SEGURANÇA EM BANCO DE DADOS: ESTRATÉGIAS DE PREVENÇÃO E MITIGAÇÃO DE INJEÇÃO SQL.

JOÃO PEDRO GEROTTO FERNANDES[1]
MARIÂNGELA FERREIRA FUENTES MOLINA[2]

RESUMO

Com o crescente uso de bancos de dados relacionais, a preocupação com a segurança das aplicações web também aumenta, sendo os ataques de injeção SQL uma das ameaças mais comuns. Esses ataques exploram falhas nos sistemas que muitas vezes passam despercebidas pelos desenvolvedores, não por falta de conhecimento técnico, mas pela necessidade de uma compreensão mais profunda do funcionamento interno dos Sistemas de Gerenciamento de Bancos de Dados (SGBDs).
Este trabalho apresentou algumas das principais técnicas de injeção SQL, ilustrando como essas vulnerabilidades podem ser exploradas. Além disso, foram analisadas estratégias de mitigação, como o uso de Views, Prepared Statements e Stored Procedures, que se mostraram eficazes para reduzir os riscos desses ataques. Também foram exploradas ferramentas e boas práticas para prevenir e minimizar essas ameaças.

Palavras-chave: Banco de Dados; Código; Método; SQL; SQL Injection; Técnica.

ABSTRACT

With the increasing use of relational databases, concerns about web application security are also growing, with SQL injection attacks being one of the most common threats. These attacks exploit system vulnerabilities that often go unnoticed by developers, not due to a lack of technical knowledge but because of the need for a deeper understanding of how Database Management Systems (DBMS) work.
This study presented some of the main SQL injection techniques, illustrating how these vulnerabilities can be exploited. Additionally, mitigation strategies such as the use of Views, Prepared Statements, and Stored Procedures were analyzed, proving to be effective in reducing the risks of such attacks. Furthermore, tools and best practices were explored to prevent and minimize these threats.

Keywords: Database; Code; Method; SQL; SQL Injection; Technique.

[1]Graduando em Análise e Desenvolvimento de Sistemas pela Faculdade de Tecnologia de Mogi das Cruzes – FATEC MC. Mogi das Cruzes – SP. E-mail: joao.fernandes33@fatec.sp.gov.br
[2]Docente, Faculdade de Tecnologia de Mogi das Cruzes – FATEC MC. Mogi das Cruzes – SP

INTRODUÇÃO

Com o avanço das tecnologias e a crescente digitalização dos dados, os sistemas gerenciadores de banco de dados tornaram-se alvos valiosos para ataques cibernéticos. O valor dos dados armazenados por empresas atrai a atenção de hackers, que buscam acessar, alterar ou roubar informações sensíveis. Entre os ataques mais comuns, destaca-se a injeção SQL, que permite a execução de comandos maliciosos em um banco de dados, gerando danos como roubo de dados e indisponibilidade de sistemas.

O interesse em realizar esta pesquisa surge da constatação de que muitas empresas ainda não conhecem ou não adotam medidas eficazes para prevenir ataques de injeção SQL. A sofisticação crescente dos cibercriminosos torna essencial entender como esses ataques ocorrem e como proteger adequadamente os bancos de dados. Este estudo visa fornecer conhecimento sobre a segurança para as vulnerabilidades associadas a esses ataques, a fim de mitigar prevenir maiores riscos.

A metodologia adotada nesta pesquisa combina uma análise teórica com uma demonstração prática das vulnerabilidades. O estudo se concentra em entender as técnicas utilizadas por hackers para realizar injeção SQL. Será demonstrado técnicas e soluções de mitigação, como *prepared statements e stored procedures*, visando a prevenção desses ataques.

O objetivo principal deste trabalho é investigar as vulnerabilidades de segurança associadas à injeção SQL e propor medidas para proteger bancos de dados contra esses ataques. A pesquisa buscará fornecer soluções aplicáveis que as empresas possam adotar para proteger seus dados sensíveis.

MATERIAL E MÉTODO

A pesquisa foi conduzida em duas etapas principais. Primeiramente, foi realizada uma pesquisa bibliográfica e exploratória para compreender o

funcionamento dos SGBDRs, analisando como as consultas são executadas e identificando estratégias para mitigar ataques de injeção SQL.

