Análise do Impacto de Funções Criptográficas no Desempenho do Google BigQuery

Luiz Guilherme Fritsch[1,3], **Giulliano Paz**[2,3], **Rodrigo Bisso**[1,3], **Diego Kreutz**[1,2,3]

[1]Laboratório de Estudos Avançados (LEA)
[2] Mestrado Profissional em Engenharia de Software (MPES)
[3] Universidade Federal do Pampa (UNIPAMPA)

1. Por que utilizar bases de dados cifradas?

Os bancos de dados cifrados surgiram com o objetivo de mitigar o impacto dos vazamentos de dados sensíveis, sejam eles por funcionários mal intencionados ou ataques externos. Diferentemente dos vazamentos de dados da maioria absoluta dos sistemas tradicionais, vazamentos de dados cifrados não comprometem a confidencialidade dos dados. Entretanto, ainda há poucos trabalhos avaliando o fator crucial e determinante para a utilização de bancos de dados cifrados, o desempenho.

Estudos sobre o HBase (`https://hbase.apache.org/`) demonstram que o tempo de computação para dados cifrados aumenta entre 47% e 70% [Pallas et al. 2016]. Outras soluções, como o CryptDB [Popa et al. 2011], utilizam um tipo diferente de criptografia, a homomórfica. Este tipo de criptografia possibilita tipos específicos de operações sobre os dados sem a necessidade de decifrá-los. Entretanto, um dos maiores limitadores dessa tecnologia ainda é o custo computacional elevado.

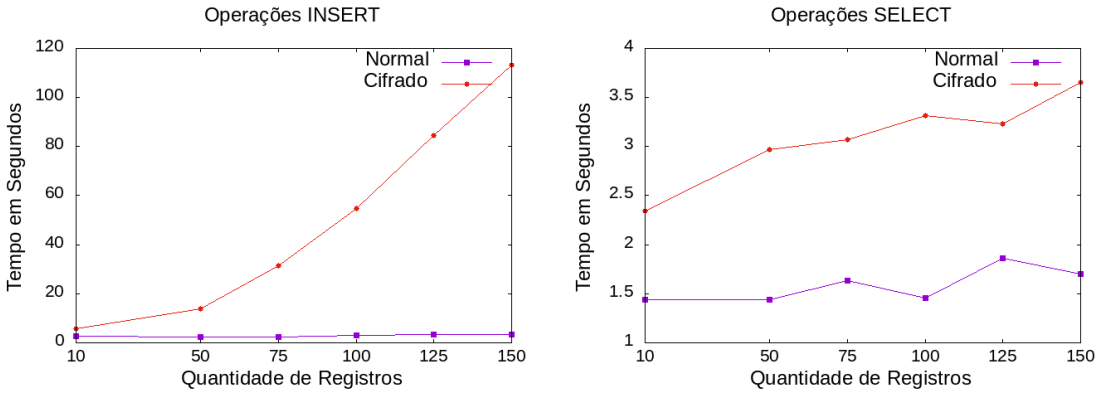
Diferentemente do HBase e do CryptDB, criados para utilização local, existem as soluções elásticas, em nuvem, como é o caso do Google BigQuery (`https://cloud.google.com/bigquery`). No BigQuery, para fins de maior segurança e robustez da solução, os dados são armazenados em blocos distintos, cada um com uma chave criptográfica única. Em caso de vazamento dos dados ou comprometimento da chave de um bloco de dados, os demais ainda permanecem protegidos. O objetivo deste trabalho é avaliar o impacto da criptografia e da distribuição dos dados em blocos, com chaves únicas, no desempenho do Google BigQuery.

2. Avaliação do desempenho do BigQuery

Para realizar os testes de desempenho, foi implementado um programa Python que realiza operações de criação de tabelas, inserção e seleção de dados de clientes fictícios (cifrados e em texto plano). Na implementação, pensando em portabilidade, foi utilizada a sintaxe SQL em vez da API Python. Cada registro é constituído pelo código identificador do cliente, o nome completo, a idade e o e-mail. Em média, os registros utilizados possuem 190 bytes. Para realizar a inserção de dados, foi criada uma função para a criação de uma *query* única, inserindo os dados em lotes. Esta abordagem foi escolhida porque a inserção de registros um-a-um mostrou-se ineficiente. Os resultados representam a média de 50 execuções para cada conjunto de registros (10, 50, 75, 100, 125 e 150). O número de 150 registros está relacionado ao limite permitido pelo BigQuery no plano gratuito. Os testes foram executados em um computador com Ubuntu 16.04.6 LTS, um processador Intel Xeon X5690, 8 vCPUs, 48GB de memória RAM e link de Internet de 200MB/s.

O primeiro teste realizado foi o de criação de tabelas. Em média, o tempo de criação das tabelas foi de 1 segundo, tanto com quanto sem a utilização das funções criptográficas. Entretanto, no teste seguinte, de inserção, ocorreu uma variação significativa no tempo de execução, como pode ser visto nos gráficos da Figura 1. Na inserção de dados, o tempo tende a crescer significantemente de acordo com a quantidade de registros. Por exemplo, a diferença chega a ultrapassar 1100% para 150 registros. No caso da seleção, a diferença se mantém praticamente constante entre as configurações com e sem criptografia de dados, apresentando variações de segundos apenas.

Figura 1. Resultados das operações *INSERT* e *SELECT*



3. Conclusão

Empresas que armazenam dados sensíveis devem priorizar bases de dados cifradas ou novas tecnologias capazes de mitigar o efeito de vazamentos de dados [Machado et al. 2019]. A avaliação do BigQuery mostra que, mesmo em soluções comerciais robustas, ainda há um preço a pagar pela segurança dos dados. Entretanto, diferentemente de soluções como a CryptDB, o BigQuery oferece um custo computacional significativamente menor. Vale ressaltar que os usuários estão cobrando cada vez mais este preço das empresas, ou seja, a proteção (e.g., confidencialidade e privacidade) dos seus dados [Machado et al. 2019].

Os próximos passos do trabalho incluem: (a) investigar o impacto da latência da rede nos tempos de execuções; (b) utilizar registros de diferentes tipos, formatos e volumes; (c) avaliar e comparar outras soluções em nuvem, como a Always Encrypted da Microsoft; (d) comparar as diferentes interfaces de utilização do BigQuery; e (e) investigar como o BigQuery gerencia as chaves.

Referências

Machado, R. B., Kreutz, D., Paz, G., and Rodrigues, G. (2019). Vazamentos de Dados: Histórico, Impacto Socioeconômico e as Novas Leis de Proteção de Dados. In *4o Workshop Regional de Segurança da Informação e de Sistemas Computacionais*, WRSeg '19, pages –. SBC. Paper: `http://tiny.cc/wrseg19-dl` Site: `http://bit.do/wrseg19-ws`.

Pallas, F., Günther, J., and Bermbach, D. (2016). Pick your choice in HBase: Security or performance. In *2016 IEEE Int. Conference on Big Data (Big Data)*, pages 548–554.

Popa, R. A., Redfield, C., Zeldovich, N., and Balakrishnan, H. (2011). CryptDB: protecting confidentiality with encrypted query processing. In *33rd ACM SOSP*, pages 85–100. ACM.