
بسم الله الرحمن الرحيم

دانشگاه پیام نور تهران

واحد شهر ری

سمینار، سمینار تحقیق و تتبع نظری

عنوان:

مدل‌های ایمن و کارآمد برای دریافت و پردازش اطلاعات از پایگاه های رمزگذاری شده در فضای ابری

استاد راهنما:

جناب آقای‌دکتر سید علی رضوی ابراهیمی

نگارش:

مجید لطفی

تابستان ۱۴۰۰

چکیده

امروزه بحث حفظ و امنیت بانک‌های اطلاعاتی با داشتن اطلاعات حساس و استراتژیک در سطح یک شرکت و حتی بعضا در سطح کشورها، موردبررسی و تحقیق پژوهشگران بخش نرم‌افزار و داده هست.

بخشی از این امنیت جلوگیری از آسیب و افشای اطلاعات از سمت هکرها و گروه های خرابکاری بوده و بخش نگران کننده دیگر ، افشای اطلاعات توسط کاربران و ارائه‌دهندگان سرویس های ابری هست زیرا گروه دوم از لایه‌های سخت‌افزاری و نرم‌افزاری رایج برای امنیت، گذر کرده و دسترسی آزاد به منابع اطلاعاتی دارند.

سرویس دهندگان ابری و کاربران یک سیستم ، به علت داشتن دسترسی آزاد به اطلاعات، براحتی می‌توانند به اطلاعات طبقه بندی شده و حساس یک بانک اطلاعاتی دسترسی پیدا نمایند و تبعات سنگینی را برای صاحبان بانک اطلاعاتی ایجاد کنند. بهترین و شاید تنهاترین ایده ای که به ذهن می رسد رمزگذاری اطلاعات هست تا این اطلاعات تنها پس از کنترل دسترسی ها داخل نرم‌افزار، به نمایش دربیایند.

اما رمزگذاری و به تبع آن رمزگشایی نیاز به محاسبات سنگین و زمان مشغولی سرور دارند و اگر نتوانیم با روش صحیح و مناسب این روش را جلو ببریم، در حجم اطلاعات بالا و پس از مدتی ، استفاده از سامانه سخت و حتی غیرمقدور می‌شود.

آشنایی با انواع روش‌های رمزگذاری ، میزان بار و هزینه اجرا آن و تعیین روش صحیح برای رمزگذاری، به ما کمک می کند به یک راهکار و روش مناسب برای حفظ محرمانگی اطلاعات برسیم.

کلمات کلیدی: بانک اطلاعاتی ، رمزگذاری ، رمزگشایی ، سرور ابری ، سرویس‌دهنده ابری

فهرست مطالب

فصل پنجم: آزمایش و ارزیابی

فصل ششم: بحث و پیشنهادها

فصل اول

مقدمه

امروزه اکثر شرکت‌ها به‌موازات اشتراک فایل در شبکه‌های اجتماعی و فضای ابری ، به سمت استفاده از فضای ابری جهت نگهداری اطلاعات و دیتای خود رفته‌اند زیرا از سوئی می‌توان با اجاره سرویس‌های آماده ، هزینه تهیه سرور را کاهش داد و از طرف دیگر امکان دسترسی هر زمان از دیتاها را فراهم نمود .

به‌موازات گسترش استفاده از این بستر ، آسیب‌ها و مشکلات آن نیز مطرح‌شده است و باید بتوان این سرویس را در امنیت و با کمترین مشکل اجرا نمود . این مشکلات گاها مربوط به حملات به سرور و سیستم‌های عامل هست با قصد از کار انداختن سرویس‌ها و گاها مربوط به نفوذ غیرمجاز به دیتاها هست پس جدا از بحث‌های امنیت شبکه و سیستم‌عامل ، خود دیتا نیز حفاظت شود.

حفاظت از داده خارج از روش‌های نرم‌افزاری مانند تعریف نقش‌ها در پایگاه داده و کنترل این نقش‌ها ، باید با روش‌هایی مانند رمزگذاری حفاظت شود زیرا سرویس‌دهنده‌های و مالکان سرورهای ابری امکان دسترسی به دیتای مشتریان خود را داشته می‌توانند از این اطلاعات حساس استفاده غیرمجاز نمایند.

رمزگذاری در دو بخش ساختار و دیتا قابل‌اجرا هست و اجرا هم‌زمان هر دو بخش می‌تواند بالاترین امنیت را ایجاد نماید زیرا با فرض امکان و یا افشای دسترسی به پایگاه داده ، داشتن نام جداول و یا ستون‌ها متداول باعث آسیب‌پذیری بیشتر اطلاعات می‌شود برای مثال اگر جدولی با نام Banks داشته باشیم قطعاً بیان‌کننده نام بانک‌ها هست درحالی‌که اگر همین جدول به‌صورت T۱۲ باشد ، امکان افشای اطلاعات کاهش و حتی غیرممکن می‌شود.

۱-۱ تعریف مسئله و سؤال

با توجه به آنچه تا اینجا بیان شد، مسئله اصلی ما حفاظت داده‌ها از بالاترین سطح که سرویس‌دهنده ابری هست و دسترسی کامل به اطلاعات دارد تا پایین‌ترین سطح که افراد غیرمجاز و هکرها هست ، تعریف می‌شود. این حفاظت به‌گونه‌ای هست که حتی اگر شخص بتواند دیتا را در به‌صورت مستقیم از جداول مشاهده نماید، باز با خطر افشای اطلاعات مواجه نشویم.

سؤالاتی که تا پایان این تحقیق پاسخ داده می‌شود را این‌گونه بیان می‌کنیم.

۱- انواع روش‌های رمزگذاری قابل‌اجرا در پایگاه‌های داده کدام‌اند؟

۲- انواع رویکردهای رمزگذاری در پایگاه‌های داده کدام‌اند؟

۳- بررسی فرآیندها و عملیات پایگاه داده رمزگذاری شده چگونه هست؟

۴- بررسی و مقایسه روش‌ها و رویکردهای رمزگذاری ازنظر هزینه اجرا و نگهداری چگونه‌اند؟

۱-۲ ضرورت تحقیق

تا اینجا با مسائل و مشکلات یک پایگاه داده ازنظر افشای اطلاعات آشنا شدیم و رمزگذاری به‌عنوان راهکار آن معرفی گردید. حال باید این موضوع رو در نظر بگیریم که صرف رمزگذاری و ایجاد لایه‌های رمزگذاری مسئله ما نیست زیرا هرچقدر لایه‌های رمزگذاری اضافه شود ، به همان اندازه لایه‌های به دست آوردن دیتاها و هزینه دریافت اطلاعات برای خودمان بالاتر می‌رود زیرا این عبارات رمزگذاری شده باید برای پردازش و استفاده مجدد ، رمزگشایی شوند و ما باید سعی کنیم رمزگذاری را بر روی دیتای حساس لحاظ نماییم و نه بر روی تمام پایگاه داده.

ایده و نگرش دیگری که در این پایان‌نامه بر آن تأکید شده، کاهش بار محاسبات رمزگذاری و رمزگشایی هست، به این صورت که در هنگام دریافت اطلاعات در یک پرس‌وجو، نیاز به رمزگشایی اطلاعات نامرتبط نباشد و سعی شود تا حدودی از میزان اطلاعات واکشی جهت رمزگشایی کاسته شود.

۱-۳ اهداف

با آنچه تا اینجا در خصوص افشای اطلاعات و ضرورت حفظ آن بیان شد ، ما در این تحقیق چند هدف را موردبررسی قرار می‌دهیم و سعی می‌کنیم در انتها با جمع‌بندی روش و رویکردها رایج در این خصوص بپردازیم . این اهداف عبارت‌اند از :

۱- آشنایی با روش‌ها رمزگذاری بدون در نظر گرفتن پایگاه داده

۲- بررسی و اجرا روش رمزگذاری مؤثر و رایج در پایگاه داده

۳- آشنایی با مدل‌ها و اصول قابل‌اجرا در رمزگذاری پایگاه داده

۴- مقایسه مدل‌ها و انتخاب مدل مناسب

۱-۴ جمع‌بندی

در این تحقیق به‌ضرورت حفظ اطلاعات پایگاه داده از افشای آن و رمزگذاری به‌عنوان بهترین روش جهت انجام آن معرفی می‌گردد. این موضوع چنانچه قبلاً نیز اشاره شد، صرفاً برای جلوگیری از حملات مخرب و غیرمجاز نیست و بررسی‌های این تحقیق در خصوص کاربران و اشخاصی هست که به پایگاه داده ، دسترسی مجاز دارند.