Em seguida, foi feita uma análise aprofundada de diferentes métodos e técnicas de proteção contra SQL Injection, avaliando sua eficácia e aplicabilidade no contexto de segurança de bancos de dados. Com base na análise, foram selecionadas técnicas mais adequadas para implementação, visando demonstrar o seu funcionamento e aplicação na prevenção desse tipo de ataque.

As fontes de pesquisa incluem livros sobre bancos de dados, SGBDs, SQL e segurança da informação, além de artigos acadêmicos sobre segurança de dados. Também foram utilizadas pesquisas online por meio de ferramentas como Google e Google Acadêmico, bem como bibliotecas físicas e digitais, tanto institucionais quanto públicas.

REFERENCIAL TEÓRICO

Linguagem SQL

A linguagem SQL é a principal linguagem de comunicação com o banco de dados, isto é, consultas, alterações, exclusões, e inserção dos dados presentes. Segundo NAVATHE (2001, p.57) “A linguagem SQL pode ser considerada um dos principais motivos para o sucesso de banco de dados relacionais comerciais.”

Por ser uma linguagem forte, ela foi divida em quatro grupos de tipo de comando, sendo DDL (*Data Definition Language*), DML(*Data Manipulation Language*), DQL (*Data Query Language*) e DCL (*Data Control Language*).

Conexão com o Banco de Dados

A conexão entre uma aplicação e um banco de dados é estabelecida por meio de drivers e bibliotecas específicas para cada tecnologia, utilizando protocolos como ODBC (Open Database Connectivity) ou JDBC (Java Database Connectivity), dependendo da linguagem de programação empregada.

Em sistemas de grande escala, é comum o uso de um pool de conexões, que gerencia múltiplas requisições simultaneamente para otimizar o desempenho e reduzir a sobrecarga no banco.

Se a aplicação se conecta ao banco de dados utilizando uma conta com permissões excessivas, um invasor pode explorar falhas no código para executar comandos não autorizados, comprometendo a integridade dos dados.

Injeção SQL

Injeção SQL, encontrado pelo termo em inglês *SQL Injection,* é um método de ataque no qual tem alvo os bancos de dados de empresas no qual este ataque pode ser feito por meio de código SQL mal-intencionado.

Esse ataque é bastante comum e também um dos mais perigosos pois pode ser utilizado em qualquer aplicação que dependa de formulário, isso é, sua grande maioria. O ataque acontece por meio das entradas de usuário, onde se espera dado inofensivos, acabam encontrando código SQL.

Existe três tipos de injeção SQL, sendo uma delas injeção SQL In-Band no qual o atacante usa o mesmo canal de comunicação da aplicação e o banco de dados. Outro tipo é a Injeção SQL de Inferência onde o *hacker* não ve diretamente o resultado de seu ataque e sim faz deduções sobre o que seu ataque causou e por último temos Injeção SQL Out-Of-Band que usa um canal diferente de comunição entra a aplicação e o banco de dados, onde geralmente depende de alguns recursos ativos no servidor.

Um dos ataques mais conhecido foi a Sony Pictures em 2011, no qual os atacantes tiveram acesso a mais de 1 milhão de clientes, 75 mil códigos de músicas e 3,5 milhões de cupons de desconto da plataforma.

Para a proteção contra *SQL Injection* existe alguns métodos que tem essa função. Abaixo será listado algumas técnicas para mitigação dos riscos deste tipo de ataque.

Prepared Statements

Prepared Statements é uma técnica utilizada para otimizar, simplificar e reforçar a segurança contra ataques de injeção SQL (SQLi). Essa abordagem pode ser implementada tanto no Back-End quanto diretamente no Banco de Dados, oferecendo maior confiabilidade na execução de consultas.

O método consiste na pré-compilação de consultas pelo banco de dados, permitindo que elas sejam armazenadas e reutilizadas sempre que necessário. Essas consultas podem ser parametrizadas ou não, proporcionando eficiência na execução ao evitar a necessidade de recompilação a cada requisição.

O tempo de vida útil de um Prepared Statement varia conforme o local de implementação. Quando armazenado no banco de dados, ele permanece disponível enquanto o servidor estiver em operação ou até que entre em modo StandBy. Já no Back-End, a persistência está limitada ao ciclo de vida de uma requisição (request), sendo descartado após sua conclusão.

