2E4-OS-25a-3

NOSQL データベースでの機密データの管理と暗号化

Confidential data management in NOSQL database

岩瀬 高博[*1]
Takahiro Iwase

[*1] 株式会社 神戸デジタル・ラボ
Kobe Digital Labo, inc

This paper proves validity of using NOSQL database to manage highly confidential data. Verifies the change in the processing capabilities by incorporating encryption technology to NOSQL Database "okuyama".

1. はじめに

近年インターネットサービスの普及により多くの人々がソーシャルネットワークサービスに代表されるインターネットサービスに容易に情報を公開出来るようになってきている。またこれらのサービスを利用することで出力されるシステム上の利用ログも同時に大量に生成されている。

そのようなデータを有効に活用するため、集計、分析等の研究も活発に行われ始めている。しかしこれらのデータは個人のデータの集合であり、ライフログに近い存在であるため多くの個人情報を含むことが非常に多い。そのためこれらのログを活用するに際しては秘匿性や利用シーンの限定などを常に意識して進める必要がある。

一方これらのデータを保存、管理するため近年リレーショナルデータベース(以下 RDBMS)以外の選択肢として NOSQL データベース(以下 NOSQL)が注目されている。NOSQL は RDBMS が持つトランザクション処理や複雑な集合化処理、検索処理、ログイン認証や暗号化などを省くかわりにデータへの単純なアクセス性能とスケールアウトによる容易な処理能力の拡張に着目して開発されたデータベースである。そのため近年のビッグデータと称される大量データの管理において有効な手段として用いられ始めている。

しかし前述の通りビッグデータ管理においては単純な処理能力以外にデータの持つ属性に配慮したデータ管理などが今後必須となることが予想される。そのため NOSQL に対しても秘匿データの管理などに必須と考えられる利用者認証や暗号化機能が今後求められることが想定される。

そこで本論文では、著者が開発するオープンソース NOSQL である「分散キー・バリュー・ストア okuyama」へ暗号化機能を実装し、暗号化処理を実施前と実施後の処理能力を計測し、その結果を基に NOSQL への暗号化機能適応に関しての考察を行う。

2. 分散キー・バリュー・ストア okuyama

okuyama は Java 言語により実装されたマルチプラットホーム型のデータベースである。

データ構造には Key によりデータを索引管理する Key-Value 型を採用している。

システム構成としてはクライアント/マスターノード/データノードの 3 つの要素で構成される。

- クライアント
  クライアントは okuyama を利用するプログラム上で用いられる okuyama へのアクセスを行うライブラリである。Java 言語と PHP 言語の 2 種類が提供されている。
- マスターノード
  クライアントプログラムとデータ保存部であるデータノード間の中継を行う。またデータノードの生存監視などを行うマネージャノードでもある。
- データノード
  データ保存用ノードでありマスターノードから依頼されデータを保存、取り出し、削除を行うノードである。

これら 3 つの要素は全て TCP/IP により結ばれ処理が行われる。

特徴としては大量データの管理を得意としておりデータ量に応じて各ノードを複数のサーバで構築可能であり、マルチサーバ構成にて 1 つのデータベースとして構築可能である。また各ノードの動的追加に対応しており、ノードを追加することで処理能力、保存キャパシティー共に向上させることが可能である。