ابتداها به روش‌های رمزگذاری ، اصول و قواعد رایج در آن و سپس به رویکردهای رایج در ایجاد یک پایگاه رمزگذاری شده می‌پردازیم و درنهایت مدل‌ها و روش‌های معرفی‌شده را مقایسه تا بهترین مدل و روش را انتخاب نماییم.

فصل دوم

مقدمه

رمزگذاری روند رمز کردن پیام‌ها یا اطلاعات اســـت به‌گونه‌ای که تنها افراد مجاز قادر به خواندن آن باشـــند. پیام یا اطلاعات با اســـتفاده از یک الگوریتم، رمزگذاری شـــده و علائم رمزی به وجود می‌آید که فقط در صـــورت رمزگشـــایی قابل‌خواندن هســـتند. در رمزگذاری معمولاً یک کلید رمزگذاری شـــبه تصـــادفی تولیدشده توسط یک الگوریتم، به کار گرفته می‌شـــود. اگرچه شـــاید رمزگشـــایی پیام بدون در اختیار داشتن کلید ممکن باشد، اما در یک رمزگذاری خوب، منابع محاســـباتی زیادی برای این کار لازم اســـت. یک گیرنده مجاز به‌راحتی می‌تواند پیام را با کلید تدارک دیده‌شـــده توسط صادرکننده پیام، رمزگشایی کند اما گیرنده غیرمجاز نمی‌تواند.

هدف از رمزگذاری اطمینان از این اســـت که فقط کســـانی که مجاز به دستیابی اطلاعات هستند، قادر به خواندن آن و استفاده از کلید رمزگذاری باشند.

۲-۱ روش‌های رمزگذاری :

رمز عبارت اســـت از تبدیل کاراکتر به کاراکتر یا بیت به بیت بدون آن‌که به محتویات زبان‌شـــناختی (ادبیات) آن پیام توجه شـــود. رمزگذاری دیجیتال، متن قابل‌خواندن (که اصـــطلاحاً به آن plaintext می‌گویند) را می‌گیرد و آن را به متنی غیرقـــابـــل تشـــخیص تبـــدیـــل می‌نمـــایـــد و اصـــطلاحاً آن را رمزگذاری می‌کند. البته برای انجام چنین کاری، الگوریتم‌های بسیار پیچیده و قدرتمند وجود دارد.

این الگوریتم‌هـــا از متغیرهـــایی بـــه نام کلید بهره می‌برند که پیچیدگی رمزگذاری را چندین برابر می‌کنند. این کلیدها به‌طور تصـــادفی تولید می‌شـــوند و منحصـــربه‌فرد هســـتند. یعنی اگر یک هکر بخواهد بانفوذ به یک بانک داده اطلاعات کارت اعتباری کاربران را بدزدد، نه‌تنها باید از الگوریتم اســـتفاده‌شـــده خبر داشـــته باشـــد بلکه باید بداند کدام کلید اســـتفاده‌شـــده اســـت. چنین کاری به‌هیچ‌عنوان کار راحتی نیســـت و به همین خاطر رمزگذاری‌های دیجیتال در برابر حملات جستجوی فراگیر (brute force) مقاومت بسیار بالایی دارند.

دو روش برای رمزگذاری قابل‌اجرا هست : روش متقارن و روش نامتقارن.

۲-۲ رمزگذاری متقارن :

در روش متقارن تنها یک کلید برای رمزگذاری استفاده می‌شود و فرستنده و گیرنده اطلاعات از همان کلید مشترک برای رمزگذاری و رمزگشایی داده استفاده می‌کنند. تمام رمزنگاری‌های کلاسیک از نوع متقارن هستند و تا قبل از دهه ۷۰ تنها نوع رمزنگاری به‌حساب می‌آمد.

برخی از الگوریتم‌های متداول در رمزگذاری بر مبنای روش متقارن به‌صورت ذیل هست:

۱- Order-Preserving Encryption: رمزگذاری با حفظ ارزش عبارت رمزگذاری نسبت به اصل عبارت مثلاً اگر رمزگذاری دو عبارت A, B بشود E۱, E۲ و A>B باشد قطعاً E۱>E۲ و برعکس و این مقایسه بدون رمزگشایی معتبر هست و مناسب برای دیتای عددی و این مزیت خود منجر به ضعف شده و می‌توان توسط استنباط تا حدودی به اطلاعات اصلی رسید.

۲- Deterministic Encryption: یا همان DES الگوریتم رمزگذاری‌ای هست که توسط یک کلید ، همواره یک عبارت با به متن مشخصی رمز می‌نماید و متأسفانه ارزش عبارت اصلی را حفظ نمی‌نماید و در صورت افشای کلید ، به‌راحتی می‌توان سایر عبارات را رمزگشایی کرد! با شکسته شدن الگوریتمDES این استاندارد در سال ۱۹۹۸ تمدید نشد.

۳- Advanced Encryption Standard: یک الگوریتم رمزنگاری بلوک جایگزینی شبکه (SPN) است و جایگزین DES شد. دقیقاً به این معنی است که الگوریتم یک بلوک از متن ساده را می‌گیرد و مقادیر متناوب جایگزینی را به آن اعمال می‌کند. AES شامل سه بلوک از رمزهای AES-۱۲۸، AES-۱۹۲ و AES-۲۵۶ است. هر رمز، عملیات رمزنگاری و رمزگشایی را بر روی بلوک‌های ۱۲۸ بیتی داده، با استفاده از کلیدهای ۱۲۸، ۱۹۲ و ۲۵۶ بیتی خود انجام می‌دهد و محبوبیت بالایی دارد.

۴- Blowfish: یک رمزنگاری قطعه‌ای سریع محسوب می‌شود که داده‌ها در قطعه‌های ۸ بایتی رمز می‌کند ، به‌جز هنگامی‌که کلیدها عوض می‌شوند. هر کلید جدید نیازمند پیش-پردازشی است که معادل رمزگذاری یک فایل متنی تقریباً ۴کیلوبایتی است، که نسبت به دیگر رمزنگاری‌های قطعه‌ای بسیار کند است.

۲-۳ رمزگذاری نامتقارن :

در روش نامتقارن از دو کلید استفاده می‌نماید ، یک کلید عمومی و یک کلید خصوصی که کلید عمومی یک عبارت رمزگذاری شده جهت رمزگذاری در سطح کاربران هست که خود کلید عمومی توسط کلید خصوصی رمزگذاری شده است. این سطح از رمزگذاری منجر به کاهش سرعت و افزایش هزینه استفاده از سیستم می‌شود.

برخی از الگوریتم‌های متداول در رمزگذاری بر مبنای روش نامتقارن به‌صورت ذیل هست:

۱- Homomorphic Encryption: نوعی رمزگـذاری برای انجـام عملیـات جمع و ضـــرب بدون باز کردن عبارت رمزگذاری شـده و یا داشـتن کلید و مناسـب دیتای عددی و دارای قابلیت اجرای دستورات SQL بر روی عبارات رمزگذاری.

۲- Randomized Encryption: این رمزگـذاری یک عبارات را هر بار به عبارت رمزگذاری متفاوتی تبدیل می‌نماید و بالاترین سـطح امنیت هسـت اما به علت نوع رمزگذاری قابلیت اجرای دسـتورات SQL بر روی عبارات رمزگذاری را ندارد.

۳- Onion Encryption: در این رمزگذاری از لایه‌های مختلف با متدهای مختلف رمزگذاری استفاده می‌شود به صورتی که لایه داخلی با متد رمزگذاری امنیت پایین و لایه بیرونی دارای رمزگذاری با امنیت بالا و مشکل آن کند شدن پردازش پرس‌وجو و طولانی شدن عبارات رمزگذاری نسبت به عبارت اصلی هست.