Uma das principais vantagens desse método é que os dados fornecidos pelos usuários são sempre tratados exclusivamente como valores, sem possibilidade de interferência na estrutura da consulta SQL. Dessa forma, a utilização de Prepared Statements elimina a possibilidade de execução de comandos maliciosos inseridos em entradas manipuladas, tornando-se uma solução eficaz para a mitigação de ataques por injeção de código.

Stored Procedures

Conhecidas simplesmente como Procedures, as Stored Procedures representam uma forma avançada de programação dentro do SQL, permitindo a criação de estruturas de controle, como seleções e repetições, além da recepção de parâmetros fortemente tipados. Essa abordagem possibilita o encapsulamento de operações no banco de dados, promovendo maior organização e segurança no acesso aos dados.

As Stored Procedures são amplamente utilizadas em linguagens específicas de gerenciamento de banco de dados, como PL/SQL (Oracle Database) e T-SQL (SQL Server). Sua compilação e armazenamento compartilham semelhanças com os Prepared Statements, mas diferem quanto à sua persistência: enquanto os Prepared Statements têm um ciclo de vida limitado, as Stored Procedures permanecem ativas no banco de dados até sua exclusão explícita. Além disso, as procedures seguem o mesmo tratamento literal dos dados, garantindo maior segurança contra injeção de código malicioso.

Funcionalmente, as Stored Procedures apresentam semelhanças com funções em linguagens de programação estruturadas ou com métodos em linguagens orientadas a objetos. Sua utilização contribui para a modularização do código, promovendo reutilização e eficiência no processamento de transações dentro do banco de dados.

Princípio do privilégio minimo

O Princípio do Privilégio Mínimo estabelece que um usuário deve ter acesso apenas aos recursos necessários para desempenhar suas funções, sem permissões adicionais que possam comprometer a segurança do sistema. Conforme destacado pela Cloudflare (2024) quanto mais um determinado usuário tiver acesso, maior será o impacto negativo se sua conta for comprometida ou se ele se tornar uma ameaça interna.

Os Sistemas Gerenciadores de Banco de Dados Relacionais (RDBMS) modernos oferecem suporte à criação de usuários e à gestão de permissões por meio dos comandos GRANT e REVOKE, permitindo a atribuição e a revogação de privilégios de forma granular. Essa abordagem possibilita um controle mais rigoroso sobre as operações que cada usuário pode executar, reduzindo riscos de acessos indevidos ou manipulação não autorizada de dados.

Embora o conceito seja relativamente simples, sua implementação ainda é negligenciada por muitas organizações, seja por falta de conscientização sobre sua

importância, seja pela ausência de profissionais especializados. A adoção do Princípio do Menor Privilégio é uma estratégia essencial para mitigar vulnerabilidades e minimizar os impactos de possíveis incidentes de segurança.

Higienização das entradas

A validação e o tratamento adequado dos dados fornecidos pelo usuário são etapas essenciais para garantir a segurança de uma aplicação. Segundo a Cloudflare (2024), todas as entradas devem ser monitoradas e validadas continuamente para mitigar e eliminar vulnerabilidades, prevenindo ataques bem-sucedidos e o vazamento de dados armazenados.

Uma das principais técnicas de proteção envolve a filtragem e sanitização dos dados de entrada, removendo caracteres especiais potencialmente maliciosos, como ponto e vírgula (;), aspa simples ('), cerquilha (#), entre outros. No entanto, esse processo muitas vezes é negligenciado por desenvolvedores, seja pela falta de familiaridade com manipulação de strings, seja por descuidos na implementação de medidas de segurança adequadas.

A sanitização das entradas é mais eficiente quando realizada no Back-End, pois essa abordagem permite um controle mais rigoroso sobre a lógica de negócios, além de oferecer maior flexibilidade na aplicação de regras de validação e processamento. Dessa forma, torna-se possível mitigar riscos relacionados a ataques como SQL Injection, Cross-Site Scripting (XSS) e outras explorações baseadas na manipulação de dados de entrada.

