Uma análise de desempenho de funções de encriptação nativas de SGDBs Open Source

Andre de Souza Rosa, Nilson M. Lazarin

[1]Bacharelado em Sistemas de Informação – Centro Federal de Educação Tecnológica Celso Suckow da Fonseca (Cefet/RJ) – Nova Friburgo, RJ – Brazil

andre.rosa@aluno.cefet-rj.br, nilson.lazarin@cefet-rj.br

Abstract. *Given the growing need for security in digital environments, global legislation has emerged to regulate, among other things, data storage. However, security measures impact system performance, causing greater slowdowns and resource consumption. This study evaluates the performance of encryption functions in three Open Source Database Management Systems, MariaDB, PostgreSQL and Firebird, with different workloads. Metrics such as average execution time, average CPU usage and average RAM consumption were considered. Taking into account the results obtained and the security provided by the supported operation modes, PostgreSQL is indicated as the best option.*

Resumo. *Dada a crescente necessidade de segurança em ambientes digitais, legislações globais surgiram para regular, entre outros aspectos, o armazenamento de dados. Porém, as medidas de segurança impactam o desempenho de sistemas, causando maior lentidão e consumo de recursos. Este estudo avalia o desempenho das funções de encriptação em três Sistemas Gerenciadores de Banco de Dados Open Source, o MariaDB, o PostgreSQL e o Firebird, com diferentes cargas de trabalho. Foram consideradas métricas como tempo médio de execução, uso médio de CPU e consumo médio de memória RAM. Levando em conta os resultados obtidos e a segurança proporcionada pelos modos de operação suportados, o PostgreSQL é indicado como melhor opção.*

1. Introdução

Segurança de computadores é proteção direcionada a sistema de informação, visando preservar a confidencialidade, a integridade e a disponibilidade de seus recursos [Stallings 2015]. Os princípios de *Segurança* e *Prevenção* propostos na Lei Geral de Proteção de Dados Pessoais (LGPD) determinam: i) a aplicação de medidas técnicas e administrativas para resguardar os dados pessoais de situações, que ocasionem perda, alteração, comunicação ou divulgação não desejada; e ii) a implementação de medidas para evitar danos oriundos do tratamento de dados [Cidadania 2021]. No contexto dos sistemas de informação, um elemento passível da aplicação de medidas que resguardam os princípios de *segurança* e *prevenção* dos dados sensíveis são os Sistemas Gerenciadores de Banco de Dados (SGBD).

Parte das políticas de segurança a serem implementadas pelas organizações se encontram na dimensão de controle de dados [Carturan et al. 2022], através da adequação da camada de persistência [Pitta et al. 2020], utilizando encriptação para proteger dados em repouso. Para este cenário, os SGBDs possuem funções de encriptação nativas, que

auxiliam na manutenção segura de dados sensíveis no banco de dados. Para auxiliar na tomada de decisões de projetistas de bancos de dados e organizações, trabalhos têm sido propostos [Istifan and Makovac 2022, Kilonzo 2020, Shastri et al. 2019] com o intuito de comparar o desempenho de diferentes SGBDs, proprietários e open source. Entretanto, não abordam os modos de operação dos algoritmos criptográficos [Dworkin 2001] utilizados para proteção de dados nos SGDBs, pois dependendo ao algoritmo e modo de operação utilizados podem ocorrer padrões nos dados em repouso [Lima et al. 2009, Lazarin and Xexéo 2014], como, por exemplo, no modo ECB, que dada a repetição de um bloco de texto em claro submetido ao algoritmo com a mesma chave, resulta sempre num bloco de texto cifrado específico.

A encriptação é crucial para cumprir legislações, porém traz o desafio do equilíbrio *Desempenho x Segurança*, com custo computacional elevado para cifrar/decifrar dados em grande volume. Organizações podem gerar muitos metadados ao atender às legislações, impactando processamento e estrutura de armazenamento [Shastri et al. 2019]. A seleção do SGBD torna-se complexa, pois o desempenho varia com operações e volume de dados, assim como a segurança conforme algoritmos e modos de operação suportados.

Buscando contribuir a tomada de decisão de projetistas de banco de dados, este trabalho apresenta uma avaliação de desempenho de funções de encriptação nativas dos SGBDs open source MariaDB, FireBird e PostgreSQL.

2. Fundamentação Teórica

Nesta seção são apresentados os conceitos do modo de operação e das funções de encriptação dos SGBDs MariaDB, PostgreSQL e Firebird.

2.1. Modo de operação

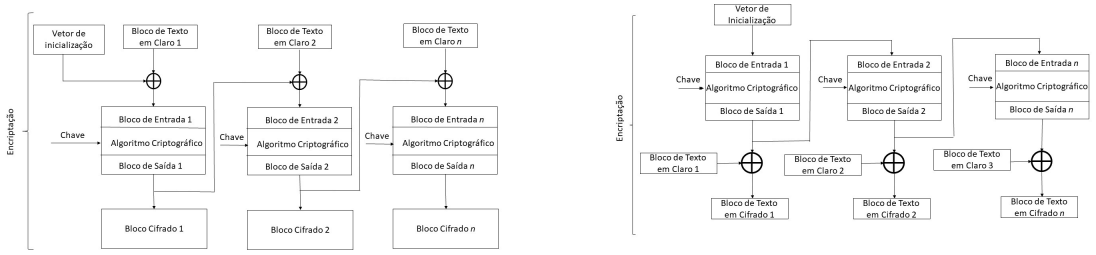
Uma cifra de bloco utiliza bloco de texto de tamanho fixo, de comprimento de X bits, e uma chave como entrada, produzindo uma saída de bloco de texto cifrado de tamanho X. Se a entrada submetida for maior do que X bits, ela pode ser dividida em N blocos de X bits. O Modo de operação é uma técnica utilizada para aperfeiçoar o efeito de um algoritmo de criptografia ou adaptá-lo para uma aplicação [Stallings 2015]. Abaixo são descritos os modos de operação analisados neste trabalho:

