiza riscos comerciais relacionados à segurança.

Com isso, conclui-se que o desenvolvimento seguro de aplicativos e sistemas web ou desktop é de extrema importância para garantir a segurança da informação, prevenir vulnerabilidades e mitigar riscos cibernéticos desde o início do processo de desenvolvimento. Com o aumento de vazamentos de dados e ataques cibernéticos, o desenvolvimento seguro tornou-se indispensável para proteger empresas contra ameaças. A adoção do SSDLC permite criar uma aplicação mais segura, robusta e envolve todas as partes interessadas na conscientização das considerações de segurança, a ponto de minimizar os riscos comerciais relacionados à proteção de dados.

2.3 A IMPORTÂNCIA DO DESENVOLVIMENTO SEGURO

Para Andrea Rosti (Gerente de Marketing Digital e especialista em segurança na Safeway), em 22 de outubro de 2022, o desenvolvimento seguro reduz o número de incidentes relacionados à segurança da informação e ajuda a empresa a estar de acordo com as regulamentações e legislações relacionadas com a segurança dos dados.

De acordo com o artigo da CECyber (21 de junho de 2022), grande parte das vulnerabilidades se dão por conta de um de algum descuido no desenvolvimento da aplicação, tornando-a vulnerável a algum tipo de cyber ataque. O desenvolvimento seguro visa mitigar ao máximo essas vulnerabilidades.

A Perallis Security em seu blog (acessado em 07 de dezembro de 2022) informa que, ainda que haja diversos vetores de ataque, a maioria deles se dão por conta da exploração de uma vulnerabilidade, decorrida da falta de cuidado dos desenvolvedores durante a produção de seus aplicativos e sistemas web.

Até mesmo empresas grandes como Rockstar e Uber, que são grandes empresas e com uma grande maturidade em cibersegurança sofreram ataques cibernéticos no último ano de 2022.

Segundo o artigo "Grand Theft Auto 6 Leak: Who Hacked Rockstar and What Was Stolen?" publicado no jornal The Guardian em Setembro de 2022, no dia 18 de setembro, um

indivíduo nomeado como “teapotuberhacker” divulgou uma série de vídeos no fórum GTAForums. Esses vídeos possuíam algo em torno de 50 minutos de gravações sobre uma versão em processo de desenvolvimento do jogo Grand Theft Auto 6, desenvolvido pela Rockstar Games. Após a divulgação desses vídeos, eles se espalharam pelas redes sociais e no restante da internet.

Em dezembro de 2022, o site Olhar Digital divulgou uma notícia sobre o ataque hacker sofrido pela Uber, intitulada "Uber Sofre Novo Ataque Hacker: Dados de 77 Mil Funcionários São Comprometidos". Um agente chamado "UberLeaks" vazou informações roubadas da empresa em um fórum popular de hackers. Os dados vazados incluem o código-fonte de plataformas de gerenciamento de dispositivos móveis (MDM) usadas pela Uber, Uber Eats e serviços terceirizados, como Tectivity e TripActions. Além disso, foram comprometidos relatórios de gerenciamento de TI, relatórios de destruição de dados, nomes de login do Windows, e-mails e outras informações relacionadas à empresa. Esse incidente afeta a segurança de mais de 77 mil funcionários da Uber.

De acordo com o que foi abordado, é possível entender que o desenvolvimento seguro desempenha um papel fundamental na redução de incidentes de segurança da informação e na conformidade com regulamentações e legislações relacionadas à segurança dos dados, conforme destacado por Andrea Rosti, especialista em segurança. Muitas vulnerabilidades surgem de descuidos durante o desenvolvimento de aplicativos, tornando-os suscetíveis a ataques cibernéticos. O desenvolvimento seguro visa mitigar essas vulnerabilidades, uma vez que a maioria dos ataques explora falhas decorrentes da falta de cuidado dos desenvolvedores.

