NoSQL e Segurança: Um estudo de análise para prevenção de injeção em bancos de dados NoSQL

Kassem Ubinski Awad
Centro Universitário Dinâmica das Cataratas (UDC)
Foz do Iguaçu, Brasil
kassemtop@gmail.com

Luciano Santos Cardoso
Centro Universitário Dinâmica das Cataratas (UDC)
Foz do Iguaçu, Brasil
luciano.cardoso@udc.edu.br

Alessandra Bussador
Centro Universitário Dinâmica das Cataratas (UDC)
Foz do Iguaçu, Brasil
alessandra@udc.edu.br

***Abstract*—The analysis of security vulnerabilities in NoSQL databases, with a focus on code injection attacks, reveals the challenges faced by systems widely adopted in web and cloud applications. The exponential growth of NoSQL databases, driven by their ability to handle large volumes of unstructured data and horizontal scalability, contrasts with the robustness and transactional integrity of relational databases (SQL). Despite the flexibility and efficiency offered by NoSQL systems, they present new security threats, particularly related to injection attacks, which can compromise the integrity and confidentiality of the stored data. A detailed comparison between SQL and NoSQL architectures highlights the main vulnerabilities associated with each, mapping existing methods and tools for the prevention and mitigation of attacks in NoSQL systems. Attack simulations in a controlled environment replicate real-world usage scenarios, including techniques such as tautologies, Piggyback Queries, and UNION attacks, to test vulnerabilities and evaluate the effectiveness of defense strategies. Input validation through Deterministic Finite Automata (DFA) emerges as an effective approach to prevent injection attempts before they can impact the database. RSA encryption is explored as an additional layer of protection for sensitive data, reinforcing the security of NoSQL systems against attacks. The results demonstrate that, while NoSQL injections represent a significant threat, the application of advanced mitigation techniques, such as DFAbased input validation and RSA encryption, can substantially reduce the associated risks. The research provides practical and theoretical insights to strengthen the security of NoSQL systems in various application contexts.**

***Keywords*—NoSQL; Security; Injection.**

Resumo — ***A análise de vulnerabilidades de segurança em bancos de dados NoSQL, com ênfase nos ataques de injeção de código, revela os desafios enfrentados por sistemas amplamente adotados em aplicações web e na nuvem. O crescimento exponencial dos bancos de dados NoSQL, impulsionado por sua capacidade de lidar com grandes volumes de dados não estruturados e escalabilidade horizontal, contrasta com a robustez e integridade transacional dos bancos de dados relacionais (SQL). Apesar da flexibilidade e da eficiência oferecidas pelos sistemas NoSQL, eles apresentam novas ameaças de segurança, especialmente relacionadas a ataques de injeção, que podem comprometer a integridade e a confidencialidade dos dados. A comparação detalhada entre as arquiteturas SQL e NoSQL destaca as principais vulnerabilidades associadas a cada uma, mapeando métodos e ferramentas existentes para a prevenção e mitigação de ataques em sistemas NoSQL. Simulações de ataques em um ambiente controlado replicam cenários reais de uso, incluindo técnicas como tautologias, Piggyback Queries, e ataques UNION, para testar as vulnerabilidades e avaliar a eficácia das estratégias de defesa. A validação de entradas por meio de Autômatos Finitos Determinísticos (DFA) surge como uma abordagem eficaz para prevenir tentativas de injeção antes que possam impactar o banco de dados. A criptografia RSA é explorada como uma camada adicional de proteção para dados sensíveis, reforçando a segurança de sistemas NoSQL contra ataques.***