(ECB) No modo *Eletronic Codebook* cada bloco de texto em claro é cifrado e decifrado de forma independente, utilizando a mesma chave. Uma mensagem de entrada maior que X é quebrada em blocos de X bits, sendo o último bloco completado caso a quantidade de dados não alcance o tamanho X. Entretanto, se o mesmo bloco de texto em claro de tamanho X ocorrer mais de uma vez na mensagem, o texto cifrado será o mesmo, isso faz com que a aplicação do deste modo não seja recomendável para mensagens longas, pois a repetição pode ocasionar padrões de pares de textos claros/encriptados a serem explorados por criptoanalistas [Stallings 2015].

(CBC) No modo *Cipher Block Chaining*, apresentado na Figura 1a, o primeiro bloco de texto em claro passa por uma operação de XOR com um *Vetor de Inicialização* (VI) e depois é submetido ao algoritmo de encriptação. A partir disso, a entrada da função de encriptação passa a ser a saída da operação *XOR* do bloco de texto em claro a ser cifrado com o bloco cifrado anteriormente. A entrada da função de encriptação não possui

nenhum relacionamento fixo com o texto em claro, dessa forma, repetições de bloco de tamanho *X* de texto em claro não produzem a mesma saída cifrada. Caso faltem dados para preencher o último bloco a ser cifrado, o mesmo também é preenchido [Stallings 2015].

(OFB) No modo *Output Feedback*, apresentado na Figura 1b, para cifrar o primeiro bloco, VI é submetido à função de encriptação. A saída dessa função passa por uma operação XOR com o primeiro bloco de dados em claro, dando origem ao primeiro bloco cifrado. A saída da função criptográfica de um bloco serve de VI para função criptográfica do bloco seguinte. Considerando os blocos de tamanho *X*, se o último bloco for de tamanho menor que *X*, os bits mais significativos são usados para a operação *XOR*. O VI no modo OFB deve ser único para cada execução de encriptação, pois a função de encriptação do primeiro bloco recebe apenas a chave e o VI, sendo a saída um fluxo de bits fixo. Considerando duas mensagens diferentes com bloco de texto em claro idêntico, na mesma posição, o criptoanalista poderia determinar essa parte do fluxo de bits [Stallings 2015].



(a) Modo CBC **(b) Modo OFB**

Figura 1. Modos de operação para cifra simétrica de bloco adaptados de [Dworkin 2001].

2.2. MariaDB

As funções *AES_ENCRYPT* e *AES_DECRYPT* no MariaDB, na versão utilizada nesse trabalho, só recebem como argumento o texto a ser cifrado/decifrado, e a chave a ser utilizada. Nativamente, o tamanho de chave utilizado é de 128 bits. Para utilizar chaves de 256 bits, é necessário fazer mudança no código-fonte do SGDB. Também não é possível alterar o modo de operação utilizado, que por padrão é o ECB.

2.3. PostgreSQL

O PostgreSQL possui a extensão *pgcrypto*, que disponibiliza diversas funções de encriptação. São elas as funções gerais de *hash*, funções de *hash* de senha, funções de encriptação PGP (*Pretty Good Privacy*), funções de encriptação bruta e funções de dados aleatórios [PostgreSQL 2023].

A versão de funções de encriptação utilizadas nesse trabalho no PostgreSQL é a de encriptação bruta. Elas usam a chave diretamente no processo de cifragem, não fornecem nenhum meio de verificação de eventuais alterações em dados encriptados, todo gerenciamento de parâmetros da função ficam por conta do projetista, inclusive o VI e não tem manipulação opcional de dados [PostgreSQL 2023]. Porém, é a versão que mais se aproxima da sintaxe utilizada pelo Firebird, visto que por padrão ambas necessitam da informação explicita do vetor de inicialização a ser utilizado.

2.4. Firebird

No Firebird, as funções de encriptação de campo foram disponibilizadas a partir da versão 4.0. As funções *ENCRYPT()/DECRYPT()* possuem diversos parâmetros, como o texto a ser cifrado/decifrado, o algoritmo de criptografia a ser aplicado, o modo de operação utilizado e a chave criptográfica. Alguns outros parâmetros são aplicados dependendo do modo de operação escolhido, como o VI. A versão das chaves utilizadas não se define na atribuição do algoritmo, mas pelo tamanho da chave utilizada. Uma chave de 16 bytes indica a versão AES-128, uma chave de 24 bytes indica AES-192 e uma chave de 32 bytes indica AES-256 [Firebird 2023].