۴- RSA: در این‌چنین سـیسـتم‌های رمزنگاری، کلید رمزگذاری عمومی اسـت و از کلید رمزگشـایی که مخفی اسـت، جداسـت. هرکسـی می‌تواند از این کلید عمومی برای رمزگذاری یک پیام استفاده کند، اما تنها کسی که آن دو عدد اولی که کلید بر اسـاس آن‌ها سـاخته‌شـده را می‌داند، قادر به رمزگشایی پیام است. شکستن رمزگذاری RSA به مسـئله‌ی RSA معروف اسـت. تاکنون هیچ روشـی برای شـکست دادن این سیستم(در صـورت اسـتفاده‌ی کلید به‌اندازه‌ی کافی بزرگ) منتشـرنشـده اسـت. RSA به‌صورت نسبی، الگوریتم کندی است و به همین علت، کمتر برای رمزگذاری مستقیم اطلاعات کاربر استفاده می‌شود.

۵- ElGamal: خصـوصیت این الگوریتم به‌گونه‌ای است که در هر مرحله از فرایند رمزنگاری یک کلید تصادفی k تولیـد می‌شـــود مقـدار این کلیـد کاملاً تصـــادفی بوده و در هر مرحله از اجرای فرایند رمزنگاری کلید تولیدشـده با کلید قبلی متفاوت خواهد بود. این خصـوصـیت موجب می‌شود در دو مرحله متفاوت خروجی متفاوت تولید شـــود, که این خصـــوصـــیت از ویژگی‌های تولید کلید K در هر مرحله از فرایند رمزنگاری هست.فرایند تولید تصادفی کلید K موجب می‌شود حدس زدن کلید K از روی مقدار C۱ امکان‌پذیر نباشد.

بـا توجـه به تنوع این الگوریتم‌ها ما باید با در نظر گرفتن هزینه اجرا ، ســـرعت و میزان حســـاســـیت اطلاعات ، در بخش‌های مختلف پایگاه داده و سیستم نرم‌افزاری ، الگوریتم کاربردی‌تر و مناسب‌تر را انتخاب نماییم .

فصل سوم

مقدمه

پایگاه داده با توجه به ثبت اطلاعات به‌صورت متن ساده و داشتن اطلاعات حساس ، نیازمند بررسی و مطالعه در جهت حفظ اطلاعات از افشای آن توسط کاربران و صاحبان سرورها دارد. البته نسخه‌های بالاتر بانک‌های اطلاعاتی دارای دستورات جهت رمزگذاری و رمزگشایی هست اما ما به دنبال رویکرد و الگوریتم‌هایی هستیم تا بتوانیم بر روی تمامی پایگاه‌های داده اجرا نماییم.

امروزه نرم‌افزارهایی به‌صورت سرویس‌دهنده بانک‌های اطلاعاتی که نقش واسطه بین کاربر و بانک اطلاعاتی دارند، با ایجاد قابلیت مدیریت کدگذاری بانک اطلاعاتی توانسته‌اند این موضوع را مدیریت کنند اما اگر بخواهیم بانک اطلاعاتی خود را با یک استراتژی قابل تنظیم ، از افشای اطلاعات محافظت نماییم ، نیازمند آشنایی با رویکردهای فعلی و تحقیقات انجام‌شده در این زمینه‌داریم.

۳-۱ رویکردهای فعلی رمزگذاری در پایگاه داده :

یکی از روش‌های رمزگذاری یک پایگاه داده ، تبدیل رکوردها به بلوک رمزگذاری هست به این صورت که هر ردیف از اطلاعات با هر تعداد از ستون ، تبدیل به یک دیتا رمزگذاری شده در جدول رمزگذاری شود اما این روش امکان اجرای دستورات SQL را بر روی دیتاهای رمزگذاری شده را از بین می‌برد و حتماً باید دیتا رمزگشایی و دستور اجرا شود و این در اطلاعات با رکوردهای بالا منجر به هزینه و زمان بالا می‌شود.

برای حل این مشکل محققان روشی را در نظر گرفتند به این صورت که ستونی دیگر به این جداول رمزگذاری شده اضافه گردد که در آن دسته‌بندی ردیف‌ها مشخص باشد و در زمان دریافت اطلاعات ، با توجه به دسته‌بندی موردنظر ، ما تعداد رکورد کمتری را دریافت و برای رمزگشایی پردازش نماییم. این دسته‌بندی‌ها درجایی به‌صورت رمزگذاری شده نگهداری می‌شود.

البته این روش هم امکان افشای این دسته‌بندی‌ها و سنگین‌تر شدن پردازش‌ها را دارد.

در اینجا بعضی سیستم‌های رمزگذاری در پایگاه داده معرفی می‌گردد که دارای امنیت و کارکرد مناسبی می‌باشند.

۳-۲ معرفی CryptDB :

این سیستم به‌عنوان اولین سیستم عملی برای اجرای دستورات SQL از طریق پایگاه‌های داده رمزگذاری شده توسعه داد شد که توسط Onion Encryption طراحی گردیده است و هر داده توسط بیش از یک الگوریتم رمزنگاری ، رمزگذاری می‌شود که در آن رمزهای رمزگذاری بیرونی تولیدشده توسط یک الگوریتم رمزگذاری تصادفی انجام می‌شود. (نمایی از نوع رمزگذاری و پردازش‌ها در این سیستم)

این نوع رمزگذاری هزینه محاسبات رمزگذاری و رمزگشایی را بالا می‌برد البته این موضوع توسط MONOMI حل‌شده است به این صورت که اجرای دستورات به دو بخش تقسیم‌شده‌اند : دستوراتی که بر روی اطلاعات رمزگذاری شده قابل‌اجرا هست و دستوراتی که بر روی اطلاعات رمزگشایی‌شده در سمت کاربر اجرا می‌شود. همچنین برای بهبود سرعت رمزگذاری ، از الگوریتم AES-NI بجای AES استفاده‌شده است که از روش‌های بهتری برای اجرای رمزگذاری AES استفاده می‌نماید.

۳-۳ معرفی FHOPE :

این سیستم از الگوریتم homomorphic باقابلیت order-preserving برای اجرای دستورات SQL در اطلاعات عددی استفاده می‌کند و همان‌طور که قبلاً گفته شد قابلیت order-preserving به ما این امکان را می‌دهد تا بدون رمزگشایی بتوانیم بر روی‌داده‌های رمزگذاری شده کارکنیم. البته این سیستم بر روی تعداد رکورد کم آزمون شده است و باید در دیتاهای بزرگ‌تر آزمون شود.

۳-٤ معرفی P-McDb :

همان‌طور که گفته شد قابلیت اجرای دستورات بر روی دیتاهای رمزگذاری شده بعضاً منجر به افشای اطلاعات از طریق استنتاج می‌شود. این سیستم برای جلوگیری از این حمله ، توسط دو سرور که یکی برای ذخیره‌سازی و جستجوی اطلاعات و دیگری برای تصادفی سازی و تغییر شکل پایگاه داده که این خود منجر به‌کندی سیستم می‌شود. این سیستم نیز بجای جستجوی کلی بر روی دیتاهای رمزگذاری شده ، از جستجوی جزئی استفاده می‌نماید. همچنین در لغو دسترسی یک کاربر به بانک اطلاعاتی برخلاف سایر طرح‌های SSE (Server Sent Events)، نیاز نیست تمامی اطلاعات مجدد رمزگذاری شود بلکه به‌تمامی CSP (Content Security Policy) اطلاع داده می‌شود و دسترسی کاربر بسته می‌شود و حتی اگر کاربر کلید رمزگشایی و دسترسی به یکی از CSP ها را داشته باشد نیز امکان دسترسی به اطلاعات را نخواهد داشت.

۳-۵ معرفی SDB :

یکی از راه‌های افزایش سرعت و کارایی سیستم در مواجهه با رمزگذاری داده‌ها ، تقسیم داده به حساس و غیر حساس و رمزگذاری بر روی دیتاهای حساس هست و این سیستم از این روش استفاده می‌نماید. داده حساس نیز به دو بخش اشتراکی تقسیم می‌شود، مالک داده‌ها بخش اول اشتراک و بخش دوم اشتراک را سرویس‌دهنده ابری نگهداری می‌کند. در این حالت سرویس‌دهنده ابری تا زمانی که مالک داده بخش اشتراکی خود را اجرا ننماید ، هیچ دسترسی‌ای به داده‌ها نخواهد داشت.