3 METODOLOGIA

Com o intuito de contextualizar o tema do artigo em questão foi desenvolvida uma pesquisa bibliográfica baseada em artigos e sites, procurando fazer um levantamento destacando os principais pontos com relação a vulnerabilidades de segurança em códigos, bem como apresentar as melhores práticas de codificação segura para evitar esses problemas

Para apresentar uma visão sobre as principais vulnerabilidades de segurança em códigos, bem como mostrar as melhores práticas de codificação segura para evitar esses problemas, foi elaborado um cenário que simulasse esses ambientes através de códigos independentes, simples de entendimento, mas o necessário, para contribuir com os objetivos propostos nesse artigo. Ao final, espera-se contribuir para a conscientização sobre a importância da segurança no desenvolvimento de sites e aplicações web e fornecer orientações práticas para

a correção de vulnerabilidades, promovendo a construção de aplicações mais seguras e confiáveis.

Para ajudar nessa tarefa, foram utilizadas as boas práticas de desenvolvimento seguro da OWASP (*Open Web Application Security Project*): XSS (CROSS-SITE SCRIPTING), SQL INJECTION E IDOR (INSECURE DIRECT OBJECT REFERENCE),

4 APLICAÇÃO, ANALISE DOS RESULTADO E DISCUSSÕES

Este capitulo tem como objetivo apresentar as 3 formas de desenvolvimento seguro sugeridas, bem como alguns exemplos práticos de vulnerabilidades, o risco que elas apresentam e a correção das mesmas.

4.1 XSS (CROSS-SITE SCRIPTING)

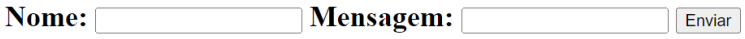
A partir do contexto adquirido anteriormente, podemos seguir para o primeiro dos exemplos, onde será apresentado um código simples escrito em HTML e PHP, de um formulário que receberá duas informações e irá exibi-las em uma página criada com as mesmas linguagens. De acordo com a Figura 1, o código do formulário é este, feito em HTML:

Figura 1: Código HTML

Fonte: Os autores (2023)

Ele exibe dois campos para o usuário possa digitar seu nome e definir uma mensagem conforme desejado. Exibido na Figura 2:

Figura 2: Formulário HTML



Fonte: Os autores (2023)

De acordo com a Figura 3, o código responsável por receber e exibir essas informações é esse:

Figura 3: Código HTML e PHP

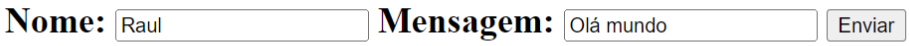
send-message.php > ...

