情報漏えいに備えよ！
PostgreSQL で透過的暗号化を実現
－富士通の技術者に聞く！PostgreSQL の技術－

近年メディアで数多く取り上げられる情報漏えい問題やマイナンバー制度のスタートに伴い、企業のセキュリティに対する関心がますます高まってきている。セキュリティ事故の中でも社会的影響が大きく、問題となるのが情報漏えいである。情報漏えいを防ぐためのデータベースセキュリティ対策にはアクセス制御、権限分掌、DB 監査、モニタリングなどさまざまあるが、暗号化も重要な対策のひとつだ。
オープンソースデータベースである PostgreSQL にも暗号化機能が備わっているが、富士通はエンタープライズ利用に耐えうる高度な暗号化機能として「透過的データ暗号化（Transparent Data Encryption）」の実装に取り組んだ。ここでは、その暗号化機能について技術者がどのように機能を検討し開発したのか？暗号化において避けて通れない性能問題はどのように克服したのか？トランザクションログの暗号化で突きつけられた課題とは？その真相に迫る。

綱川 貴之 Takayuki Tsunakawa

専門分野：データベース

暗号化の有効性とは

ここ数年、メディアで情報漏えい問題を目にすることが増えてきました。情報漏えいに対する関心は高まる一方ですね。

綱川

そうですね。情報漏えいを含むセキュリティに対する関心が高まってきていると感じています。個人情報保護法や改正マイナンバー法など、情報に対する法律の整備が進んでいることも背景にあると思います。また法律だけでなく、例えばクレジットカード業界では「PCI DSS（注 1）」というカード会員のデータ保護を目的としたセキュリティ基準が整備されています。情報を守るために“何をすべきか”が具体的に定められていますし、クレジットカードを扱う業界も多いですから、いろいろな箇所で採用されてきています。その中でもデータを暗号化することが義務付けられています。他には、医療情報の保護を目的とした「HIPAA 法 / HITECH 法」で、医療情報のプライバシー保護とセキュリティ対策が義務付けられています。

注 1　Payment Card Industry Data Security Standards の略で American Express、Discover、JCB、MasterCard、VISA の 5 大カードブランドが 2004 年 12 月に共同で策定したクレジット業界におけるグローバルセキュリティ基準。2013 年 11 月に改訂版である V3.0 が発表されている。

情報漏えいの対策として暗号化は有効なのでしょうか？

綱川

はい。もちろん、暗号化だけで情報漏えいを全て防ぐことはできませんが、防衛手段の一つとして有効です。ただ、お客様は単純に情報を保護するという目的だけではなく、コンプライアンスという観点から暗号化に対する関心が高まっていると私たちは考えています。例えば、個人情報保護法では『高度な暗号化が施されていれば個人情報の提供者への連絡を省略できる』と定められています。つまり、盗まれた情報に適切な暗号化が施されていれば顧客に告知しなくてよいということになります。また、不正競争防止法では『利益を得るために営業秘密を盗んだ相手を罪に問える』と定められています。ここで重要なのは、適切な保護をしていないデジタル情報は保護対象と見なされないという点です。暗号化を施していれば適切な保護をしていると見なされるので、盗んだ相手を訴えることができます。

暗号化に対するお客様からのニーズの高まりを感じて、「透過的データ暗号化（Transparent Data Encryption 以後、TDE）」の開発に取り組んだのですね。PostgreSQL にも暗号化機能が用意されていますが、TDE とはどのような機能なのですか？

綱川

PostgreSQL には拡張モジュールによっていくつかの暗号化機能が用意されています。しかし、「自社製データベースに PostgreSQL を取り込んだ富士通の戦略とは？」で当社の千田が述べたようにエンタープライズ利用で求められる機能としては足りないと考えていました。

- 【関連特集】自社製データベースに PostgreSQL を取り込んだ富士通の戦略とは？

私たちが開発した TDE は、指定したテーブル空間内における全てのテーブルやインデックス、トランザクションログ（以後、WAL）、それにソート処理などで作られる一時ファイル内のユーザーデータが暗号化されます。暗号化の対象となるデータ型にも制限がないので、テーブル作成時に専用のデータ型を持つカラム（列）を用意するといった煩わしさもありませんさらに、バックアップデータも暗号化されます。データベースクラスタ（注 2）のファイルそのものが暗号化されていますので、pg_basebackup や tar などでコピーしたバックアップも暗号化されているということです。