***Palavras-chave*—NoSQL; Segurança; Injeção.**

I. Introdução

Durante mais de uma década, os bancos de dados NoSQL têm crescido exponencialmente. A despeito desse crescimento, os bancos de dados relacionais ainda mantêm sua relevância. Os modelos tradicionais impõem uma estrutura de esquema rígido, o que dificulta a escalabilidade e inibe a modificação de dados em ambientes diversos. Por outro lado, os bancos de dados NoSQL suportam protótipos simples [1]. Em 1998, Carlo Strozzi foi o pioneiro ao utilizar a sigla NoSQL ao nomear seu banco de dados "relacional" leve e de código aberto que não utilizava SQL. Com o NoSQL, é possível desenvolver rapidamente uma aplicação com alto desempenho e escalabilidade. Sistemas NoSQL lidam tanto com dados estruturados quanto não estruturados, conseguindo processar rapidamente grandes volumes de dados não estruturados (*BigData*). Por isso, tornou-se popular em aplicações em larga escala na nuvem e na web, como Google, Facebook, Adobe, eBay, Cisco, etc. Ambos os bancos de dados SQL e NoSQL continuam sujeitos a ataques de injeção até os dias atuais, As análises de segurança mostram que cerca de 23% das vulnerabilidades críticas encontradas em 2023 estavam

relacionadas a injeções, o que inclui tanto SQL quanto NoSQL. Este dado reflete a importância da sanitização de entradas e da proteção contra esses tipos de ataques [2]. Em 2012, hackers conseguiram roubar mais de 450.000 credenciais de login do Yahoo explorando uma injeção SQL em seus servidores. Com o aumento da popularidade dos bancos de dados NoSQL, as ameaças à segurança também aumentaram [3]. O crescimento exponencial dos bancos de dados NoSQL trouxe uma nova era de soluções tecnológicas voltadas para o armazenamento e processamento de grandes volumes de dados não estruturados. Apesar das vantagens em termos de flexibilidade, escalabilidade e velocidade, esses sistemas também são acompanhados de um aumento nos desafios de segurança, principalmente no que diz respeito a injeções de código, que comprometem a integridade e a confidencialidade dos dados armazenados. Diante deste cenário, emergem questões relacionadas à segurança de sistemas NoSQL: como as injeções de código afetam a integridade desses bancos de dados, quais são as suas variantes mais comuns, e quais estratégias podem ser desenvolvidas para prevenir e mitigar essas ameaças. A relevância dessas questões se destaca ainda mais à luz do crescente número de vulnerabilidades críticas associadas a injeções relatadas em 2023, as quais representaram cerca de 23% das falhas de segurança críticas identificadas. O estudo de casos como o ataque de injeção que comprometeu mais de 450.000 credenciais de login no Yahoo reforça a necessidade de estratégias de defesa robustas para proteger dados sensíveis em sistemas NoSQL. Portanto, o objetivo desta pesquisa é analisar a injeção de código em bancos de dados NoSQL, investigando seus mecanismos, impactos na segurança da informação e as melhores práticas para mitigação. Em suma, este estudo pretende contribuir para um ambiente mais seguro na utilização de bancos de dados NoSQL, por meio da implementação de técnicas de defesa que assegurem a integridade e confidencialidade dos dados, promovendo um avanço nas práticas de segurança cibernética em aplicações modernas.