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>XSS</title>
</head>
<body>
    A mensagem recebida foi: <?= $_POST['message']; ?>
</body>
</html>
```

Fonte: Os autores (2023)

Ele exibirá na tela o que o usuário enviar para a aplicação, sem realizar nenhuma tratativa. Figura 4 e Figura 5.

Figura 4: Formulário de entrada



Fonte: Os autores (2023)

Figura 5: Mensagem de saída

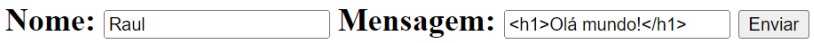
Fonte: Os autores (2023)

Repare que, como dito anteriormente, não há nenhuma tratativa da informação enviada, a aplicação apenas recebeu e exibiu essa informação. Isso deixa nosso código com

uma vulnerabilidade chamada XSS. Uma fragilidade comum em aplicações web que permite a injeção de scripts maliciosos nessas aplicações.

Dessa forma, há a possibilidade de um usuário mal-intencionado inserir tags HTML nesses campos e essas tags acabarem se misturando com o código da aplicação, de forma que o navegador interprete como uma tag HTML. Um exemplo disso é inserindo a tag H1. Demonstrado na Figura 6:

Figura 6: Formulário de entrada



Fonte: Os autores (2023)

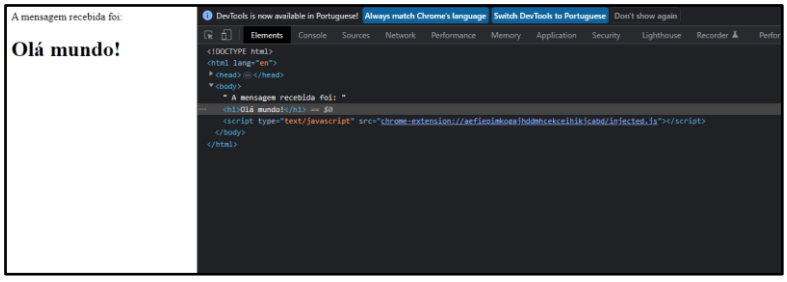
Ela será enviada para a página que exibe as informações e será interpretada pelo navegador. Exemplificado na Figura 7:

Figura 7: Formulário de entrada

Fonte: Os autores (2023)

Repare que no código fonte da página ela também aparece como uma tag que faz parte de nosso site. Exibido na Figura 8:

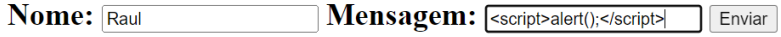
Figura 8:



Fonte: Os autores (2023)

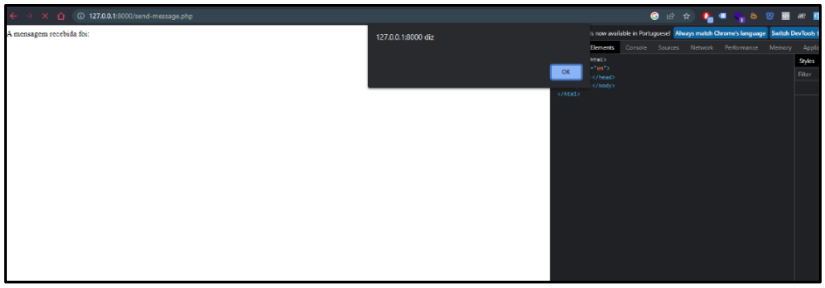
Dessa forma, um agente malicioso pode ir além e inserir tags de script, onde ele pode inserir códigos *javascript* que também serão interpretados pelo navegador. Apresentado na Figura 9 e Figura 10:

Figura 9: Formulário de entrada



Fonte: Os autores (2023)

Figura 10: Mensagem de saída



Fonte: Os autores (2023)

Com isso um atacante pode redirecionar o usuário para uma página de *phishing* ou acessar seus cookies. Isso permite o acesso à sessão e todas as informações do usuário, isso chama-se Session Hijacking. Medidas de segurança, como validação principalmente da saída dados e codificação apropriada, são importantes para prevenir ataques de XSS. A função htmlspecialchars() em PHP é utilizada para tratar informações antes da exibição. Conforme na Figura 11:

Figura 11: Código PHP

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>XSS</title>
</head>
<body>
    A mensagem recebida foi: <?= htmlspecialchars($_POST['message'], ENT_QUOTES, 'UTF-8') ?>
</body>
</html>
```

Fonte: Os autores (2023)

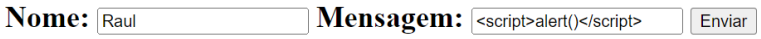
Neste exemplo, a função *htmlspecialchars()* é aplicada aos dados antes de serem exibidos. Os parâmetros utilizados são:

1. O primeiro parâmetro do formulário é a string a ser codificada, no caso, a mensagem que foi recebida via POST.

2. O segundo parâmetro ENT_QUOTES garante que tanto as aspas simples quanto as duplas sejam convertidas em entidades HTML.
3. O terceiro parâmetro 'UTF-8' especifica a codificação dos caracteres.

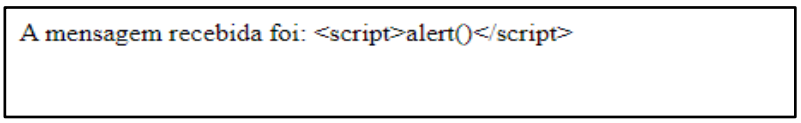
A função htmlspecialchars() converte caracteres especiais em entidades HTML correspondentes, como <, >, ' e ". Isso evita que o código seja interpretado como parte do conteúdo HTML, prevenindo a execução de scripts maliciosos. O resultado final é o apresentado na Figura 12 e Figura 13:

Figura 12: Formulário de entrada



Fonte: Os autores (2023)

Figura 13: Mensagem de saída



Fonte: Os autores (2023)

É importante aplicar essa função sempre que houver saída de dados para uma página web, especialmente quando esses dados são obtidos a partir de entradas do usuário ou de outras fontes externas. A OWASP descreve esse processo como Codificação de Dados de Saída.