3. Trabalhos relacionados

No estudo de [Shastri et al. 2019], destaca-se a necessidade de organizações cumprirem a legislação de proteção de dados, que contrasta com princípios de design e práticas em sistemas de computação modernos. O objetivo do trabalho é analisar como modificações para conformidade com a GDPR afetam SGDBs. Foram avaliados o Redis (NoSQL) e dois SGDBs relacionais: um empresarial não divulgado e o PostgreSQL. Sob a GDPR, os SGBDs viram desempenho reduzir de 2 a 5 vezes. Concluiu-se que o SGBD NoSQL sofreu maior impacto, comparado aos relacionais, e que a conformidade é mais fácil com os relacionais.

No estudo [Kilonzo 2020], destaca-se que a encriptação impacta o desempenho, dificultando a implementação de medidas de segurança por organizações e indivíduos. O objetivo do trabalho é avaliar o uso do AES no *Transparent Data Encryption* (TDE) para proteger bancos de dados inteiros em repouso. Foram analisados SGBDs Relacionais (MySQL 8.0 e Oracle 12c) e NoSQL (MongoDB e Cassandra). O AES-256 aumentou o tempo em cerca de 16% em média em relação ao AES-128. Oracle e Cassandra foram mais afetados com a variação da chave. O MySQL teve pior desempenho que o Oracle com AES-128. Na categoria NoSQL, o Cassandra superou o MongoDB.

Em [Istifan and Makovac 2022], destaca-se a falta de estudos sobre o impacto da encriptação de dados em repouso, utilizando o mesmo algoritmo, na taxa de Transações Por Minuto (TPM) com aumento de usuários simultâneos. O objetivo é avaliar a diferença de desempenho, em termos de taxa de transferência, entre versões diferentes tamanhos de chaves do AES (*Advanced Encryption Standard*) para encriptação. O estudo comparou MariaDB e MySQL, utilizando AES-192 para análise de desempenho. Os resultados mostram diferença significativa de desempenho em termos de TPM do AES-192 entre MySQL (AES-256) e MariaDB. MariaDB superou MySQL em testes de carga e estresse para o AES-192.

Este estudo avalia o impacto das medidas técnicas para proteção de dados pessoais, necessárias para cumprir a legislação. Ao contrário de [Shastri et al. 2019], focamos nas funções nativas dos SGDBs, sem modificar o software. Diferente de [Kilonzo 2020], consideramos a encriptação de campos específicos sem instruções de CPU dedicadas. Comparado a [Istifan and Makovac 2022], nosso estudo expande a comparação de desempenho e segurança, avaliando algoritmo e modos de operação em mais opções de software open source.

4. Métodos

O objetivo desse trabalho é avaliar o desempenho de funções de encriptação nativas dos SGBDs MariaDB, Firebird e PostgreSQL. Para isso, serão aplicadas diferentes faixas de cargas de trabalho aos SGBDs através de uma aplicação desenvolvida, utilizando instruções SQL de inserção, consulta, atualização e remoção de registros.

São realizados testes com texto em claro e utilizando as funções de encriptação. Dessa forma, será possível avaliar o impacto que as funções de encriptação causam no desempenho dos SGBDs. A massa de dados utilizada nos experimentos é referente a registros de ocupação hospitalar, adquirida na plataforma openDataSus[1] em formato *.CSV*. A versão utilizada nos testes desse conjunto de dados foi adquirida em novembro de 2022, composto de 745015 linhas, de tamanhos de dados variados. Para retratar o conjunto de dados, na camada de persistência, foram projetadas em 5 entidades. O modelo relacional definido para os experimentos pode ser visto na Figura 2.

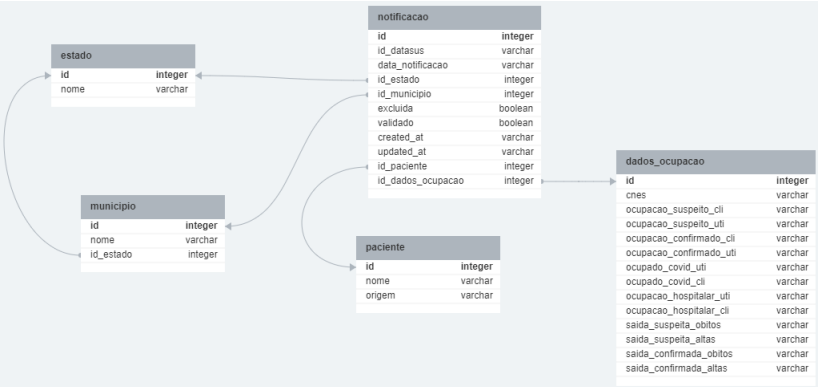


Figura 2. Modelo conceitual de entidades adotado.

A decisão de dividir o conteúdo do conjunto de dados em 5 tabelas foi tomada, para que se tornasse possível realizar operações condicionais nas operações de *UPDATE* e *DELETE* envolvendo 2 tabelas distintas. Isso permite criar instruções complexas, como condicionais envolvendo verificação de conteúdo de colunas de tabelas diferentes com conteúdo encriptado, no caso de operações com as funções de encriptação, aumentando a demanda nos SGBDs com o processo de decriptação e verificação das condicionais, por exemplo. A aplicação utilizada para realizar os testes, o arquivo .CSV, o projeto da base de dados que segue a mesma estrutura nos 3 SGBDs e os resultados absolutos obtidos podem ser baixados[2].