注 2　PostgreSQL の独自用語であり、1 つのサーバーインスタンスで管理されるデータベースの集合体を意味する。冗長化されたシステムを意味する「クラスタ」とは異なる。

ストレージやバックアップメディア、データファイルのコピーによる盗難にあってもデータを見られる心配がない訳ですね。「データ暗号化（Data Encryption）」の前に「透過的（Transparent）」と付いていますが何か意味があるのですか？

綱川

はい。PostgreSQL に用意されている暗号化機能を使うには、利用者によるインストール作業などの事前準備に加え、アプリケーションを修正する必要があります。また、暗号化機能により少なからずオーバーヘッドが発生するため、レスポンス低下など性能への影響があります。TDE では、暗号化機能を利用する際の“アプリケーションの修正やカラムのデータ型変更を不要”にし“暗号化による性能への影響を極小化”しました。アプリケーションとの接続性や性能について、暗号化しない場合と同じように使える点が「透過的」を意味しています。

アプリケーション開発者が何も意識をする必要がない＝透過的ということですね。PostgreSQL の暗号化機能ではアプリケーションの修正が必要とのことですが、もう少し具体的に教えてください。

綱川

PostgreSQL で代表的な暗号化機能に、「pgcrypto」という contrib モジュールがあります。これはデフォルトではインストールされないため、利用者がインストールする必要があります。この pgcrypto では『どの列を暗号化するか』を利用者が選別し、その列に処理を行なう INSERT や UPDATE などの SQL 文に対して、データを暗号化・復号する関数を明示的に呼ぶようにアプリケーションを修正する必要があるのです。pgcrypto 以外では、暗号化専用のデータ型を指定した列を用意する必要があるものもあります。しかし、これでは既にデータがある状態だと暗号化したいカラムのデータ型を変更しなければなりません。いずれにしてもこれらの方法では、購入したパッケージアプリケーションや、元の開発者も設計書も存在しないような古いカスタムアプリケーションでは手を加えることが困難なため、実質的に暗号化機能を利用できないと考えています。

確かにその方法ですと暗号化機能を利用するためのハードルが高いと感じますね。TDE を使えばアプリケーションやデータベースに手を加えることなく、セキュリティを高めることができるのですね。

綱川

そのとおりです。TDE は暗号化アルゴリズムに「キー長 1286 ビット / 256 ビットの AES」を使っています。キー長に 256 ビットを使用する AES は、現在最も強固な暗号化方式の一つと言われています。また、暗号化機能をストリーミングレプリケーションと組み合わせて使うこともできます。スタンバイサーバー側のデータや、プライマリーサーバーとスタンバイサーバーで送受信する WAL も自動的に暗号化されます。さらに、pgcrypto など他の暗号化機能と比べて機能的に強化されています。その 1 つとして、先にも説明しましたとおり暗号化・復号を行なうにもアプリケーションの修正はもちろん、利用者の意識が不要です。PostgreSQL のサーバープロセスがストレージへの読み書き時にのみ自動的にバックグラウンドでデータを暗号化・復号しますので、アプリケーションが暗号化キーを管理する必要もありません。この点も強化の 1 つです。

暗号化キーの管理は？

暗号化キーの管理も必要ないのですか？

綱川

はい。pgcrypto では『このデータの暗号化キーはこれで』、『その暗号化キーはどこに格納する』といったことをアプリケーションが管理する必要があります。これを TDE では意識不要にしています。