4.2 SQL INJECTION

A SQL injection é uma vulnerabilidade de segurança em aplicações web com banco de dados. Ocorre quando um invasor insere ou manipula comandos SQL em uma aplicação que não valida ou filtra corretamente as entradas de dados do usuário. Isso permite que o invasor execute comandos maliciosos no banco de dados, comprometendo a segurança e a integridade dos dados. Pode resultar em vazamento de dados, alterações indevidas e ações não autorizadas no sistema.

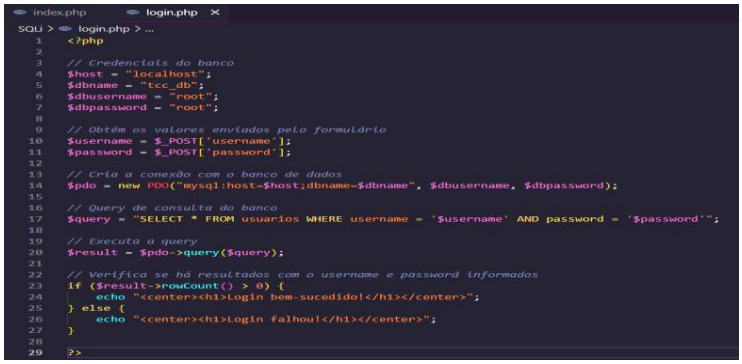
Na Figura 14 pode ser encontrado um exemplo de um formulário simples de login, feito em HTML e processado em PHP:

Figura 14: Código PHP que implementa o formulário de entrada

Fonte: Os autores (2023)

E o código responsável por processar e validar os dados enviados através deste formulário é o que está exibido na Figura 15:

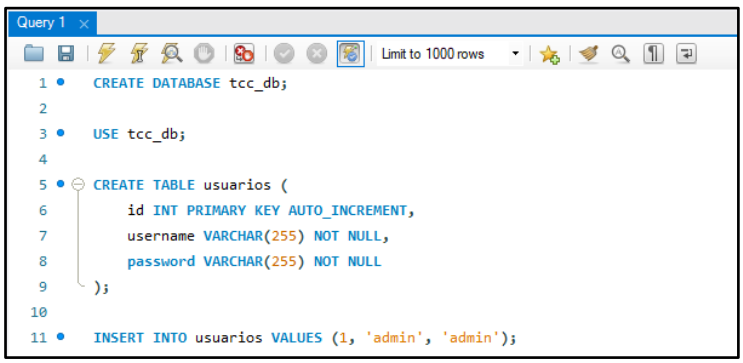
Figura 15: Código PHP para validação



Fonte: Os autores (2023)

O código de criação do banco é este da Figura 16, feito na linguagem SQL:

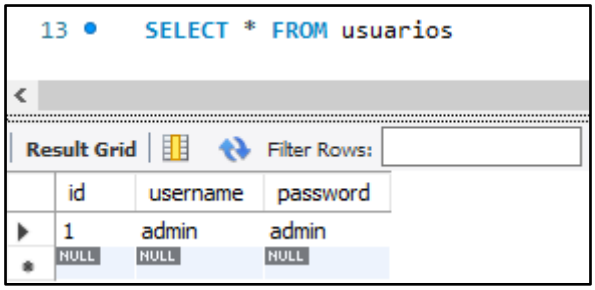
Figura 16: Código de criação do banco de dados



Fonte: Os autores (2023)

O usuário criado no banco de dados é o usuário de nome “admin” e senha “admin”. Conforme a Figura 17:

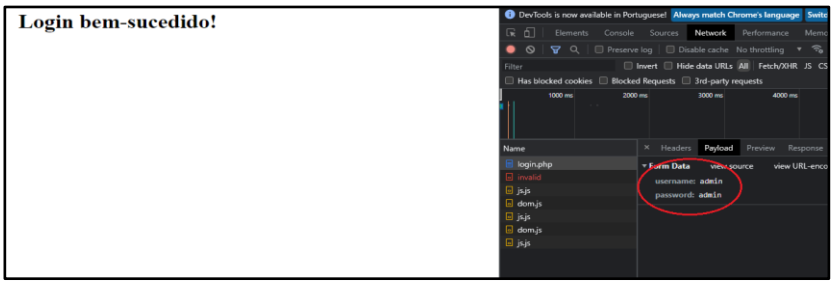
Figura 17: Inserção login do usuário



Fonte: Os autores (2023)