۳-٦ معرفی CASB :

این سیستم توسط SafeBox تحت عنوان Access Security Broker طراحی شد و به‌صورت نرم‌افزار یا سخت‌افزار داخلی که به‌عنوان واسطه بین کاربران و ارائه‌دهندگان خدمات ابری عمل می‌کند اجراشده است. این ایده امکان جستجو و اشتراک داده‌ها و حتی فایل‌ها با حفظ امنیت آن را فراهم می‌کند. یکی از امکانات این سیستم جستجوی کلمات کلیدی بر روی اطلاعات رمزگذاری شده را فراهم می‌کند.

۳-۷ سایر تحقیقات :

برخی از محققان از تکنیکی به نام "Bucketization" استفاده کردند که در آن ردیف‌ها داده به بیش از یک منبع متصل می‌شود. این فنّاوری امکان استفاده از پایگاه داده به‌عنوان یک سرویس‌دهنده برای اجرای دستورات بر روی‌داده‌های رمزگذاری شده را فراهم می‌کند (Database as service) .

برخی از محققان نیز از فضای ابری ترکیبی برای تقسیم داده‌ها به دو بخش حساس و غیر حساس استفاده نموده که بخش غیر حساس را در فضای عمومی و بخش حساس را فضای خصوصی کاربران قرار می‌دهند. یکی از اشکالات این روش به این صورت هست که کاربران اکثراً دارای اطلاعات حساس می‌باشند و از سوی دیگر یکپارچه‌سازی بین دو بخش مجزا دارای سختی‌های خود هست.

گروهی دیگر از تکنیکی برای جلوگیری از افشای اطلاعات توسط ارائه‌دهندگان خدمات ابری غیرقابل‌اعتماد و مشکوک پیشنهاد کرده‌اند. این تکنیک بر اساس تقسیم‌بندی عمودی است، که در آن هر ستون رمزگذاری شده حساس به یک سرور ابر متفاوت برون‌سپاری می‌شود. البته این روش دارای تأخیر برای ارتباط بین بخش‌های مختلف هست خصوصاً اگر دستور SQL پیچیده باشد! واسط این کار یک پراکسی هست که عملیات رمزگذاری و رمزگشایی را انجام می‌دهد.

با توجه به اینکه تقسیم و برون‌سپاری داده باعث ایجاد حساسیت‌ها و نگرانی‌هایی نسبت به افشای اطلاعات می‌شود ، گروهی از محققین یک سیستم سخت‌افزاری / نرم‌افزاری را برای رفع مشکل نشت محرمانه بودن در پایگاه‌های داده برون‌سپاری شده معرفی کرده‌اند. روش کار به این صورت هست که کاربر کارت هوشمند میانجی را که در کنار آن وصل است نگهداری و کنترل می‌کند .این کارت هوشمند مسئول رمزگذاری داده‌ها قبل از قراردادن آن‌ها در پایگاه داده و رمزگشایی داده‌ها قبل از ارسال آن‌ها به کاربر است .عیب عمده این روش این است که کاربر با ظرفیت کارت هوشمند محدود می‌شود و نمی‌تواند از فضای ذخیره‌سازی ارائه‌شده توسط سرویس‌های ابری بهره‌مند شود.

فصل چهارم

مقدمه

تا اینجا با انواع رمزگذاری و سیستم‌های رایج رمزگذاری آشنا شدیم و این رمزگذاری را در پایگاه داده موردبررسی قراردادیم اما در یک بانک اطلاعاتی صرف رمزگذاری اطلاعات تنها می‌تواند در ذخیره اطلاعات مؤثر باشد و ما در یک پایگاه اطلاعاتی عملیات‌های متنوعی مانند دریافت ، انتخاب و محاسبه در اطلاعات داریم.

برای ایجاد یک پایگاه داده رمزگذاری شده باید تمامی عملیات‌های قابل‌اجرا در بانک اطلاعاتی را بررسی نماییم و فرآیندی را پیش‌بینی کنیم تا از ابتدای ذخیره‌سازی ، تغییرات و دریافت اطلاعات به مشکل نخوریم. در این فرآیند قصد داریم چنانچه قبلاً گفتیم از یک سیستم واسطه بنام QM (Query Manager) استفاده نماییم که بین کاربران و بانک اطلاعات قرارگرفته و از یک‌سو به کاربران خدمات‌رسانی کند و از سوی دیگر با رمزگذاری اطلاعات از افشای آن‌ها جلوگیری کنیم.

چنانچه بخواهیم تمامی اطلاعات اعم از حساس و غیر حساس را رمزگذاری کنیم، ناچار هزینه بالای ذخیره و خواندن اطلاعات را قبول کردیم اما اگر تنها اطلاعات حساس را رمزگذاری کنیم کمی از بار هزینه کم می‌شود. در این رویکرد یا اطلاعات رمزگذاری شده را در یک ستون نگهداری می‌کنیم و یا در جدولی مجزا.

٤-١ استراتژی رمزگذاری :

برای تأمین حریم خصوصی و سطح بالاتری از امنیت ، ما از AES-CBC برای رمزگذاری داده‌های حساس استفاده می‌نماییم فقط مدیر پرس‌وجو (QM) کلیدهای مخفی (SK) را نگهداری می‌کند. ما برای دستیابی به سطح بالاتری از امنیت و پردازش سریع‌تر رمزنگاری ، از الگوریتم متقارن به‌جای الگوریتم نامتقارن استفاده کردیم.

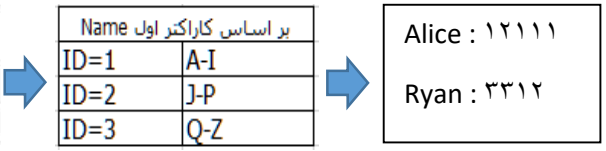
مدیر پرس‌وجو (QM) به‌عنوان یک سرور قابل‌اعتماد در فضای خصوصی یک سازمان یا شرکت قرار می‌گیرد و به‌عنوان یک واسطه بین کاربران و سرویس ابری عمل می‌کند و وظیفه پردازش درخواست‌ها (Query) و رمزگذاری BV (bit vector) برای هر جدول را دارد.

عنصر اصلی همه سیستم‌های پیشنهادی PT (Partitioning Tree) است که در آن پرس‌وجو به‌طور مناسب برای اجرا توسط سرور ابری بازنویسی می‌شود. مقادیر موجود در هر ستون یک جدول به چند پارتیشن تقسیم می‌شوند که در آن‌ها هر پارتیشن شامل مجموعه‌ای از مقادیر است سپس ، QM بر اساس این مشخصات PT را می‌سازد .

مثال در جدول زیر ، بر اساس حرف اول Name می‌تواند چندین Range تعریف نمود مثال A-I و J-P و برای هرکدام از این محدوده‌ها یک شناسه اختصاص دهیم و مجموع این محدوده‌ها می‌تواند یک PT را تشکیل دهد .

Table 3.1 The Students Table

ID	Name	SSN	VisaType	Department
01	Alice	12701	J-1	MATH
02	Ryan	25678	F-2	BUSN



ما در راه‌حل‌های خود نگران مصرف حافظه نیستیم زیرا اطلاعات حساس در هر ردیف به بیت‌ها رمزگذاری می‌شوند (یعنی کوچک‌ترین واحد محاسبه) و حافظه زیادی هم برای جستجو نیاز ندارد .

برای آشنایی بهتر می‌خواهیم عملیات اصلی و جبر رابطه‌ای یک پایگاه داده رابطه‌ای را توسط QM توضیح دهیم.

٤-١-١ عملیات اصلی پایگاه داده :

- دستور ***Insert*** یک عملیات مستقیم هست. QM دستور درج را دریافت می‌کند ، BV جدیدی برای رکورد تازه‌وارد شده ایجاد می‌کند ، آن را به ماتریس بیتی مربوطه (BVM - Bit Vectors as a Matrix) اضافه می‌کند و سپس داده حساس را رمزگذاری می‌کند و به فضای ابری می‌فرستد.
- دستور ***Select*** یک عبارت اساسی در تمام برنامه‌های پایگاه داده است. در QM ، فرآیند اجرای دستور این‌گونه است که بررسی کنید کدام‌یک از موارد زیر قابل‌اجرا است و سپس آن را اجرا کنید .

الف) هیچ‌یک از ستون‌های درخواستی در عملیات Select دارای اطلاعات حساس نیست ، پس رمزگذاری‌ای در ذخیره انجام‌نشده است و مستقیماً اطلاعات از پایگاه داده دریافت و نمایش داده می‌شود.