暗号化キーは重要な要素ですよね？TDE ではどのような単位で存在し、誰が管理するのでしょうか？

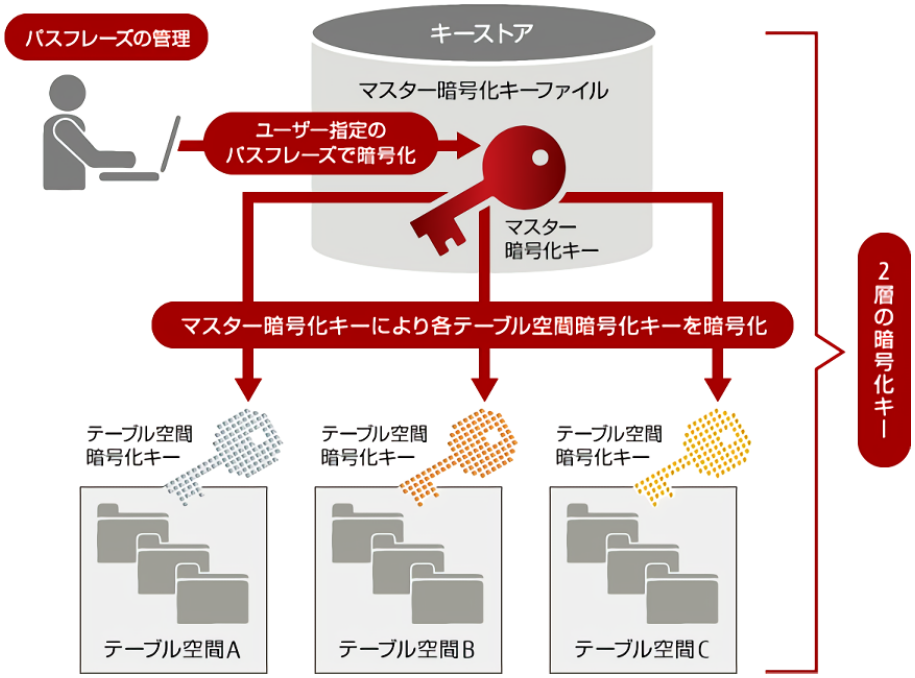
綱川

暗号化キーは 2 種類あり、1 つは「テーブル空間暗号化キー」というもので、テーブル空間ごとに 1 つずつあります。テーブル空間暗号化キーは、そのテーブル空間内のデータを暗号化・復号するために使われます。もう 1 つは「マスター暗号化キー」です。マスター暗号化キーは、すべてのテーブル空間暗号化キーを暗号化するために使われます。それぞれの暗号化キーはランダムなビット列で、暗号化キーごとに個別の「キーストア」というファイルに暗号化して格納されます。

暗号化キーは全て暗号化されてストレージ上で安全に保管されるということですね。マスター暗号化キーはどのような方法で暗号化されるのですか？

綱川

マスター暗号化キーは、利用者が指定するパスフレーズに基づいて暗号化されます。
最終的にセキュリティ管理者はマスター暗号化キーのパスフレーズを意識しておくだけでよいことになります。



図：TDE における暗号化キーの管理

煩雑な暗号化キーの管理から開放されるのですね。他にはどのような機能強化がありますか？

綱川

暗号化したテーブルに対してもインデックススキャンができるようになっています。pgcrypto の場合、暗号化した列のインデックスはレンジスキャンに使えませんが、TDE ではこのような制限はありません。暗号化した列に対する大小比較なども問題なく行な

えます。また、記憶域のオーバーヘッドも改善しています。pgcrypto の場合、暗号化のパディングのために暗号化データが平文より大きくなるのに対し、TDE では暗号化してもデータの大きさが変わらずストレージのオーバーヘッドがありません。

暗号化しても必要な記憶領域が大きくならないということですね？

綱川

そうです。ですから、システムに暗号化を取り入れるためにメモリ容量やストレージ容量を見直したり、必要に応じて追加したりという作業を行なう必要がありません。これも透過的である理由の 1 つです。暗号化・復号という機能を透過的に利用できるようにすることで、私たちが提唱する「運用性／使い勝手としてのオープン」を具現化しているのです。

暗号化を意識することなく安全なデータ管理を行なえるというのは、PostgreSQL をシステムで利用する企業も、そのシステムからサービスを受ける顧客にとっても双方にメリットがあることですね。

暗号化へのこだわり

暗号化というと性能への影響が少なからずあると思います。先程「暗号化による性能への影響を極小化した」と伺いましたが詳しく説明していただけますか？

綱川

データベースの暗号化は性能への影響が大きいのでは？と心配されるお客様も多いと思います。事実 pgcrypto は、SQL 文を実行するたびにデータベースキャッシュにあるデータを暗号化・復号するので、オーバーヘッドが大きくなります。この点、私たちの開発した TDE では、ストレージにデータを読み書きする時だけ暗号化・復号するので、データがデータベースキャッシュに載っている場合にはオーバーヘッドがありません。

しかし、必ず全てのデータがデータベースキャッシュに載る訳ではないですよね？ストレージへの読み書き時に発生するオーバーヘッドは大丈夫なのでしょうか？

綱川

確かに、TDE でもストレージへデータを読み書きする際に発生するオーバーヘッドは存在します。しかし、ここは CPU の専用命令を活用して暗号化・復号を高速化することによって解決しました。サーバーなどに用いられる Intel Xeon プロセッサーの 5600 番台（開発コード名：Westmere-EP）以降には、AES-NI（Advanced Encryption Standard New Instructions）と呼ばれる AES 暗号化・復号処理をハードウェアで高速化するためのアクセラレーション機能が搭載されています。AES は暗号強度が高い分、アルゴリズムが複雑で負荷が大きいと言われますが、暗号化機能のデファクトスタンダードとして CPU に専用命令が搭載されているのです。また、データベースのチューニングでも通常は可能な限りキャッシュヒット率を上げるのが碇石です。これらのことから、オーバーヘッドを気にせずアプリケーションで扱う全てのデータを暗号化できるようになっています。