4.1. Estrutura dos testes

Os testes para medir o desempenho dos SGBDs foram realizados com faixas de cargas de trabalho. A definição de registro utilizada corresponde a uma linha do arquivo .CSV, cujo conteúdo das colunas que a compõe é utilizado para alimentar as 5 tabelas da base de dados. A definição de carga utilizada em um teste corresponde a quantidade de registros do arquivo .CSV utilizadas nas operações de teste. As faixas de carga são 1000

[1] `https://opendatasus.saude.gov.br/`
[2] `https://github.com/LabRedesCefetNF/Rosa_2023`

registros, 2000 registros, 4000 registros e 8000 registros. Esses testes foram realizados, na mesma quantidade, utilizando as funções de encriptação e executadas as operações com os dados em claro, para cada SGBD. As funções de encriptação foram utilizadas em todos os campos textuais (*varchar*) na inserção dos registros, assim como as funções de decriptação a esses campos nas consultas e nos campos envolvidos nas condicionais de alteração e remoção de registros. As operações de inserção e leitura envolvem as 5 tabelas. As operações de atualização e remoção acessam as 5 tabelas para recuperar os dados de um registro. Após isso, é feita a verificação condicional entre os dados da tabela *estado* e *notificacao*, para atualizar o nome do estado vinculado daquela notificação para 'Alagoas', caso não seja. O campo *nome* de estado ao ser atualizado, é armazenado com seu valor encriptado, nas operações que empregam encriptação. Nos testes de exclusão, essa relação é novamente avaliada para remover todos os registros que tenham como estado 'Alagoas', sendo assim removidos todos os dados da base.

Para medir o desempenho dos SGBDs durante a execução dos testes, foram mensurados o tempo médio de execução de cada operação realizada, o percentual médio de CPU utilizado durante cada operação e a quantidade média de memória RAM ocupada durante cada operação.

4.2. Ambiente utilizado

O ambiente que os testes foram executados utilizaram a mesma configuração na máquina virtual e de Sistema Operacional (S.O.). A máquina virtual utilizou como S.O. o Debian GNU/Linux 11, contou com 4 GB de memória RAM, o processador foi limitado a utilização de apenas 1 núcleo, com o intuito de evitar vantagem de execução entre os modos de operação. Isso pelo fato de que o ECB pode ser executado em paralelo, enquanto o CBC e o OFB não podem ser paralelizáveis [Detomini 2010]. As máquinas virutias rodaram num cluster que possui as CPUs: 4 x Intel(R) Core(TM) i5-3470 CPU @ 3.20GHz (1 Socket) — 8 x Intel(R) Xeon(R) CPU E5450 @ 3.00GHz (2 Sockets).

A versão dos SGBDs utilizados foram o PostgreSQL 14.7, o MariaDB 10.8.5 e o Firebird 4.0.2. Para interagir com o PostgreSQL, foi utilizada a ferramenta pgAdmin 4, para interagir com o Firebird foi utilizada a ferramenta FlameRobin e para interagir com o MariaDB foi utilizado seu cliente nativo via terminal. Foram selecionados apenas SGBDs open source pelo fato de que existe uma limitação legal imposta pelo licenciamento dos SGBDs proprietários que impedem a divulgação de estudos de benchmark de seus produtos sem autorização da Empresa detentora dos direitos, como citado nas discussões gerais do trabalho de [Istifan and Makovac 2022].

5. Resultados

Nessa seção, serão apresentados os resultados aproximados obtidos após a execução dos testes e serão feitas algumas considerações a respeito dos mesmos. Cada teste resultou em um arquivo de monitoramento da CPU, memória RAM e *swap* para operações (Insert, Select, Update, Delete). A área de *swap* não foi utilizada em nenhum teste em nenhum SGBD e, portanto, foi desconsiderada como métrica. Também, gerou-se um arquivo para o tempo de execução de cada operação em cada teste, totalizando 13 arquivos por teste.

Foram realizados 10 testes para cada faixa de carga de trabalho (1000, 2000, 4000 e 8000 registros). Desconsiderando os arquivos com registros de *swap* , cada teste deu

origem a 9 arquivos a serem analisados. Levando em conta que a mesma carga de teste foi aplicada para execução com dados em claro e com a aplicação de encriptação, para cada um dos SGBDs, o total de arquivos com registros capturados analisados foram de 2160 arquivos.