ب) تمامی ستون‌ها دارای اطلاعات حساس هست پس این اطلاعات به‌صورت رمزگذاری شده ذخیره‌شده‌اند. در این حالت QM برای هرکدام از ستون‌هایی که در دستور وجود دارد ، موقعیت BV ها از BVM دریافت می‌نماید.

سپس عملیات منطقی AND / OR را بر اساس شرایط موجود در فهرست‌های برگشتی برای هر ستون انجام می‌دهد .به‌عنوان‌مثال ، اگر شرط پرس‌وجو " Where Name='Mark' AND visa type = 'F-۲' " باشد ، QM هر BV را پیدا می‌کند که بیتی را نشان می‌دهد که مقادیر "QZ" را به گره نشان می‌دهد زیرا Mark زیرشاخه این گره در PT هست و تمامی BV ها به‌دست‌آمده در یک آرایه ثبت می‌شود. سپس عملیات منطقی AND رابین دو لیست انجام می‌دهد و برای دستور را مطابق رکوردهای یافت شده اصلاح می‌نماید.

- دستور ***Update*** نیز یکی از عملیات QM هست. برای این کار ابتدا QM توسط فرآیند توضیح داده‌شده Select ، رکورد را دریافت و رمزگشایی می‌نماید، سپس اطلاعات جدید را در رکورد یافت شده اصلاح می‌نماید و سپس آن را به سرویس ابری ارسال می‌نماید.
- دستور ***Delete*** نیز مانند Update عمل می‌کند با این تفاوت که پس از یافتن رکورد موردنظر دیگر نیاز به رمزگشایی نیست

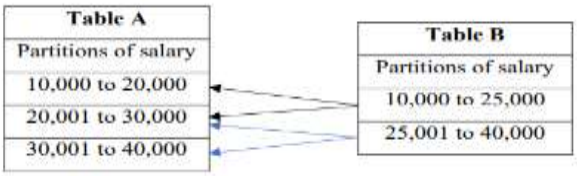
در سه دستور Update و Delete و Insert تمامی تغییرات بر روی BVM نیز اعمال می‌گردد تا در مراجعات بعدی معتبر باشد.

- در دستور Alter نیز در صورت حذف یک ستون ، تمامی BV ها از BVM حذف می‌شود و سپس دستور حذف ستون ارسال می‌شود و چنانچه ما بخواهیم ستونی را اضافه نماییم و تعیین کنیم دارای اطلاعات حساس هست ، تمامی رکوردهای موردنیاز BV در BVM ساخته و آماده می‌شود و اگر ستون دارای اطلاعات حساسی نباشد ، عملیات بدون پردازش خاصی انجام می‌شود.

٤-١-٢ عملیات جبر رابطه‌ای :

- در دستور ***Join*** شاید بهترین و تنهاترین راه‌حل ، دریافت کل اطلاعات رمزگذاری شده جدول و انجام این عملیات بر روی اطلاعات رمزگشایی‌شده باشد اما در حجم بالای اطلاعات ، این راهکار مناسب نیست برای همین باید تا جایی که امکان دارد، ستون‌های مورد پیوند Join بدون رمزگذاری در پایگاه ابری ثبت شود مگر آنکه ستون‌های پیوند دارای اطلاعات حساس باشد.

زمانی که پیوند ما با اطلاعات حساس روبرو باشد ، ما عملکردی مانند Select را خواهیم داشت یعنی QM ابتدا به ازای جداول پیوند فهرستی از PT را ساخته که شامل محدوده مقادیر مربوط به ستون‌های پیوند هست مثل تصویر روبرو و سپس ردیف‌هایی که دارای شرط پیوند هست را تعیین می‌کند و سپس دستور را بازنویسی کرده و ارسال می‌نماید. درواقع این‌گونه لیست سازی منجر می‌شود که ما بتوانیم اطلاعات کمتری جهت رمزگشایی دریافت و محاسبات کمتری انجام دهیم.



- در دستور ***Union*** هدف یکی کردن خروجی دو یا چند دستور هست که در آن باید مشخصات خروجی‌ها باید یکی باشد. این دستور مقادیر تکراری را نیز از خروجی نهایی حذف می‌نماید. البته باید جداول و خروجی‌هایی که در این عملیات شرکت می‌کنند حتماً با یک کلید یکسان رمزگذاری شده باشد. QM ابتدا ردیف‌های شرکت‌کننده در این عملیات را رمزگشایی می‌کند و آن‌ها را به یک LinkedHashSet (اگر نمی‌خواهید نظم درج را حفظ کنید اما می‌خواهید اشیاء منحصربه‌فرد را ذخیره کنید از HashSet استفاده می‌کنید اما اگر نظم درج مهم باشد از LinkedHashSet استفاده می‌شود) منتقل می‌کند.
- دستور ***Intersection*** فرآیندی است برای یافتن خروجی مشترک از دو یا چند خروجی و این کار بدون رمزگشایی اطلاعات ، کار سختی هست. حال اگر حجم اطلاعات بالا باشد ، بار سنگینی برای به دست آوردن اطلاعات ایجاد می‌شود و باید سعی کنیم این کار را بر روی سرور انجام دهیم چون امکان پردازش از سمت کاربر نیست. در این دستور هم مانند Union باید خروجی‌ها دارای مشخصات یکسان باشد و چون دنبال اعضای مشترک هستیم ، می‌توانیم از فرآیند دستور Join استفاده کنیم و خروجی حاصل‌شده را برای کاربر ارسال نماییم.

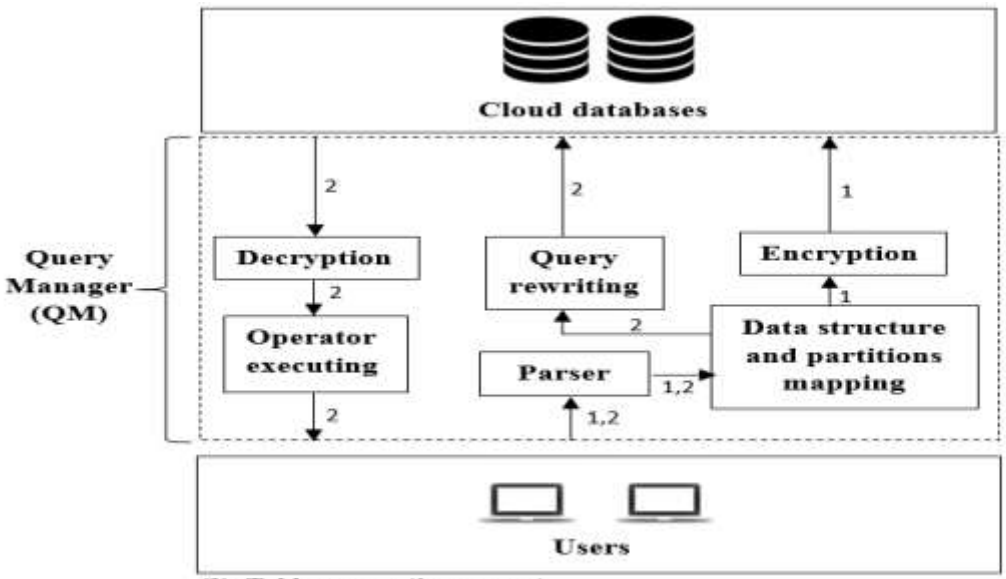
- عملگر ***Difference*** فرآیندی برعکس Intersection برای به دست آوردن خروجی خای غیرمشترک هست. برای انجام این عملیات از طریق Intersection ابتدا ردیف‌های مشترک را به دست آورده و آن را از خروجی اصلی خارج می‌کنیم تا خروجی به دست بیاید.
- عملگر ***Duplication Removal*** یا همان distinct فرآیندی برای حذف مقادیر تکراری از خروجی‌ها هست. در فضای ابری این عملیات غیرممکن است زیرا ما از الگوریتم رمزگذاری غیرقطعی (AES-CBC) استفاده می‌کنیم. بنابراین ، ماقبل از ارسال خروجی به کاربر ، این عمل را در QM انجام می‌دهیم. در این مورد نیز مانند Union می‌توانیم از یک LinkedHashSet باقابلیت حذف مقادیر تکراری استفاده کنیم.
- عملگرهای ***Aggregation and Sort*** شامل max, min, and count هست. ما در اینجا ستون‌ها با مقادیر دامنه‌ای و غیر دامنه‌ای داریم که در ستون‌های غیر دامنه‌ای می‌توانیم پردازش را در QM و در کاربری انجام دهیم. ما از محاسبات بر روی اطلاعات رمزگشایی‌شده به خاطر حجم محاسبات، اجتناب می‌کنیم. چنانچه قبلاً در دستور Select ما ابتدا اطلاعات مربوطه را همراه شرط‌هایی که لحاظ شده بود را تو BV و BVM یافت می‌کردیم ، این بار هم مرحله اول به این صورت هست. در مورد دستور count عملیات راحت هست چون ما ردیف‌ها را پیدا کردیم و کافی ست تعداد آن را به دست بیاوریم اما در خصوص Sum مجبور به رمزگشایی هستیم و در دستور Avg باید هم Sum و هم Count را انجام دهیم.
- عملگر ***Project*** یک عملیات بر اساس ستون‌ها هست که QM تمام ردیف‌ها مطابق ستون‌های درخواستی انتخاب می‌کند و چون انتخاب ستون هست ما نیازی به PT نداریم اما به‌هرحال نیاز به رمزگشایی هست و البته نیازی به حفظ اطلاعات تکراری نداریم.