性能への影響は具体的にどのくらいになるのでしょうか？

綱川

実際に pgbench を用いた更新の多い OLTP ベンチマークで計測したところ、暗号化によるオーバーヘッドは 3%未満でした。一方、ディスクへの読み書きが多いバッチ処理とロードでは、暗号化のオーバーヘッドがやや目立ちます。それでも、AES-NI があると暗号化によるオーバーヘッドは 10%未満に抑えられます。

性能のオーバーヘッドを極力無くすためにさまざまな努力を行なっているのですね。ここまで暗号化にこだわるのには何があるのでしょうか？

綱川

そもそも「PostgreSQL に TDE を実装してみたい」という技術者としての思いがありました。TDE は商用データベースでもメジャーな製品にしか実装されておらず、オープンソースデータベースでは実装例がほとんどありません。そこに“やりがい”を感じました。

TDE の実装に強い思いがあったのですね。開発の際に苦労もあったのではないでしょうか？

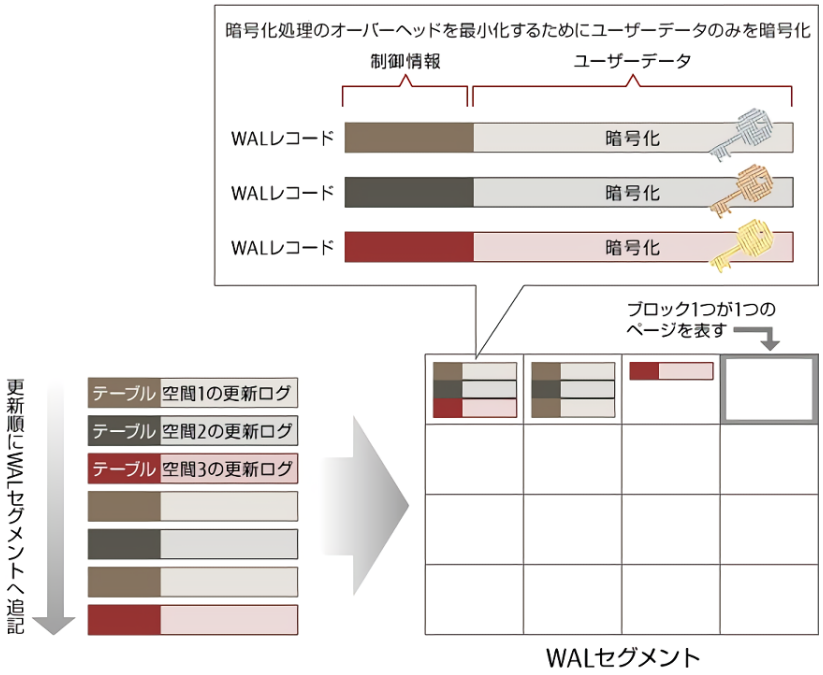
綱川

WAL の暗号化はとても苦労しました。データベースを暗号化するロジックは比較的単純なのです。なぜなら、ページ単位でストレージから読み書きするときに、暗号化・復号するだけでよいからです。テーブル空間単位に暗号化の有無を指定するので、中身を意識することなくページ全体を暗号化・復号すればよいのです。でも WAL はそうはいきません。ページ単位でストレージから読み書きするという点では、WAL もデータベースと同じです。しかし WAL の各ページには、異なるテーブル空間のデータに対する更新ログが混在する可能性があります。そのため、どの暗号化キーでページを暗号化・復号するかを決めることができないのです。

WAL は複雑なのですね。どのように解決されたのですか？

綱川

正攻法ですが、WAL レコード毎に暗号化しています。1 つの WAL レコードに対し、更新対象となるデータが含まれているテーブル空間用の暗号化キーで暗号化します。さらに、暗号化処理のオーバーヘッドを最小化するために、WAL レコード内のユーザーデータのみを暗号化するよう工夫しました。大変だったのは、多数ある WAL レコードの“どの部分がユーザーデータか？”を見極めて、その暗号化処理と復号処理を 1 つ 1 つ作り込んでいったことです。そういう努力をしたからこそ低いオーバーヘッドを実現できたと思います。