II. REFERENCIAL TEÓRICO

A. *Bancos de Dados NoSQL*

Os bancos de dados NoSQL são sistemas gerais de banco de dados distribuídos, que podem não requerer dados estruturados, são tipicamente projetados para escalar horizontalmente, e podem ser *open source*. O NoSQL apresenta a capacidade de armazenar, gerenciar e indexar conjuntos de dados arbitrariamente grandes, ao mesmo tempo em que oferece suporte a um grande número de solicitações de usuários simultâneos [4]. O advento dos sistemas NoSQL oferecem soluções para os problemas que lidam com dados enormes, em contraste com os sistemas de gerenciamento de bancos de dados relacionais, os sistemas NoSQL oferecem um paradigma diferente para armazenar dados, lidando eficientemente com dados não estruturados como multimídia, e-mails, documentos, etc. As características gerais dos sistemas NoSQL podem ser resumidas como alta escalabilidade, confiabilidade e modelo de dados muito simples [5]. Entre todos os 340 sistemas de banco de dados, o MongoDB é nomeado como o banco de dados NoSQL mais popular de acordo com o rastreamento do DB-Engines que é uma plataforma que fornece uma visão abrangente e atualizada sobre a popularidade e uso de sistemas de gerenciamento de banco de dados. O MongoDB é um banco de dados NoSQL de código aberto que é escrito em C++. É um banco de dados baseado em documentos. O documento é chamado de BSON (JSON Binário), que é semelhante ao formato JSON. O MongoDB armazena esses documentos formatados em BSON dentro de uma coleção. Ele utiliza o paradigma de processamento de dados Mapreduce para condensar grandes quantidades de dados em resultados agregados úteis. Map-reduce é como uma operação de agrupamento em SQL [3]. Ao contrário dos bancos de dados relacionais, não há um esquema tipificado estaticamente no MongoDB, razão pela qual é chamado de sem esquema. Assim, cada documento em uma coleção pode possuir atributos diferentes. Embora o MongoDB não suporte consultas SQL tradicionais como o MySQL, ele possui consulta de documentos por meio da qual podemos executar SQL para encontrar dados menores ou maiores que um valor específico ou usar expressões regulares para correspondência de padrões. O MongoDB pode ser escalado dentro e entre vários centros de dados distribuídos com bom desempenho, o que o torna mais preferível em relação a outros bancos de dados relacionais como o MySQL [3].

B. *Bancos de Dados SQL*

Os bancos de dados SQL (*Structured Query Language*), estabelecidos no modelo relacional proposto por Edgar Codd, são amplamente utilizados devido à sua eficiência em gerenciar dados relacionais. Os sistemas de gerenciamento de bancos de dados relacionais integraram funcionalidades de serviço e a linguagem descritiva SQL para descrição, seleção e manipulação de dados [8]. Os bancos de dados SQL são conhecidos por suas características robustas de integridade de dados, processamento de transações ACID (Atomicidade, Consistência, Isolamento e Durabilidade) e a capacidade de consulta eficiente. No entanto, esses sistemas enfrentam limitações em termos de escalabilidade horizontal e manipulação de grandes volumes de dados não estruturados, o que os torna menos adequados para certas aplicações modernas de grandes dados [9]. O SQL, ou Linguagem de Consulta Estruturada, é descrito como uma linguagem descritiva, na qual as declarações descrevem o resultado desejado sem especificar os passos computacionais necessários para alcançá-lo. Este paradigma permite que as operações de seleção e manipulação se baseiem em conjuntos, facilitando a execução de múltiplas

ações dentro do sistema de gerenciamento de banco de dados por meio de uma única consulta SQL [9]. Além disso, os bancos de dados SQL têm sido o alvo de diversos tipos de ataques de segurança, especialmente injeções SQL, que são bem documentadas e têm uma variedade de medidas de mitigação desenvolvidas ao longo dos anos para combater essas vulnerabilidades [9].

C. Comparação entre SQL e NoSQL

Os bancos de dados SQL tradicionais, diferentemente dos bancos de dados NoSQL, são projetados para expansão fácil e rápida através de muitos nós de servidor, oferecendo reparo automático, *failover* e redução de latência para os usuários [8]. Em relação à garantia de integridade dos dados, os bancos de dados SQL são orientados para consultas complexas e garantem a integridade dos dados através da conformidade ACID. Por outro lado, os bancos de dados NoSQL são projetados para oferecer consistência eventual e são mais adequados para aplicações que podem relaxar algumas restrições de integridade para ganhar em escalabilidade e desempenho [6]. A natureza sem esquema dos bancos de dados NoSQL os torna ideais para o armazenamento e recuperação de dados não estruturados, adaptando-se melhor às necessidades de aplicações modernas que lidam com grandes volumes de dados variados, algo que seria desafiador para os esquemas rígidos dos bancos de dados SQL. Bancos de dados NoSQL geralmente requerem menos recursos para gerenciar dados não relacionais em grande escala, já bancos de dados SQL necessitam de uma infraestrutura significativa para manter a integridade transacional e processamento complexo de consultas em grandes conjuntos de dados [8]. Finalmente, cada tipo de banco de dados tem suas aplicações ideais: NoSQL é preferível para ambientes de web modernos e grandes volumes dados, onde a escalabilidade e flexibilidade são críticas, enquanto SQL é crucial em aplicações que exigem transações complexas e alta integridade de dados, como sistemas financeiros [1].