٤-٢ روش‌های قابل‌اجرای رمزگذاری :

ما برای ثبت اطلاعات رمزگذاری شده ، دو روش را موردبررسی قرار می‌دهیم. یک روشی که در آن اطلاعات رمزگذاری شده به‌صورت یک ستون در همان جدول نگهداری می‌شود و روش دوم، روشی که اطلاعات رمزگذاری شده را در جدول مجزا نگهداری کنیم.

٤-٢-١ استفاده Bit Vectors به‌صورت یک ستون (BVSAC):

ما مدلی را بررسی می‌کنیم که Bit Vectors را به‌عنوان یک ستون اضافی در جدول رمزگذاری شده اصلی ذخیره می‌کند و این مدل را برای اجرای دستورات جبر رابطه‌ای مختلف بر روی داده‌های رمزگذاری شده طراحی کردیم. علاوه بر این ، ما محاسبه را به دو طرف تقسیم کردیم: سمت سرویس‌گیرنده و سرویس‌دهنده ابر (CP – Cloud Process) که در آن ما بیشترین محاسبه را با بازنویسی دستورات که CP را قادر به اجرای آن‌ها می‌کند ، به سایت CP منتقل می‌کنیم. این مدل در تصویر زیر نمایش داده‌شده است و این مدل برای تمام عملگرهایی که اینجا تحلیل شدند اجرا می‌گردد. در این مدل ما از رمزگذاری BV که تا اینجا توضیح داده شد استفاده کرده‌ایم و در آن به‌جای آنکه محاسبات را مداوم توسط QM در سمت کاربر انجام دهیم ، آن را به CP منتقل می‌کنیم.



(1): Tables encryption request.
(2): Relational algebra queries manipulation.

در این مدل جدول رمزگذاری شـده دارای یک ستون است که حاوی اطلاعات BV هست و چون اطلاعات به‌صورت بیت هست نیاز به نگرانی بابت حجم آن نیست اما چون نهایت سایز ذخیره برای هر ویژگی ۶۴Bit هست نیاز به بیش از یک ویژگی داریم. حتی اگر یک مهاجم بتواند از طریق استنتاج BV به اطلاعات دست پیدا کند، چون ما نام جداول و ستون‌ها را توسط الگوریتم رمزگذاری قطعی (AES) (که در آن متن‌های رمز برای هر متن ساده همیشه یکسان است) رمزگذاری می‌کنیم، این احتمال ضعیف‌تر و مشکل مرتفع می‌شود.

به‌عنوان‌مثال ، سـتون "Student-Rank" شـامل محدوده‌ای از مقادیر مانند senior ، grad ، junior و غیره اسـت. یک مهاجم ممکن اسـت بتواند مقادیر احتمالی را اسـتنباط کند ، حتی وقتی آن‌ها به‌طور تصـادفی با AES-CBC رمزگذاری شده‌اند.

اشتراک بیش از یک متن رمزگذاری با کلید یکسان باعث نگرانی نمی‌شود زیرا هکرها نمی‌توانند متن اصلی را به دست آورد مگر اینکه کلید رمزگذاری را به دست آورد و برای حل این نگرانی ، کلید به سرور ابری منتقل نمی‌شود.

برای مثال می‌خواهیم یک جدول را با کلید رمزگذاری کنیم. ابتدا QM نام جدول و سـتون‌های حسـاس را توسط AES-DET رمزگذاری می‌کنیم به این صورت که جدول دیگری در فضای ابری ساخته می‌شود و با آنچه تا اینجا درباره PT و الگوریتم‌های رمزگذاری گفته شـد ، QM ردیف‌ها را به‌صـورت BV رمزگذاری می‌کنید و جدول رمزگذاری شـده دارای یک سـتون Index است که در آن هر Index بخشی از BV هر ردیف است.

این فرآیند برای جدول List در شـکل زیر توضـیح داده‌شـده اسـت به این صـورت که مشاهده می‌کنید ستون‌های Name - Rank و Visa Type و Department در جدول رمزگذاری شـده توسـط رمزگذاری AES-DET و یک کلید مشـترک به‌صورت نام غیر آشنا درآمده است.

ID	Name	Rank	Visa type	Department
110	Alice	freshman	F1	Computer science
111	Sara	senior	J1	Computer engineering
112	John	junior	None	Information system
113	Ryan	Sophomore	J2	Math

Index	ID	ct_n	ct_r	ct_{vt}	ct_d	Reference
01	110	*&^	*_^%	*/d	^%^H	10001000010000100000
02	111	%^&	/+$	&^/	&&%$	00010100000100010000
03	112	)(#	%$/*	+-*&	)*#R	01000010000001001000
04	113	$#!	!@~K	*/f	@$%*	00100001000010000010

زمانی که دستوری برای خواندن اطلاعات ارسال می‌شود QM ابتدا ستون‌های استفاده‌شده در دستور به ستون‌های رمزگذاری شده ترجمه می‌کند مانند Select Name From List Where [Visa Type]=? تبدیل به Select Ctn From Cta where Ctvt=?

حال با پردازش بخش Where شروع به دست آوردن ردیف‌ها با توجه PT می‌نماید و ردیف‌های مربوطه یافت می‌شود و دستور بازنویسی شده را با توجه به ID ها می‌سازد. البته چنانچه قبلاً گفته‌شده PT شامل محدوده مقادیر هست و Query بازنویسی شده نیز محدوده‌ای از اطلاعات را می‌آورد و نه ردیف‌های قطعی! مثلاً اگر شرط [Visa Type]='F۱' باشد با توجه به حرف اول F و خاصیت Bit بودن ستون محدود ای به این صورت در نظر گرفته می‌شود Reference&۵۲٤۲۸۸>۰ سپس با توجه به حرف دوم که ۱ هست یک شرط ترکیبی ساخته می‌شود و درنهایت ممکن است دستوری مانند زیر ساخته شود Reference&٦٥٥٣٦>۰ And Reference&۵۲٤۲۸۸>۰

درنهایت ردیف‌های به‌دست‌آمده توسط QM رمزگشایی می‌شود و مجدد شرط بر روی آن اعمال می‌گردد و اطلاعات به خروجی ارسال می‌شود.

عملیات‌هایی مانند Insert و Update و Delete نیز چنانچه قبلاً گفته شد اعمال می‌گردد با این تفاوت که این بار از BVM استفاده نمی‌کنیم و این اطلاعات را در ستون Reference داریم.