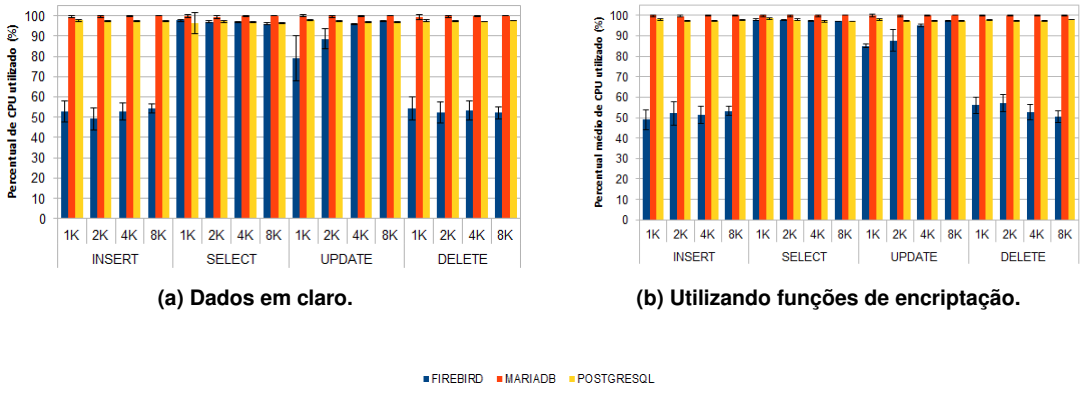
A análise dos resultados consistiu, no primeiro momento, na média dos valores registrados, que fazem parte do escopo de métricas adotadas, em cada arquivo. Ao analisar o percentual de CPU utilizado, com exceção da primeira linha do arquivo, para cada linha de dados registrados, foram somados os valores referentes ao uso de CPU por parte do sistema com o valor de CPU usado pelo usuário. Ao final do arquivo, foi calculada a média dos valores somados e o desvio padrão relacionado.

Ao fazer o primeiro passo da análise como os arquivos referentes a ocupação de memória RAM durante a execução dos testes, foram levados em conta o total de memória disponível (MT) e a memória livre (ML). Foi realizada a subtração dos valores de MT por ML para cada linha, sendo calculada ao fim do arquivo a média dos valores e o desvio padrão.

Após calcular a média e desvio padrão das métricas em cada arquivo, foi calculada a média e o desvio padrão dos resultados obtidos nesses arquivos por faixa de carga de trabalho para cada operação. Esses resultados permitiram a comparação do desempenho entre SGBDs com base nas métricas utilizadas. A barra de erro nos gráficos em cada faixa de carga de trabalho indica o desvio padrão correspondente.

5.1. Comparativo dos SGBDs com relação à média do uso de CPU

É possível perceber que o MariaDB obteve a maior taxa de uso da CPU, seguido pelo PostgreSQL. O Firebird teve uma utilização da CPU menor que os demais. Porém, o uso de CPU reduzido do Firebird se dá pelo grande número de esperas que o SGBD executa durante a execução das operações, que se reflete no maior consumo de tempo de execução citado anteriormente. A Figura 3a exibe gráfico baseado nesses resultados.



(a) Dados em claro. **(b) Utilizando funções de encriptação.**

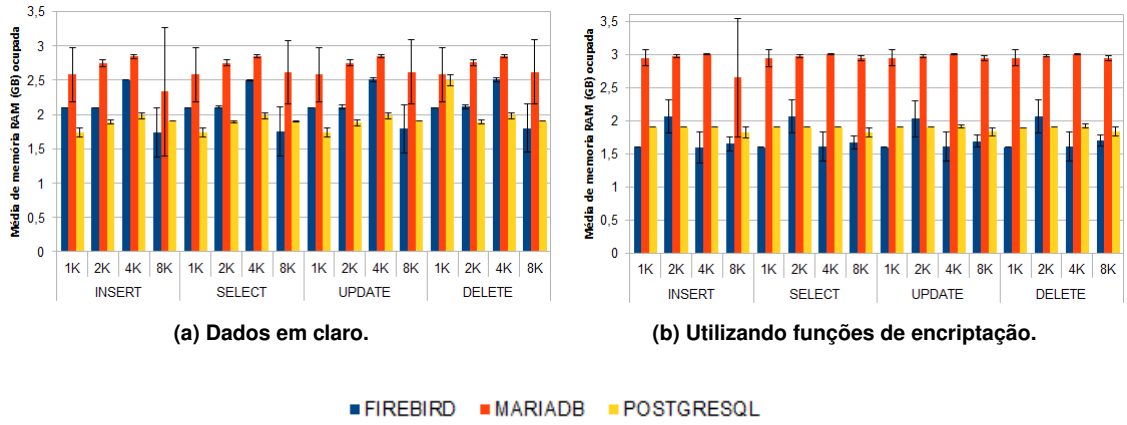
Figura 3. Comparação do percentual médio de uso de CPU durante a execução das operações.

Os resultados revelam que, de forma semelhante aos resultados das operações com dados em claro, o MariaDB possui o maior percentual de CPU utilizado, enquanto o FireBird continua com uma taxa de utilização menor. A diminuição do uso de CPU, vista em algumas faixas de carga de trabalho, ocorre com o aumento de ocorrência de estado

de espera durante a execução das operações com as funções de encriptação. A Figura 3b expõe as diferenças.

5.2. Comparativo dos SGBDs com relação à média de ocupação de memória RAM

Os resultados referentes à ocupação de memória RAM, em operações com dados em claro apresentam, de modo geral, que o MariaDB foi o SGBD com maiores médias de ocupação de memória RAM. O PostgreSQL é o que possui menor taxa de ocupação de memória RAM na maior parte de cargas de trabalho aplicada, com exceção da faixa de 8000 registros. Nessa faixa, ele supera o Firebird em todas as operações. Uma peculiaridade percebida é que, na faixa de 4000 registros, os SGBDs atingem seu pico de ocupação em todas as operações. Os resultados são representados na Figura 4a.



(a) Dados em claro.

(b) Utilizando funções de encriptação.