Vale a pena notar que se alguém tentar acessar através das credenciais que estão no banco de dados, o a conexão ocorre normalmente. Conforme mostrado na Figura 18:

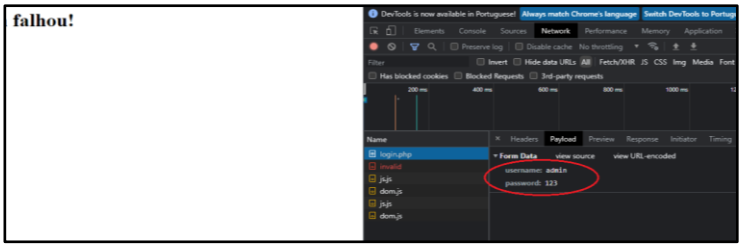
Figura 18: Mensagem de usuário logado.



Fonte: Os autores (2023)

Caso as credenciais sejam incorretas, irá gerar um aviso informando que o login falhou. Conforme pode ser visualizado pela Figura 19:

Figura 19: Mensagem de falha de login

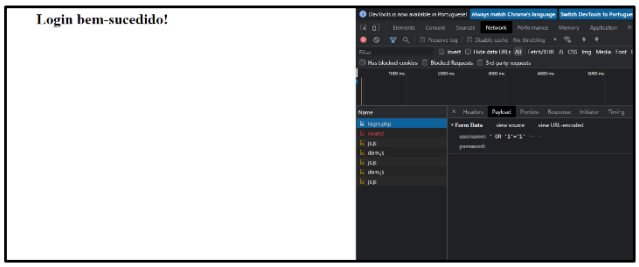


Fonte: Os autores (2023)

Porém, se a entrada de dados, ou seja, os dados que o cliente inserir no formulário não forem devidamente tratados, um atacante pode utilizar *payloads* de SQL Injection para modificar o comportamento da aplicação ao seu favor.

A Figura 20 mostra um exemplo simples disso, utilizando a seguinte *payload* em um dos campos: **' OR '1'='1' -- -**

Figura 20: Mensagem de usuário logado.



Fonte: Os autores (2023)

No código em questão, a vulnerabilidade de SQL Injection ocorre por conta da forma com que a consulta SQL é feita. O código recebe os valores do formulário e eles são diretamente inseridos na consulta SQL, sem validação ou tratamento adequado.

Então se ele inserir **‘ OR ‘1’=’1’ -- -** no campo de nome de usuário e deixar a senha em branco, a consulta SQL ficará assim: **SELECT * FROM usuarios WHERE username = ‘’ OR ‘1’=’1’ -- - AND password = ‘’**.

Após "-- -", os comentários não são processados. A condição OR ‘1’=’1’ será sempre verdadeira, permitindo que o invasor acesse informações de outros usuários ou realize ações indesejadas. Essa vulnerabilidade possibilita a execução de comandos maliciosos na consulta SQL, incluindo exclusão, modificação e obtenção de dados confidenciais. Para prevenir, utilize técnicas seguras de codificação, como prepared *statements* ou consultas parametrizadas, evitando a execução de código SQL malicioso.