Tabelas Hash

As Tabelas Hash são estruturas de dados amplamente utilizadas para otimizar o processo de busca e recuperação de informações. Essa estrutura adota um modelo chave-valor, onde cada chave corresponde a um valor específico, sendo conceitualmente semelhante às relações utilizadas em bancos de dados relacionais.

Linguagens modernas como Java, C# e Python já oferecem suporte nativo a essa estrutura, o que facilita sua implementação.

No contexto da segurança da informação, as Tabelas Hash podem ser empregadas como uma técnica para reforçar a proteção contra ataques baseados na manipulação de comandos SQL. Nessa abordagem, cria-se um "dicionário" de comandos SQL, no qual as chaves representam os diferentes tipos de comandos SQL (como *SELECT, INSERT, UPDATE e DELETE*), e os valores são listas que armazenam as respectivas instruções associadas a cada categoria.

O processo de validação dos dados fornecidos pelo usuário envolve duas iterações consecutivas: a primeira percorre os diferentes tipos de comandos SQL armazenados na tabela hash, enquanto a segunda verifica se os dados de entrada contêm comandos correspondentes a cada categoria. Caso uma correspondência seja identificada, o comando é removido dos dados informados, garantindo um nível adicional de sanitização e mitigando potenciais ataques.

Embora essa técnica não substitua abordagens mais robustas, ela adiciona uma camada extra de segurança ao sistema. Sua implementação pode ser particularmente útil como um mecanismo complementar de filtragem antes da aplicação de outras medidas de proteção.

View

As Views são amplamente utilizadas no contexto de bancos de dados para simplificação de consultas e organização da apresentação de dados. Elas são baseadas em consultas do tipo projeção, utilizando o comando SELECT para exibir informações de forma estruturada, sem armazená-las fisicamente. Devido a essa característica, as Views são frequentemente denominadas "tabelas virtuais".

Cada View é construída a partir de uma ou mais tabelas de referência, podendo combinar dados de diferentes fontes em uma única exibição. Ao contrário das tabelas convencionais, as Views não armazenam os dados diretamente; a cada

consulta, os dados são extraídos das tabelas subjacentes, garantindo que as informações estejam sempre atualizadas.

No contexto da segurança contra SQL Injection, as Views desempenham um papel relevante ao permitir a restrição dos dados acessíveis por determinados usuários. Por meio dessa técnica, é possível encapsular e limitar a exposição de informações sensíveis, ocultando detalhes da estrutura do banco de dados e minimizando a superfície de ataque. Dessa forma, as Views contribuem para a implementação do princípio do menor privilégio, reduzindo o risco de exploração por usuários mal-intencionados.

A importância da segurança para os Banco de Dados

A segurança da informação sempre foi um tema crítico, especialmente diante de ataques como a injeção SQL. A proteção contra essas ameaças é essencial, pois os usuários confiam na aplicação para armazenar suas informações confidenciais, o que impõe uma grande responsabilidade aos desenvolvedores.

A integridade, confidencialidade e disponibilidade dos dados são aspectos fundamentais, e qualquer falha pode resultar em consequências graves, como perdas financeiras e danos à reputação. No Brasil, a Lei Geral de Proteção de Dados (LGPD) reforça essa importância, exigindo medidas rigorosas para a proteção de informações sensíveis, como CPF e nome de pessoas físicas.

Dessa forma, prevenir acessos não autorizados e mitigar os riscos associados a esses ataques deve ser uma prioridade em qualquer sistema que manipule dados sensíveis.

RESULTADO E DISCUSSÃO

A partir dos estudos realizados, verificou-se uma carência significativa de documentação detalhada sobre a injeção SQL, abrangendo tanto os mecanismos desse tipo de ataque quanto as estratégias utilizadas pelos atacantes. Essa lacuna

dificulta a adoção de práticas preventivas eficazes por parte dos desenvolvedores e administradores de banco de dados, tornando os sistemas mais vulneráveis.

A análise do estudo de caso evidenciou que a injeção SQL continua sendo uma ameaça recorrente, especialmente em sistemas que não implementam mecanismos de validação e sanitização adequados. Durante a pesquisa, observou-se que muitas aplicações não realizam o tratamento adequado das entradas fornecidas pelos usuários, permitindo a manipulação de consultas SQL e, consequentemente, o acesso indevido a dados armazenados.