D. Métodos e Ferramentas para Prevenção e Mitigação de Ataques

Para mitigar ataques de injeção NoSQL, existem vários métodos e ferramentas que podem ser utilizados. A proteção básica contra injeção seria a higienização da entrada, mas isso não salva aplicativos de todos os tipos de injeções, já que o tipo de injeção completa a string de consulta, equilibrando o início e o fim de cada string na consulta [7]. Uma abordagem mais complexa seria a do Autômato Finito Determinístico (DFA), que é empregada para propor um mecanismo de validação de entrada para um banco de dados MongoDB, a fim de detectar e prevenir ataques de injeção NoSQL. Um autômato é um dispositivo computacional abstrato que segue uma sequência programada de operações automaticamente. Um autômato com um número finito de estados é chamado de Autômato Finito (FA) ou Máquina de Estados Finitos (FSM, na sigla em inglês) [5].

E. Ataques de Injeção NoSQL

Embora existam muitas tecnologias para enfrentar ameaças à segurança de software, o ataque de injeção SQL e NoSQL ainda é uma grave ameaça à segurança para aplicações web [3]. Os ataques de injeção NoSQL tendem a ocorrer a uma taxa menor porque as consultas processadas em PHP a partir de *strings* tradicionais são convertidas em objetos antes de serem executadas [6].

Para exemplificar alguns ataques, o artigo de GOMES; KAMALANATHAN, 2019. [6] exemplifica um ataque em um ambiente controlado onde pode ser testado os ataques e a segurança.

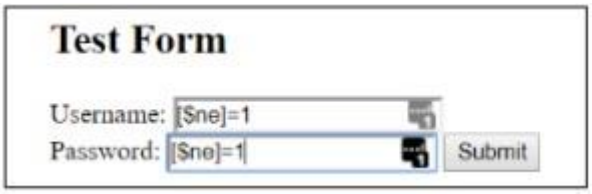

Fig.1. Ataque de Injeção de *Array* PHP

Ataque de Injeção de *Array* PHP: Na página do formulário de login para o site NoSQL, a expressão [$ne]=1, que é o ataque, foi inserida tanto no campo de usuário quanto no campo de senha, como mostrado na fig 1.

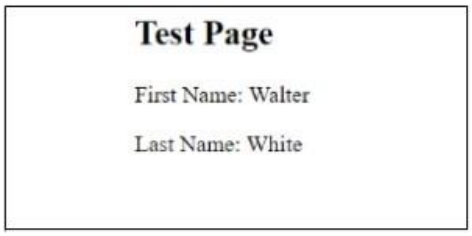

Fig.2. Resultado do ataque de injeção do *array* PHP

A figura 2 mostra o resultado do ataque de injeção do *array* PHP: Neste ataque, o nome de usuário e a senha usados na consulta são apenas afirmações verdadeiras, então os campos solicitados "fname"e "lname"não precisam pertencer a um "username"e "password"específicos. Assim, todos os campos "fname"e "lname"na tabela "login"são retornados.