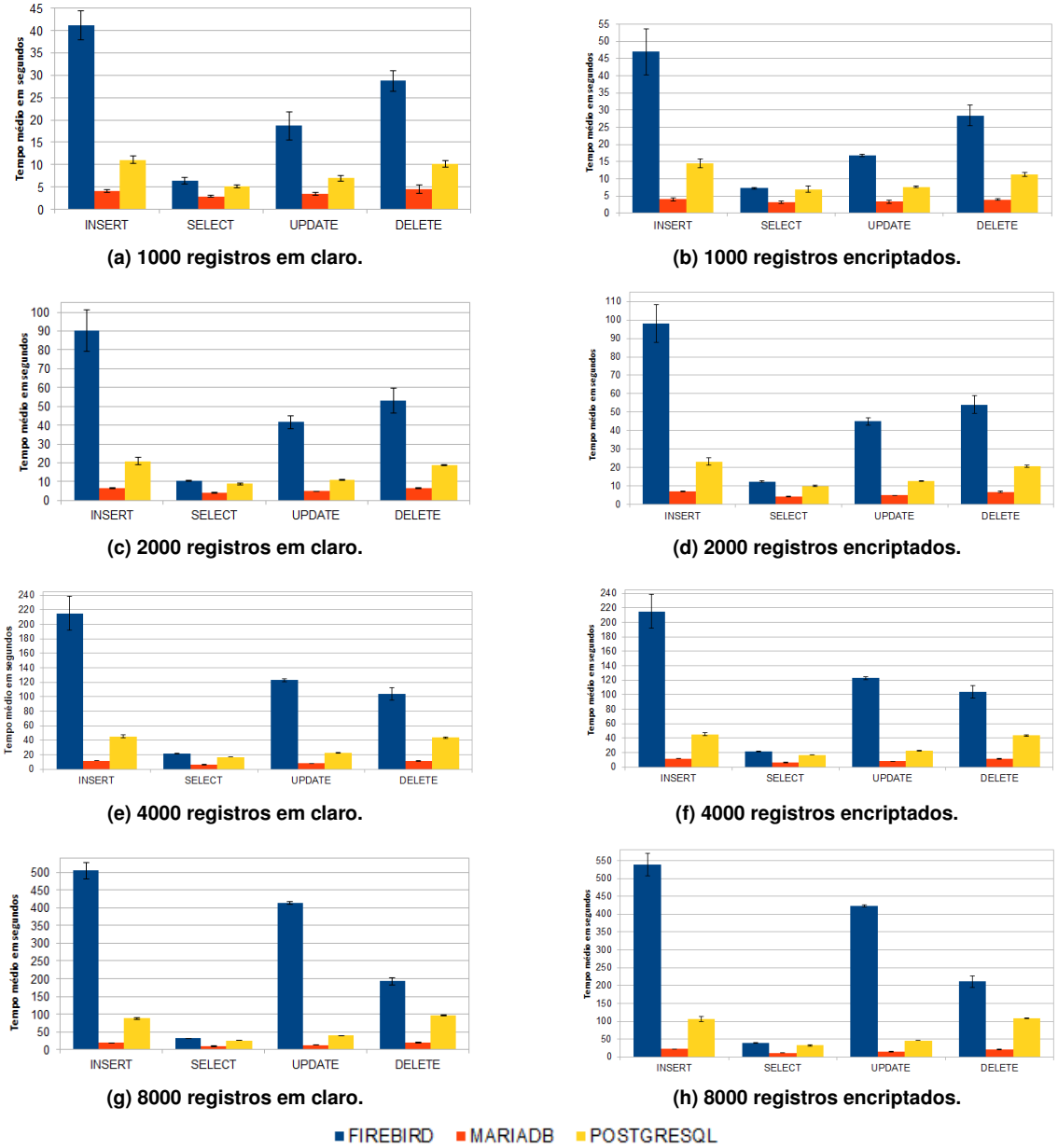
Figura 4. Comparação da média de ocupação de memória RAM (GB) durante a execução as operações.

Os resultados obtidos para os testes de ocupação de memória RAM durante a execução das operações, utilizando funções de encriptação demonstram é possível perceber que o PostgreSQL passa a ocupar mais memória RAM em relação ao Firebird em mais faixas que as verificadas, quando utilizados dados em claro. Com exceção da faixa de 2000 registros de carga de trabalho, o PostgreSQL ocupou mais memória RAM que o Firebird em todas as outras. Uma possível causa para essa mudança, é o aumento no tempo de execução das operações quando utilizadas as funções de encriptação, visto que o Firebird tem o processamento mais ocioso nesse cenário. O gráfico é apresentado na Figura 4b.

5.3. Comparativo dos SGBDs com relação à média do tempo de execução em claro

Nesse critério, vemos que o Firebird teve o pior desempenho em relação aos demais, principalmente para operações envolvendo escrita de dados, consumindo muito mais tempo do que o PostgreSQL, que obteve o segundo melhor resultado. O MariaDB obteve um desempenho muito superior aos demais. Porém, se faz necessária a ressalva da influência do modo de operação, já que o ECB utilizado por ele, não faz uso de vetor de inicialização e, portanto, tendo as instruções mais simples que os demais. A partir do desvio padrão de cada média apresentada por faixa, é possível verificar o grau de variação dos dados utilizados para cálculo em torno da média, sendo o Firebird o SGBD que, de modo geral, apresenta a maior variação. A Figura 5 apresenta os resultados deste experimento.