Além disso, foi identificado que a várias técnicas são frequentemente negligenciadas, seja por desconhecimento técnico ou pela falta de atenção às boas práticas de segurança. Essa ausência de tratamento adequado favorece a exploração de falhas e reforça a necessidade de implementar métodos robustos de proteção, como a parametrização de consultas e o uso de Prepared Statements.

Outro aspecto relevante é a predominância de abordagens empíricas no desenvolvimento de aplicações, sem a devida consideração das vulnerabilidades de segurança. Isso reforça a importância de incluir diretrizes de proteção contra injeção SQL nos processos de desenvolvimento e manutenção de software, garantindo que os sistemas sejam projetados com mecanismos que minimizem os riscos associados a esse tipo de ataque.

Dessa forma, os achados desta pesquisa corroboram estudos anteriores ao destacar que a ausência de validação de entradas e a falta de conscientização sobre segurança da informação são fatores determinantes para a persistência da injeção SQL como uma das principais vulnerabilidades exploradas na web.

CONCLUSÃO

Os ataques de injeção SQL permanecem como uma das principais ameaças aos sistemas de bancos de dados, explorando falhas na manipulação de entradas para acessar, modificar ou excluir informações sensíveis. A pesquisa evidencia que

muitas empresas e desenvolvedores ainda falham na implementação de medidas de proteção, muitas vezes por desconhecimento técnico ou por priorizar a entrega rápida de software em detrimento da segurança.

Para mitigar esses riscos, foram analisadas diversas técnicas de proteção, incluindo *Views, Prepared Statements e Stored Procedures*, que ajudam a restringir o acesso e evitar a manipulação indevida das consultas. Além disso, a validação de entradas e o uso de consultas parametrizadas demonstraram ser estratégias eficazes na prevenção de ataques.

No entanto, a proteção absoluta contra injeção SQL ainda é um desafio, exigindo um esforço contínuo para acompanhar novas técnicas de ataque e reforçar a segurança. Diante de normas como ISO 27001 e LGPD, torna-se essencial que empresas invistam na capacitação de seus profissionais e na implementação de controles rigorosos para garantir a proteção dos dados.

REFERÊNCIAS BIBLIOGRÁFICAS

CLOUDFLARE. **Princípio do Menor Privilégio. Cloudflare**, 2024. Disponível em: https://www.cloudflare.com/pt-br/learning/access-management/principle-of-least-privilege/. Acesso em: 13 out. 2024.

FARIAS, M. B. de. **Injeção de SQL em Aplicações Web: Causas e Prevenção**. 2009. 38 folhas. Área de Concentração: Segurança da Informação - Universidade Federal do Rio Grande do Sul, Porto Alegre, 2009.

IMASTERS. **Procedures como Defesa à SQL Injection**. iMasters, 2019. Disponível em: https://imasters.com.br/sql-server/procedures-como-defesa-a-sql-injection. Acesso em: 08 out. 2024.

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, **ISO/IEC. 27001:2022 – Information technology – Security techniques – Information security management systems – Requirements.** Genebra: International Organization for Standardization, 2022.

MICROSOFT. **SQL Injection**. Microsoft, 2024. Disponível em: https://learn.microsoft.com/pt-br/sql/relational-databases/security/sql-injection?view=sql-server-ver16. Acesso em: 04 ago. 2024.

NAVATHE, S. B. **Sistemas de Banco de Dados**. 6ª edição. São Paulo: Pearson, 2010.

NORDVPN. **O que é injeção SQL**? NordVPN, 2023. Disponível em: https://nordvpn.com/pt-br/blog/o-que-e-injecao-sql/. Acesso em: 15 ago. 2024.

ORACLE. **O que é banco de dados?** Oracle, 2020. Disponível em: https://www.oracle.com/br/database/what-is-database/. Acesso em: 20 mar. 2024.

SECURITYFIRST. **Hackers roubam dados de 2 milhões em injeção de SQL e ataques XSS**. SecurityFirst, 2024. Disponível em: https://securityfirst.com.br/hackers-roubam-dados-de-2-milhoes-em-injecao-de-sql-e-ataques-xss/. Acesso em: 21 mar. 2024.