Dito isso, podemos proteger esse código utilizando o recurso de *prepared statement*, da forma apresentada na Figura 21:

Figura 21: Código PHP para proteger da injeção de SQL

Fonte: Os autores (2023)

E, ao inserir a mesma payload, o login falhará. Como podemos visualizar na Figura 22:

Figura 22: Mensagem de falha de login

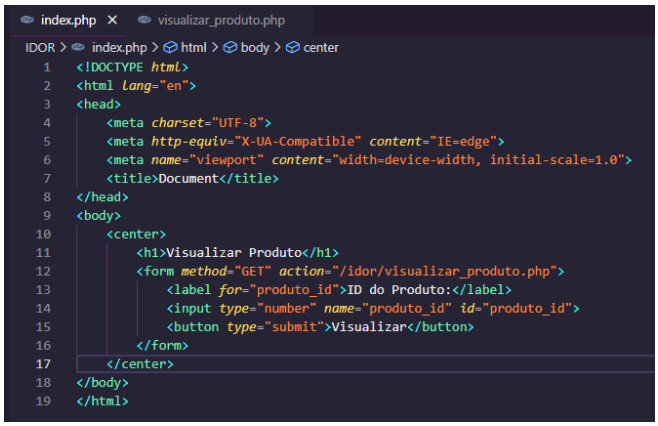
Fonte: Os autores (2023)

A consulta SQL corrigida usa *prepared statements* com marcadores :username e :password. O método execute() conecta esses valores de forma segura, evitando código SQL malicioso. Essa estratégia separada de processamento de valores impede a injeção de SQL indesejado, tornando a consulta mais segura e recomendada. A OWASP descreve esse processo como Codificação de Dados de Entrada.

4.3 IDOR (INSECURE DIRECT OBJECT REFERENCE)

A vulnerabilidade IDOR ocorre quando uma aplicação permite acesso direto a objetos sem verificar permissões adequadas. Isso ocorre devido a falhas no controle de acesso. Um usuário mal-intencionado pode manipular identificadores para acessar recursos protegidos, comprometendo a autenticidade, confidencialidade e disponibilidade dos dados. A gravidade varia de acordo com o contexto da aplicação.Figura 23 temos um código feito em HTML e PHP, vulnerável à IDOR:

Figura 23: Código HTML e PHP



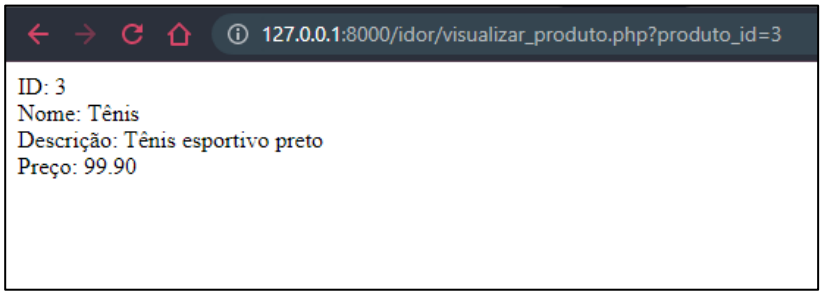
Fonte: Os autores (2023)

Figura 24: Código PHP

Fonte: Os autores (2023)

Ele gera uma página de busca de produtos, onde é inserido o ID do produto e o mesmo é retornado, se estiver cadastrado no banco de dados. Como apresentado na Figura 25:

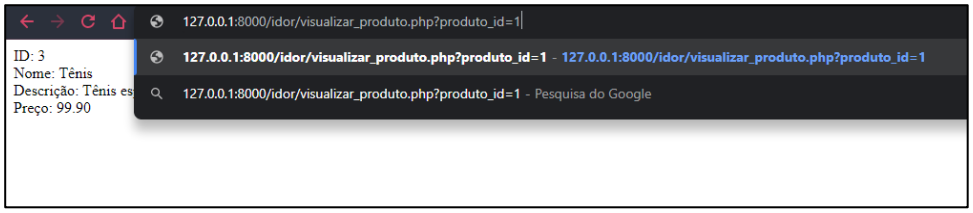
Figura 25: Mensagem de retorno dos dados



Fonte: Os autores (2023)