保存されるデータはメモリ上もしくはディスク上に BASE64 エンコードフォーマットにて保存され、性能要求に応じて保存場所を変更することが可能となっている。

3. 暗号化箇所について

暗号化を行う上で実施箇所についての検討が必要である。本検証ではその際、既存の RDBMS を参考とし実装機能を設計することとした。

参考とした RDBMS はオープンソースソフトウェアであり、利用率も高い MySQL とした。

調査[MySQL-参考 1]を実施したところ、暗号化機能は大きく 2 種類存在することがわかった。

(1) 通信暗号化

クライアントとデータベース間の通信を SSL にて行い通信内容の盗聴に備える暗号化機能。

(2) 保存データ暗号化

保存されるデータを暗号化し保存データの盗難に備える暗号化機能。

本検証では調査結果から 2 箇所の暗号化機能を okuyama へ実装するものとした。

連絡先:岩瀬高博,(株)神戸デジタル・ラボ,兵庫県神戸市中央区江戸町 93 番 栄光ビル 5F, iwase@kdl.co.jp

4. 実証実験

4.1 実験背景と目的

実験では暗号化を行うことでのデータアクセス性能の変化を調べるものとする。これは暗号化による処理能力の変化を明らかにすることで、ビッデータの中にある秘匿性の高いデータを管理する目的で NOSQL に対して暗号化機能を搭載にすることの影響を明らかにし、秘匿データ管理における NOSQL の活用の可能性を検証すると同時に有効性を検証するためである。

4.2 実験内容

通信経路の暗号化及び、データ自体の暗号化を行いそれぞれのケースでの性能を測定し、最終的に両方の暗号化を行なった上での性能検証を行う。

データへの処理内容としては新規登録及び取得とし処理を行うクライアントを 10 クライアント並列にて行うものとする。

その際に秒間当たりの処理数を測定し各ケースでの処理回数を計測する。

(1) 登録データ

- Key データ：20byte のユニークな文字列
- Value データ：50byte のランダムな文字列
  データ長としては秘匿情報ということを前提に、氏名、住所、クレジットカード番号などを想定し 50byte とした。

(2) ストレージ

okuyama はメモリ及び、ディスクをストレージとして利用出来、切り替えが出来る機能を有している。本検証ではディスク利用による IO 等のオーバヘッド時間が発生することを考慮し、メモリを利用したストレージを用いて暗号化処理にフォーカス出来るように検証を行うものとする。

(3) 冗長化構成

NOSQL の最も代表的な特徴としてデータの冗長化管理がある。okuyama にもこの機能は搭載されており最大で 2 つの冗長化データを作成することが可能である。冗長化数を増やした場合データの保全性は向上するが処理数が増えるため処理能力は低下する。

本検証では実データとコピーを 1 つ作成する冗長化構成とし検証を行うものとする。

(4) 暗号化方式

暗号化箇所として通信経路及び、保存データの 2 箇所が存在する。それぞれ以下の暗号化方式とする。

- 通信経路暗号
  TCP/IP 経路上を Secure Socket Layer (以下 SSL)認証にて 128bit 暗号化し情報を送信する。これは okuyama のマスターノード/データノード間を暗号化するものとした。
- 保存データ暗号
  マスターノードからデータノードへ送信後データノード内で Value を AES 暗号化方式(128bit)にて暗号化し保存するものとした。取得時は復号化が行われマスターノードへ返却される。

(5) 実験環境

実験用として以下のスペックのサーバを 4 台利用した。

- CPU：Intel(R) Core(TM) i5-2400 CPU @ 3.10GHz
- メモリ：4GB
- HDD：200GB(7200rpm)
- ネットワーク：1Gbps

(6) 実験用 okuyama 構成

okuyama の構成として以下とした。

- データノード:2 台/4 ノード
- マスターノード:1 台/1 ノード
- クライアント:1 台/10 クライアント
- データ冗長化:1 データ 1 コピー

(7) 実験方法

データ登録テスト及び、指定したデータの取得テストを行う。

- データ登録テスト
  okuyama 内のデータが 0 件の状態より始め 1 クライアント当たり 100,000 件のデータを登録するものとし、10 クライアントを同時に実行し合計 1,000,000 件のデータが登録完了までの時間を計測し 1 秒間当たりの登録処理数を算出する。

- データ取得テスト
  データ登録テストにて登録した 1,000,000 件のデータを利用し 10 クライアンで同時にデータ取得を行い全てのデータが取得完了するまでの時間を計測し 1 秒間当たりの取得処理数を算出する。