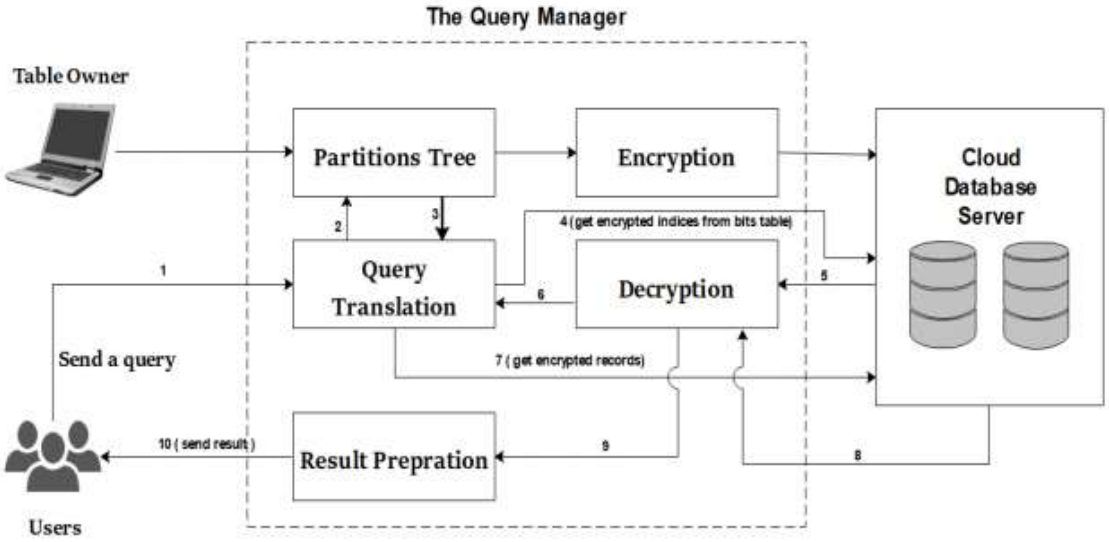
٤-۲-۲ استفاده Bit Vectors به‌صورت یک جدول مستقل (BVSIT):

در این روش می‌خواهیم BV را در یک جدول مستقل در سرور ابری ذخیره می‌کنیم و بین جدول رمزگذاری شده و جدول بیت‌ها یک ستون Index اضافه می‌کنیم که در آن مقدار Index هر رکورد در جدول رمزگذاری شده برابر با Index رکورد مربوطه در جدول بیت‌ها است.

برای اطمینان از اینکه در جدول بیت‌ها هیچ بیت داده‌ای توسط هکر از طریق استنتاج نشت نمی‌کند، ما Index موجود در جدول بیت‌ها را رمزگذاری می‌کنیم و سعی می‌کنیم حتی ازلحاظ قرارگیری موقعیت رکوردها نیز یکسان نباشد.

برای امنیت در BVSIT ، داده‌ها در تمام ستون‌های حساس با الگوریتم متقارن (AES-CBC) که رمزگذاری تصادفی است رمزگذاری می‌شوند.

فرآیند کلی این روش در شکل زیر نمایش داده‌شده است.



در این عملیات برای پردازش دستورات ، QM دو پرس‌وجو صادر می‌کند. پرس‌وجو اول ، شاخص‌های رمزگذاری شده تمام ردیف‌های دارای شرایط درخواستی را برای پرس‌وجو انتخاب‌شده از جدول بیت‌ها بازیابی می‌کند. پرس‌وجو دوم ، ردیف‌های رمزگذاری شده را مطابق جدول بیت‌ها انتخاب می‌کند و اطلاعات رمزگذاری شده را دریافت و رمزگشایی می‌کند و نتیجه به خروجی ارسال می‌شود. نحوه ساخته‌شدن جدول بیت‌ها و جدول رمزگذاری شده در تصویر زیر نمایش داده‌شده است.

اطلاعات قرارگرفته در جدول بیت‌ها همان PT بر اساس محدوده‌های کاراکترها هست که قبلاً توضیح داده‌شده است

< جدول بیت ها

Index	1	2	3	4	5	6	7	8	9	10	11	12	13
CT(3)	0	1	0	1	0	0	1	0	0	1	0	0	0
CT(4)	0	1	0	1	0	1	0	0	0	0	0	1	0
CT(1)	1	0	0	1	0	0	1	0	0	0	1	0	0
CT(2)	0	0	1	0	1	0	0	1	0	0	0	1	0

ID	Name	SSN	Rank	Salary
01	Alice	12701	Secretary	30,000
02	Ryan	25678	Admin	60,000
03	Mark	46932	Secretary	29,000
04	John	42213	Manager	55,000

Index	ID	Name	SSN	Rank	Salary
1	^%#	)@(	$@^#	@^#S	FS^@
2	(#*	^#*	)@H2	+ @	%@*&
3	(#(	(#&	$@%	@^#@	@%TW
4	)^@	)^#H	!@M	*@^#	+_)#FQ

< جدول رمزگذاری شده

عملیات‌هایی مانند Insert و Update و Delete از همان روش‌های گفته‌شده استفاده می‌کند و با این تفاوت که برای هرکدام از عملیات‌ها دو دستور ارسال می‌شود ، یکی برای جدول رمزگذاری شده و دیگری جدول بیت‌ها.

فصل پنجم

مقدمه

هر ایده یا اســـتراتژی در بخش نرم‌افزار باید در مرحله تســـت و آزمایش قرار بگیرد تا بتوان نســـبت به اجرا و منابع موردنیاز، بررســی و نیازسنجی شود زیرا استفاده از ایده نامناسب و پرهزینه منجر به شکست سیستم می‌شود. با آنچه از رمزگذاری و پایگاه داده گفته‌ایم می‌توان مدل‌های معتبر و رایج را بررسی و مقایسه نمود.

در آزمـایش بین مـدل‌ها و تئوری‌های اجرای پایگاه داده رمزگذاری شـــده، پارامترهای متعددی مدنظر قرار می‌گیرد مانند : حجم اطلاعات، زمان اجرا، هزینه‌های سخت‌افزاری مانند حافظه و امنیت مدل‌ها.

مدل‌های پایگاه داده رمزگذاری شـــده با توجه به ایده و اســـتراتژی که در آن قرار دارد در عملیات‌های مختلف دارای پاسخ و ارزیابی متفاوتی هست مانند آنکه مدلی در هنگام ذخیره اطلاعات به علت طراحی آن باعث هزینه بالا می‌شود و یـا مـدلی در ذخیره‌ســـازی اطلـاعـات دارای نتـایج بهتر ولی در بازخوانی اطلاعات دچار کندی یا هزینه بالای اجرا می‌شـود. البته عملیات‌های پایگاه داده تنها ذخیره و خواندن اطلاعات نیست و عملیات‌های محاسباتی نیز موردبررسی قرار می‌گیرد پس آن چیزی که به‌عنوان مدل مناسـب انتخاب می‌گردد، مدلی هست که دارای میانگین بهتری نسبت به سایر مدل‌ها باشد و درنهایت باید مدل انتخابی را در اجرای فضای ابری موردبررسی قرارداد.

۵-۱ آزمایش و ارزیابی :

تا اینجا ، ما به‌طور مفصـل مدل‌های پیشـنهادی را توضـیح دادیم و برای هر مدل ، الگوریتم‌های توسـعه‌یافته را برای اجرای عملگرهـای مختلف جبر رابطه‌ای ارائه دادیم .همان‌طور که قبلاً گفته شـــد ، این پایان‌نامه باهدف پیاده‌ســـازی ، ارزیابی و مقایسه عملکرد نمونه‌های پیشنهادی برای انواع مختلف عبارات با استفاده از معیارهای مختلف (به‌عنوان‌مثال ، تأخیر اجرای نمایش داده‌ها ، نیازهای فضا در QM و سرور ابری ، و درصد سربار محاسبات درQM) ارائه دادیم.

حال برای مقایسه مدل‌ها و سیستم‌های ارائه‌شده نیاز به آزمایش آن‌ها داریم.

برای انجام همه آزمایش‌ها برای همه سـیستم‌ها BVM ، BVSAC ، BVSIT ، OBT ، CBF ، CryptDB از یک سـیسـتم با ۶ گیگابایت رم ، ۱ ترابایت HDD و پردازنده Core i۵ با ۲,۸ گیگاهرتز اسـتفاده کردیم .برای اجرای توابع

QMدر مدل‌های پیشـنهادی ، از Java برای اسـتفاده کردیم .سـرور MySQL بر روی دستگاه کاربر استفاده شد و ما از Java Database Connectivity (JDBC) به‌عنوان اتصـال‌دهنده به MySQL اسـتفاده کردیم و تمام آزمایش‌ها بر روی دستگاه محلی انجام‌شده است تا تأخیر ارتباط از گزارش‌ها حذف شود.