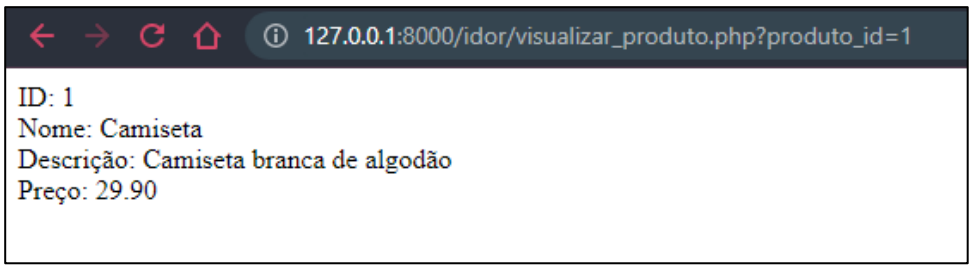
Caso valor do parâmetro “produto_id” seja alterado na URL, é retornado o produto do ID passado, sem realizar nenhum tipo de validação. Apresentado na Figura 26 e Figura 27:

Figura 26: Mensagem de retorno dos dados



Fonte: Os autores (2023)

Figura 27: Mensagem de retorno dos dados



Fonte: Os autores (2023)

Para corrigir essa vulnerabilidade, é necessário adicionar camadas de validação e controle de acesso aos recursos da aplicação. Como existem vários contextos dessa falha, aqui está uma lista de boas práticas para corrigi-la:

- Implementar controle de acesso adequado e gerenciamento de sessão
    - A vulnerabilidade IDOR (Insecure Direct Object Reference) ocorre quando identificadores inseguros são usados para acessar recursos protegidos, devido a verificações insuficientes de permissões. Para evitar essa falha, é necessário implementar controles de acesso adequados, validar as permissões do usuário e seguir as práticas recomendadas pela OWASP para garantir a segurança contra IDOR e outras vulnerabilidades.
- Evitar referências diretas a objetos
    - Utilizar referências diretas de objetos em um aplicativo é considerada insegura, especialmente quando se trata de dados confidenciais. Isso pode levar a vulnerabilidades IDOR. Nesse caso, é recomendado o uso de referências indiretas, como mapas ou hashes, para ocultar o identificador verdadeiro do objeto. Se optar por usar hashes, é importante incluir um sal forte e exclusivo

para aumentar a segurança, pois algoritmos básicos de hash podem ser facilmente revertidos. Além disso, é fundamental resolver problemas de controle de acesso para garantir a segurança geral do aplicativo.

- Utilizar GUIDs ou identificadores aleatórios
    - A vulnerabilidade de referência direta de objeto ocorre quando aplicativos usam identificadores sequenciais ou iterativos, permitindo a previsibilidade dos valores. Isso possibilita a enumeração e busca exaustiva até encontrar o objeto desejado usando técnicas automatizadas e poder computacional.
- Validar a entrada do usuário
    - A validação rigorosa dos parâmetros do usuário é eficaz para mitigar problemas de segurança, incluindo a vulnerabilidade IDOR. Ao validar o comprimento e formato dos parâmetros, dificulta-se a enumeração de identificadores. Essa validação pode ser feita no lado do cliente ou do servidor, dependendo das necessidades do sistema.

4.4 **ANALISE DOS RESULTADOS**

Os métodos utilizados revelaram falhas de segurança comuns em aplicações web, destacando a importância de práticas de desenvolvimento seguro desde as fases iniciais do projeto. Isso ajuda a prevenir vulnerabilidades que podem ser exploradas por atacantes.

Durante o desenvolvimento de sites e aplicações web, é essencial aplicar práticas como validar a entrada do usuário, verificar autenticação e estabelecer níveis de permissões. Esses cuidados devem ser considerados em todas as etapas do processo. As correções fortaleceram a segurança da aplicação, reduzindo falhas exploráveis. No entanto, a segurança é um processo contínuo e novas ameaças podem surgir. Avaliações periódicas e atualizações constantes são necessárias para garantir a proteção adequada da aplicação.

É crucial conscientizar e capacitar os desenvolvedores dessa aplicação para que possam criar códigos mais seguros e evitar possíveis vulnerabilidades.

5 CONSIDERAÇÕES FINAIS

Foi destacada a importância da segurança da informação e do desenvolvimento seguro em aplicações web. Em um cenário cada vez mais digital e interconectado, a proteção dos dados e a garantia da integridade das aplicações se tornam fundamentais para as

organizações. A segurança da informação abrange um conjunto de medidas e práticas que visam controlar riscos, prevenir roubo, danos e perdas de dados sensíveis. No contexto das aplicações web, o desenvolvimento seguro desempenha um papel crucial na proteção contra ameaças cibernéticas. Ao implementar medidas de segurança desde o início do processo de desenvolvimento, é possível mitigar vulnerabilidades e reduzir significativamente os riscos de ataques maliciosos.