これらのテストケースを通信経路のみ暗号化したモジュール、保存データのみ暗号化したモジュール、両方の暗号化を適応したモジュールの 3 パターンのモジュールにて登録、取得テストを行い処理能力の計測を行う。それらの結果と暗号化を行なっていないモジュールの結果を比較し処理能力の変化を調べるものとする。

4.3 実験結果

テストケースと処理結果をグラフ化したものが以下となる。

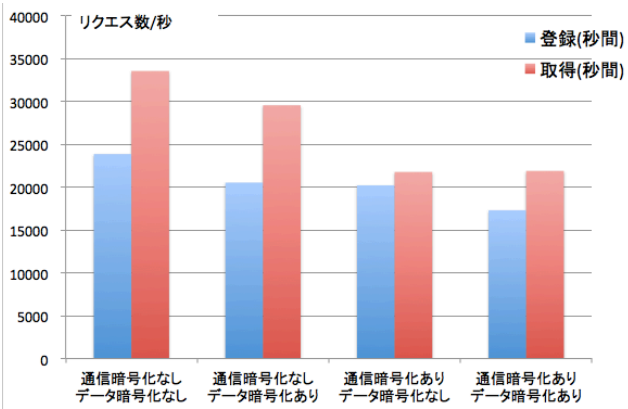

図 1

処理数の詳細は以下の表となる。

表 1

	登録処理数(秒間)	取得処理数(秒間)
通信暗号化なし データ暗号化なし	23,877	33,568

通信暗号化なし データ暗号化あり	20,539	29,570
通信暗号化あり データ暗号化なし	20,223	21,779
通信暗号化あり データ暗号化あり	17,313	21,899

(1) 結果考察

暗号化を実施していない状態では登録処理に関しては秒間約 24,000 回の登録処理が出来ており、取得に関しては秒間約 33,500 回の取得処理が出来ている。この結果を本環境での okuyama の基本処理性能としてそれぞれを比較した。

データ暗号化及び、通信経路暗号化のどちらを実施した場合であっても処理能力は通常時に比べて低下することが確認出来た。特に登録処理への影響が大きいことがわかった。

登録処理に関しては通常時に比べデータ暗号化時で 88%、経路暗号化時で 64%の処理能力となった。

取得処理に関しては、データ、通信経路の両方の暗号化時において処理能力の低下は低くおさえられており通常時の 80%以上の性能が出せていることがわかった。

両方の暗号化を行なった場合の処理能力は経路暗号化のみの場合よりもわずかに劣る程度であり、このことからデータ暗号化の負荷は低いものと考えられる。

両方の暗号化を行なった場合の処理能力の低下については、暗号化前に比べ登録処理時に 60%以上を維持できており、取得処理関しては 70%以上の処理能力を維持出来ていることがわかった。

データ暗号化に比べ経路暗号化時の処理能力低下が大きい理由として、経路暗号化の場合は必ず 1 サーバで構成されているマスターノードでの暗号化処理が行われるのに対して、データ暗号化時はデータノード上で暗号化処理が行われるため 2 サーバに処理が分散しており計算処理によるサーバリソースの消費が低く抑えられているものと考えられる。これらのことからマスターノードのノード数を増やすことで経路暗号化時に処理能力を向上させられる可能性が考えられる。

5. まとめ

本論文では、ビッグデータにおける秘匿データ管理に関して、暗号化が NOSQL の処理能力に及ぼす影響の検証及び考察を行った。

暗号化を実施したことにより一定の処理能力低下を確認することが出来た。しかし、暗号化実施後であっても処理能力が著しく低下するわけではなく登録、取得共に秒間 17,000 回以上の処理ができていることもあり今後秘匿データを含むビッグデータを管理する上で、NOSQL の活用は十分に検討の余地があると考えられる。

参考文献

[MySQL-参考 1]MySQL Developer Zone,http://dev.mysql.com/