Os resultados obtidos a respeito do tempo médio de execução utilizado nas operações, a partir dos testes com funções de encriptação apresentam, de forma geral, que ocorreu um acréscimo na quantidade de tempo consumido para todos os SGBDs, mantendo-se no, entanto, o MariaDB com o melhor desempenho e o Firebird com o pior tempo alcançado.



(a) 1000 registros em claro.

(b) 1000 registros encriptados.

(c) 2000 registros em claro.

(d) 2000 registros encriptados.

(e) 4000 registros em claro.

(f) 4000 registros encriptados.

(g) 8000 registros em claro.

(h) 8000 registros encriptados.

Figura 5. Comparação de tempo de execução (em segundos).

5.4. Comparação do impacto causado no tempo médio de execução pelas funções de encriptação

Com base nos resultados obtidos dos testes de tempo de execução, com os dados em claro e utilizando as funções de encriptação, foi feito um comparativo de como o emprego das funções afeta o desempenho dos SGBDs. Para isso, foi primeiramente calculada

a diferença percentual, entre os resultados obtidos dos testes utilizando as funções de encriptação em relação aos testes de tempo de execução com os dados em claro, em cada faixa de carga de trabalho, para cada operação realizada (INSERT, SELECT, UPDATE e DELETE). Após isso, para cada operação, foi calculada a média das diferenças percentuais encontradas em cada faixa de carga de trabalho. Tendo a média do cálculo diferencial de cada operação, foram calculadas as médias gerais do custo no desempenho, em relação a tempo de execução, para cada SGBD, apresentadas na Tabela 1.

O cálculo não foi realizado para as outras métricas devido à relação entre o tempo de execução consumido e a ocupação de memória e uso de CPU. Como exemplo, temos o Firebird, com um grande volume de interrupções que causam espera e subutilização da CPU, alongado o seu tempo de execução e apresentando uma menor ocupação de memória que os demais SGBDs na maior parte das operações realizadas, quando aplicadas as funções de encriptação. O Firebird apresenta um menor uso de CPU que os demais SGBDs na maior parte das operações e uma ocupação de memória intermediária entre os 3 avaliados, nas operações em claro, resultado num maior tempo de execução. Quando adicionada a complexidade das funções de encriptação, o Firebird passa a ter também uma ocupação de memória menor que os demais, na maior parte das cargas de trabalho, aumentando ainda mais o seu tempo de execução.

O PostgreSQL foi o mais afetado pela encriptação quando comparado seu desempenho com os dados em claro. Uma causa possível, baseado nos resultados obtidos, é a menor ocupação de memória RAM durante a execução das operações utilizando as funções de encriptação, ocasionando mais lentidão no processamento.

O MariaDB apresentou o menor impacto no tempo de execução com o emprego das funções de encriptação. O Uso de CPU e ocupação de memória RAM no MariaDB foram os maiores, tanto para operações com dados em claro, quanto para quando as funções de encriptação, executando as operações em um tempo muito menor que os demais. O ponto a ser ponderado, é o modo de operação mais vulnerável a ataques, o que pelo critério de segurança, não é desejado.

			Custo no desempenho	
SGBD	**Algoritmo**	**Modo de Operação**	**Média aprox.**	**Desv. Pad. aproximado**
MariaDB	AES-128	ECB	4,6%	4,4
Firebird	AES-128	OFB	6,7%	7,3
PostgreSQL	AES-128	CBC	16,2%	5,01

Tabela 1. Custo do emprego das funções de encriptação no tempo de execução das operações dos SGBDs.

6. Considerações finais

Esse trabalho apresentou uma análise de desempenho das funções de encriptação nativas de SGBDs open source, sendo os escolhidos para estudo o PostgreSQL, o Firebird e o MariaDB. O desempenho das funções de encriptação foi avaliado através das métricas de tempo médio de execução consumido, percentual médio de utilização de CPU, que

são métricas similares às utilizadas no trabalho de [Kilonzo 2020], além da média de ocupação de memória RAM. Essas métricas foram aferidas durante as operações de inserção, busca, atualização e remoção de registros nos SGBDs.

Diante dos resultados obtidos, foi comparado o desempenho entre os SGBDs com e sem o uso de funções de encriptação, avaliando o custo associado. Foi observado que o tempo de execução das operações é influenciado pela relação entre uso de CPU e ocupação de memória RAM.

O impacto das funções de encriptação no tempo de execução foi avaliado, sendo o MariaDB o menos afetado, com um custo médio de cerca de 4,6%. Dado o menor custo médio e os comparativos de utilização de CPU e memória RAM, o MariaDB demonstrou melhor aproveitamento dos recursos computacionais. Contudo, deve-se mencionar a fragilidade do modo de operação ECB no MariaDB, que gera repetição de textos encriptados para blocos idênticos, tornando-o inseguro. Sua simplicidade em relação a modos como CBC e OFB pode ter influenciado nos resultados.