Essa abordagem pró-ativa contribui para a preservação dos dados, a manutenção da reputação e o fortalecimento da confiança de parceiros e clientes. Ao optar por um desenvolvimento seguro de aplicações web, as organizações obtêm benefícios duradouros. Além de proteger os ativos digitais, também garantem conformidade com regulamentações e leis de proteção de dados, evitando penalidades legais e prejuízos financeiros. Além disso, a adoção de boas práticas de segurança desde o início do projeto proporciona uma base sólida, reduzindo a necessidade de correções e retrabalhos no futuro.

Nesse contexto, a OWASP (Open Web Application Security Project) desempenha um papel fundamental ao fornecer diretrizes e recomendações amplamente reconhecidas no desenvolvimento seguro de aplicações web. Seu guia, conhecido como "OWASP Top 10", identifica as principais vulnerabilidades comuns encontradas em aplicações web e fornece orientações sobre como mitigar essas ameaças.

Dentre as medidas de segurança essenciais no desenvolvimento seguro de aplicações web, destacam-se a autenticação adequada dos usuários, a correta autorização de acesso a recursos, a criptografia de dados sensíveis e a validação rigorosa de todas as entradas de dados. Além disso, é importante realizar testes de segurança regulares, como a análise estática de código, a verificação de vulnerabilidades e a realização de testes de invasão.

Conclui-se que, o desenvolvimento seguro de aplicações web é indispensável para garantir a proteção dos dados, a segurança dos usuários e a confiança dos clientes. Ao seguir as diretrizes da OWASP e adotar práticas de segurança desde a concepção do projeto até a sua manutenção, as organizações podem estar preparadas para enfrentar os desafios atuais e futuros no ambiente cibernético, protegendo-se contra ameaças e promovendo a segurança da informação de forma efetiva.

REFERÊNCIAS

A importância do Desenvolvimento Seguro nas empresas. Cecyber, 2022. Disponível em: <https://cecyber.com/a-importancia-do-desenvolvimento-seguro-nas-empresas/>. Acesso em: 29 de outubro de 2022.

A Importância Do Desenvolvimento Seguro. Por Andrea Rosti, 2020. Disponível em: <https://safewayconsultoria.com/a-importancia-do-desenvolvimento-seguro/>.
About the OWASP Foundation. OWASP Foundation, 2022. Disponível em: <https://owasp.org/about/>. Acesso em: 20 de outubro de 2022.

Grand Theft Auto 6 leak: who hacked Rockstar and what was stolen? MACDONALD, Keza; STUART, Keith; HERN, Alex. The Guardian, 19 de setembro de 2022. Disponível em: <https://www.theguardian.com/games/2022/sep/19/grand-theft-auto-6-leak-who-hacked-rockstar-and-what-was-stolen>. Acesso em: 29 de outubro de 2022.

O que é e por que é tão importante o desenvolvimento seguro? Nova8, 2021. Disponível em: <https://www.nova8.com.br/2021/10/o-que-e-e-por-que-e-tao-importante-o-desenvolvimento-seguro/>. Acesso em: 29 de outubro de 2021.

PEDRA, David. Segurança da informação: o que é e como criar uma política para proteção de dados, 2023. Disponível em: <https://www.siteware.com.br/seguranca/seguranca-da-informacao/>. Acesso em: 29 de maio de 2023.

SCHENDES, William. Uber sofre novo ataque hacker; dados de 77 mil funcionários são comprometidos. Disponível em: <https://olhardigital.com.br/2022/12/14/seguranca/uber-sofre-novo-ataque-hacker-dados-de-77-mil-funcionarios-sao-comprometidos/>. Acesso em: 29 de maio de 2023.

Segurança da Informação: conceito, função e como proteger dados sigilosos, DocuSign, 2022. Disponível em: <https://www.docusign.com.br/blog/seguranca-da-informacao>. Acesso em: 29 de maio de 2023.