図：WAL の暗号化

クラウドへの発展

1 つ 1 つを作り込む・・・ですか。オーバーヘッドを最小限に抑える裏には地道な努力があったのですね。ところで、情報の預け先として「クラウド」がますます増えていくと思いますが、クラウド上での情報漏えい対策について開発当初から何か念頭におかれていましたか？

綱川

はい、ありました。TDE を開発した当時は、データベースやストレージの PaaS で暗号化が提供され始めていました。この状況を見て私は、『いずれどの PaaS / SaaS も自動的に全ての格納データを暗号化するのが当たり前になるだろう』と考えました。事実、商用データベースを利用した DBaaS では、既に Oracle Database Cloud Service や Amazon RDS for Oracle が、TDE を利用して全データを自動的に暗号化しています。ストレージサービスでは、Amazon S3 が API でデータを書き込む時に、データ単位で暗号化の有無を指定できるようになっています。さらに Google Cloud Storage では、暗号化の有無を指定することなく自動的に全てのデータが暗号化されます。こうした中、オープンソースデータベースをクラウドで利用したいというニーズは必ず来るだろうと思い、このため『できるだけ早く TDE をサポートした PostgreSQL を DBaaS で提供しなくては』と考えたのです。

ただ TDE に対応するのではなく、どのように活用されるかも視野に入れて実装したのですね。

綱川

そうです。現在では PostgreSQL の人気が高まってきたことに伴いクラウドベンダー各社が DBaaS で提供しているのですが、当社の DBaaS だけが唯一 TDE を提供しているのです。

常にお客様のニーズを先読みして提供しているのですね。今後強化したい点はありますか？

綱川

いくつかありますが 1 つは、「HSM（Hardware Security Module）」による暗号化キーの堅牢な管理です。HSM とは鍵（暗号化キー）を守る金庫のような機能を持ったハードウェアです。高度なセキュリティが要求される金融機関などでは、HSM で暗号化キーを管理しているところもあります。先程説明しましたが、TDE では暗号化キーをキーストアというファイルに保存しています。もちろんキーストアは暗号化されているので安全ですが、攻撃者の手が届くところに“鍵”があるのは不安だというお客様もいらっしゃいます。この点、HSM はデータベースから独立したハードウェアであり、かつ耐タンパー性に優れており、暗号化キーを一切外に取り出さずに暗号化・復号ができるので安心というわけです。

専用のハードウェアによって更なる強固なセキュリティを実現するのですね。しかしハードウェアですと利用に制限があるのではないでしょうか？

綱川

HSM は 1 台数百万円と高価なため簡単に導入という訳にはいきませんし、ハードウェア装置のためパブリッククラウドでは使えませんでした。しかし、Amazon が AWS クラウド上で利用できる HSM を「AWS CloudHSM」というサービスとして提供を始めましたし、2014 年には新しい暗号化キー管理サービス「AWS Key Management Service」を発表したことから見ても、クラウドでの暗号化キー管理が一般的になってきそうです。これらサービスと TDE の連携が実現できたら、パブリッククラウドでも TDE を利用しやすくなるかなと思っています。

より高信頼な暗号化キーの管理がクラウドにまで発展するかもしれないのですね。

綱川

そうなるかもしれません。もう一つは「TPM（Trusted Platform Module）」による暗号化キーの堅牢な管理です。TPM というのは、コンピューターに搭載されているセキュリティチップです。身近なところでは、Microsoft Windows Vista 以降に搭載されている「BitLocker ドライブ暗号化機能」が TPM を利用したストレージの暗号化を行なえます。TPM はコンピューターのマザーボードなどに直付けされており、チップ内での暗号化・復号やデジタル署名の生成・検証などができ、さらに少量のデータを保存できます。この TPM にマスター暗号化キーを格納できたら安全な管理ができるのではと思っています。ただし、TPM はコンピューター

に搭載されたセキュリティチップに依存するので、仮想環境ではどうするのかという問題があります。しかしこの問題を解決するために、「仮想 TPM」と呼ばれるものが開発されているようですので、とても期待しています。

今後は暗号化キーの管理をいかに強固なものにしていくのか？というのが文字通り“鍵”となるのですね！TDE の発展がますます楽しみです。

2015 年 10 月 30 日