A avaliação do impacto do emprego das funções de encriptação no tempo de execução do SGBDs mostra que o PostgreSQL é o mais afetado, com um custo médio de aproximadamente 16,2%. O Firebird obteve um custo médio intermediário entre os SGBDs avaliados, de aproximadamente 6,7%. Verificou-se, porém, que quando avaliado em relação aos demais em relação a tempo de execução, foi o mais lento, com CPU e memória RAM subutilizadas. Foi o SGBD com maior complexidade no uso das funções de encriptação.

Ao avaliar a necessidade de equilibrar Desempenho x Segurança, acentuada por regulamentações como a LGPD, os resultados dos testes indicam que o PostgreSQL oferece maior segurança que o MariaDB. O MariaDB teve o melhor desempenho, enquanto o Firebird, apesar de ter maior foco em segurança que o MariaDB, foi mais lento. Como visto nos resultados demonstrados, o uso de funções de encriptação, especialmente em cargas maiores (4000 e 8000 registros), impacta o tempo de execução dos SGBDs, sendo o PostgreSQL o mais impactado com maiores cargas. Se num projeto hipotético em que as funções de encriptação forem aplicadas com menores porções de registros, o custo computacional agregado com a segurança proporcionada pode ser menos comprometedor. Também há que as funções de encriptação são aplicadas a campos específicos. Numa aplicação que envolve poucos dados sensíveis, o tempo de execução das operações sofrerão menos atrasos. Além do fato que os campos após a encriptação ocupam mais espaço em disco, adotando uma abordagem seletiva de aplicação de encriptação, o gasto com espaço de armazenamento se torna menor.

Os resultados e conclusões deste trabalho são específicos ao ambiente, à encriptação e aos testes realizados, não representando uma posição universal. Como sugestões para pesquisas futuras, poderia ser analisada a implementação do algoritmo AES em cada SGBD. Isso visaria a verificar possíveis diferenças no nível de segurança e oportunidades de aperfeiçoamento para melhorar o desempenho. Há possibilidade para uma análise mais aprofundada das características de implementação e configuração dos SGBDs, que impactam o desempenho evidenciado neste estudo, bem como explorar recursos além das configurações padrão para melhorar o desempenho. Após o término do estudo, uma nova versão *alpha* do MariaDB (11.2.1) foi lançada, introduzindo mais mo-

dos de operação e tamanhos de chaves. Isso pode estimular novos experimentos para verificar se seu desempenho continua superior em comparação com os outros, mesmo com instruções igualmente complexas."

Referências

Carturan, S., Matsui, B., and Goya, D. (2022). LGPD Framework: An implementation and compliance guide for technology areas. In *Anais do XLIX Seminário Integrado de Software e Hardware*, pages 176–187, Porto Alegre, RS, Brasil. SBC. https://doi.org/10.5753/semish.2022.223289.

Cidadania, M. d. (2021). Princípios da LGPD. Disponível em: https://www.gov.br/cidadania/pt-br/acesso-a-informacao/lgpd/principios-da-lgpd. Acessado em 9 de Agosto de 2022.

Detomini, R. C. (2010). Exploração de paralelismo em criptografia utilizando GPUs. *SJR Preto, Universidade Julio de Mesquita Filho*.

Dworkin, M. J. (2001). Sp 800-38a 2001 edition. recommendation for block cipher modes of operation: Methods and techniques. Technical report, Gaithersburg, MD, USA.

Firebird (2023). 8.11 Cryptographic Functions. https://firebirdsql.org/file/documentation/chunk/en/refdocs/fblangref40/fblangref40-scalarfuncs-crypto.html.

Istifan, S. and Makovac, M. (2022). Performance benchmarking of data-at-rest encryption in relational databases. https://urn.kb.se/resolve?urn=urn:nbn:se:his:diva-21309.

Kilonzo, A. M. (2020). *Impact of Symmetric-Key Encryption on Performance of Relational and Nosql Database to Preserve Data Confidentiality*. PhD thesis, United States International University-Africa.

Lazarin, N. M. and Xexéo, J. A. M. (2014). Reconhecimento de padrões em criptogramas. *Revista Militar de Ciência e Tecnologia*, v. 31(n. 2):26–43.

Lima, A. P. d., Xexéo, J. A. M., and Gomes, P. R. (2009). Novas formas de ataques de distinção a cifradores: caminhos para o futuro. *Revista Militar de Ciência e Tecnologia*, v. 26(n. 2):12–17.

Pitta, P. E. B., Costa, E., de Siqueira, J. P. L., and Lazarin, N. M. (2020). LGPD Compliance: A security persistence data layer. In *Anais da XVIII Escola Regional de Redes de Computadores*, pages 123–127, Porto Alegre, RS, Brasil. SBC. https://doi.org/10.5753/errc.2020.15200.

PostgreSQL (2023). PostgreSQL: Documentation: 15: F.28. pgcrypto. Disponível em: https://www.postgresql.org/docs/current/pgcrypto.html\#PGCRYPTO-WITH-WITHOUT-OPENSSL/. Acessado em 14 de junho de 2023.

Shastri, S., Banakar, V., Wasserman, M., Kumar, A., and Chidambaram, V. (2019). Understanding and benchmarking the impact of gdpr on database systems. *arXiv preprint arXiv:1910.00728*.

Stallings, W. (2015). *Criptografia e segurança de redes: princípios e práticas*. Pearson Education do Brasil, São Paulo, 6 edition.