III. METODOLOGIA

A pesquisa se baseia em uma revisão bibliográfica detalhada sobre as vulnerabilidades de segurança em bancos de dados NoSQL, especialmente relacionadas a ataques de injeção. Com base nos *insights* obtidos, serão conduzidas simulações em um ambiente controlado, utilizando o MongoDB, para avaliar a eficácia de técnicas de mitigação, como a sanitização de entradas e a criptografia. Esses experimentos têm como objetivo validar as estratégias propostas e responder à pergunta de pesquisa, fornecendo uma base prática para a aplicação das técnicas de segurança em sistemas NoSQL.

IV. CONCLUSÃO

Ao longo deste trabalho, foram exploradas as vulnerabilidades de segurança presentes em bancos de dados NoSQL, com especial atenção aos ataques de injeção de código. Uma das hipóteses levantadas foi que a combinação de validação de entradas e criptografia poderia fornecer uma defesa robusta contra essas ameaças. Através de testes realizados em ambiente controlado, foi possível verificar que técnicas como a utilização de Autômatos Finitos Determinísticos (DFA) para validação de entradas, juntamente com a criptografia RSA, oferecem uma camada adicional de proteção que é eficaz em mitigar os riscos de injeções em sistemas NoSQL. Os resultados obtidos reforçam a ideia de que, apesar dos desafios inerentes à segurança em bancos de dados NoSQL, a aplicação de metodologias avançadas de validação e criptografia pode significativamente reduzir as vulnerabilidades associadas. Como trabalhos futuros, serão apresentados estudos comparativos detalhados sobre a eficácia de diferentes metodologias de mitigação e uma proposta de diretrizes para o desenvolvimento seguro de aplicações que utilizam bancos de dados NoSQL.

AGRADECIMENTOS

Expresso minha profunda gratidão á minha família e ao meu orientador, Prof. Me. orientador. Assim como aos demais professores.

REFERÊNCIAS

[1] KHAN, W. et al. *SQL and NoSQL Database Software Architecture Performance Analysis and Assessments—A Systematic Literature Review*. Big Data and Cognitive Computing. MDPI, 1 jun. 2023.

[2] POSTON, H. *What is NoSQL Injection?*. Infosec Institute, 18 fev. 2020. Disponível em: https://www.infosecinstitute.com/resources/applicationsecurity/what-is-nosql-injection/. Acesso em: 2 jul. 2024.

[3] SHACHI, M. et al. *A Survey on Detection and Prevention of SQL and NoSQL Injection Attack on Server-side Applications*. International Journal of Computer Applications, v. 183, n. 10, p. 1–7, 21 jun. 2021.

[4] GUO, D.; ONSTEIN, E. *State-of-the-art geospatial information processing in NoSQL databases*. ISPRS International Journal of Geo-Information. MDPI AG, 1 maio 2020.

[5] LAWAL, M. A.; SALEH, M. A. *Security Testing Tool for NoSQL Systems*. JKAU: Comp. IT. Sci, v. 8, n. 1, p. 85–93, 2019.

[6] GOMES, R.; KAMALANATHAN, D. *Comparing NoSQL and SQL Database Systems Based on Vulnerability to Injection and Adequacy of Countermeasures*. [s.l: s.n.]. Disponível em: https://www.researchgate.net/publication/336530927. Acesso em: 15 ago. 2024.

[7] UL ISLAM, M. R. et al. *Automatic detection of NoSQL injection using supervised learning*. Proceedings - International Computer Software and Applications Conference. Anais... IEEE Computer Society, 1 jul. 2019.

[8] MEIER, A.; KAUFMANN, M. SQL & NoSQL Databases Models, Languages, Consistency Options and Architectures for Big Data Management. [s.l: s.n.].

[9] SACHDEVA, V.; SACHIN GUPTA. *Vulnerability Assesment For Advanced Injection Attacks Against Mongodb*. JOURNAL OF MECHANICS OF CONTINUA AND MATHEMATICAL SCIENCES, v. 14, n. 1, 23 fev. 2019.

[10] DEVALLA, V. et al. MURLi: A Tool for Detection of Malicious URLs and Injection Attacks. Procedia Computer Science. Anais... Elsevier B.V., 2022.