در تمامی مدل‌های خود ، از نسـخه تصـادفی AES-CBC برای رمزگذاری داده‌های حسـاس اسـتفاده کردیم و برای رمزگذاری و رمزگشایی از کتابخانه "javax.crypto" استفاده نموده‌ایم.

ما OBT و CBF را اجرا کردیم زیرا پیاده‌سـازی آن‌ها برخلاف CryptDB که برای اسـتفاده عمومی در GitHub در دسـترس اسـت به‌صورت آنلاین در دسترس نیست و برای ذخیره BV آن‌ها را به‌صورت محلی در QM در مدل اول ذخیره کردیم و آن‌ها را برای استفاده در آینده در یک فایل متنی نوشتیم و در سایر مدل‌ها آن‌ها را در داده ابری بارگذاری می‌کنیم.

برای شـروع آزمایش ما به‌طور تصـادفی چهار جدول ایجاد کردیم و برای هر ستون فهرستی از مقادیر تعریف کردیم و به یک برنامه java اجازه می‌دهیم تا با انتخاب مقادیر از فهرسـت‌ها ، ردیف‌هایی را بسازد .اندازه جداول (تعداد رکوردها) ۱۰هزار – ۲۰هزار – ۵۰هزار و ۱۰هزار ردیف بود.

ما در کل ۲۴ ستون برای همه جداول داشتیم و همه آن‌ها را به‌استثنای ستون ID به‌عنوان اطلاعات حساس در نظر گرفتیم.

در مطالعه خود ، اگرچه می‌توانستیم از مدل‌های پیشنهادی با جداول کوچک استفاده کنیم ، اما ما بر روی جدول‌های بزرگ تمرکز کردیم.

۵-۲ ارزیابی :

ارزیابی شامل موارد زیر هست :

۱) آزمایش و ارزیابی هزینه اجرای عملیات پایه پایگاه داده (Insert – Update – Deleted - Insert)
۲) آزمایش و ارزیابی هزینه اجرای عملیات محاسباتی (Sum – Min – Max - Avg)
۳) آزمایش و ارزیابی هزینه‌های اجرای عبارات جبر رابطه‌ای (Join – Union - Intersection)
۴) شناسایی نیازهای فضایی برای هر سیستم

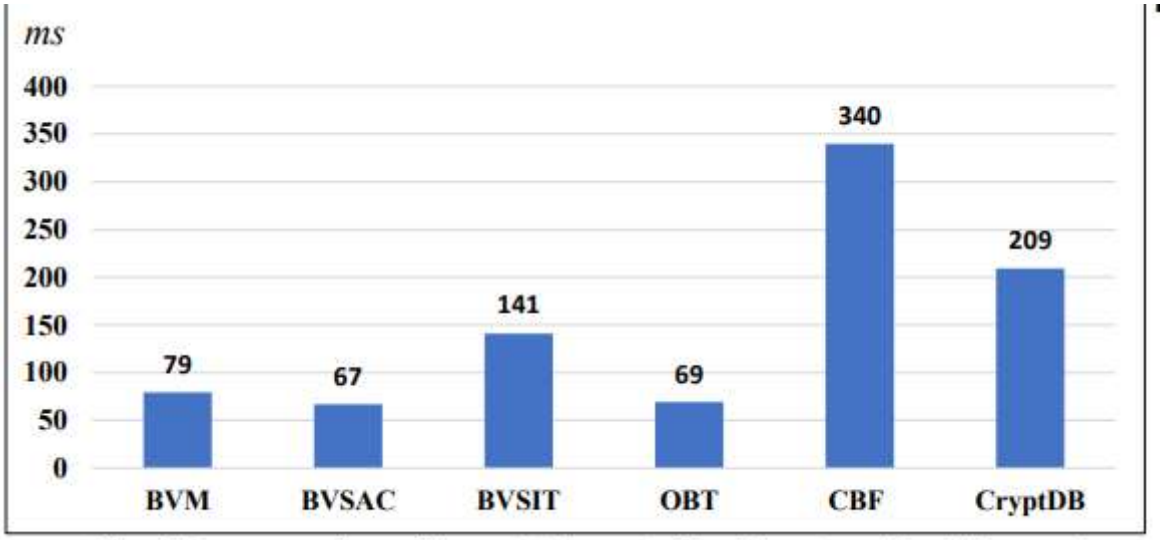
در مدل‌های پیشـنهادی ، مرحله رمزگذاری پایگاه داده اصـلی شـامل تجزیه رکوردها ، تولید شـاخص‌ها ، سـاخت BV ، رمزگذاری داده‌های حساس و قراردادن داده‌های رمزگذاری شده در جدول رمزگذاری شده در سرور ابری است .

در جدول زیر ما زمان صـرف شـده توسـط هر سـیسـتم برای رمزگذاری هر جدول را مقایسـه کرده‌ایم و همان‌طور که دیده می‌شـود ، مدل BVSACکمترین تأخیر در رمزگذاری و درج را در بین همه سـیستم‌ها تجربه کرده است زیرا QM نیازی به ذخیره و مدیریت BV ها به‌صورت محلی در QM مانند BVM ندارد.

The delay of the original database encryption comparison among all systems in minutes

N.R	BVM	BVSAC	BVSIT	CryptDB	CBF	OBT
10k	12	11	25	44	55	13
20k	26	23	48	66	112	24
50k	65	54	115	158	291	55
100k	147	112	223	308	578	113

پس از BVSAC مدل OBT است که هر ردیف را به‌صورت یک بلوک رمزگذاری می‌کند و آن را در ستونی قرار می‌دهد و درروی دیگر، مدل CBF بالاترین تأخیر فرایند رمزگذاری را دارد زیرا به ازای هر ستون باید جداگانه درج انجام شود و رکوردها را به جداول مختلف در سرورهای مختلف ابری قرار می‌دهد. مقایسه مدل‌ها برای عملیات Insert در جدول زیر نمایش داده‌شده است

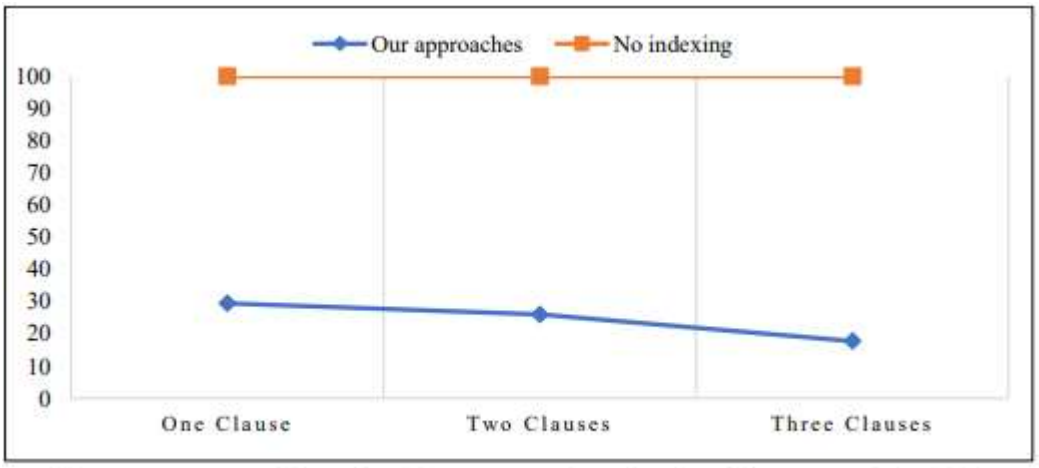


The delay comparison of insert statements for all systems, in millisecond

مدل کند بعدی مدل CryptDB هست که به دلیل محاسبات سنگین رمزگذاری Onion هست.

۵-۲-۱ آزمایش شماره یک :

ازآنجاکه پراستفاده‌ترین عملیات ، عمل انتخاب (Select) هست ، ما بیشترین تمرکز خود را بر روی این عملیات می‌گذاریم. در این آزمایش ، ما درصد متوسط ردیف‌های رمزگذاری شده واکشی شده از جداول رمزگذاری شده برای مدل‌های پیشنهادی را محاسبه کردیم .ما همچنین مطالعه کردیم که چگونه تعداد بندهای موجود در شرایط پرس‌وجو می‌تواند به کاهش دامنه موارد واکشی کمک کند.



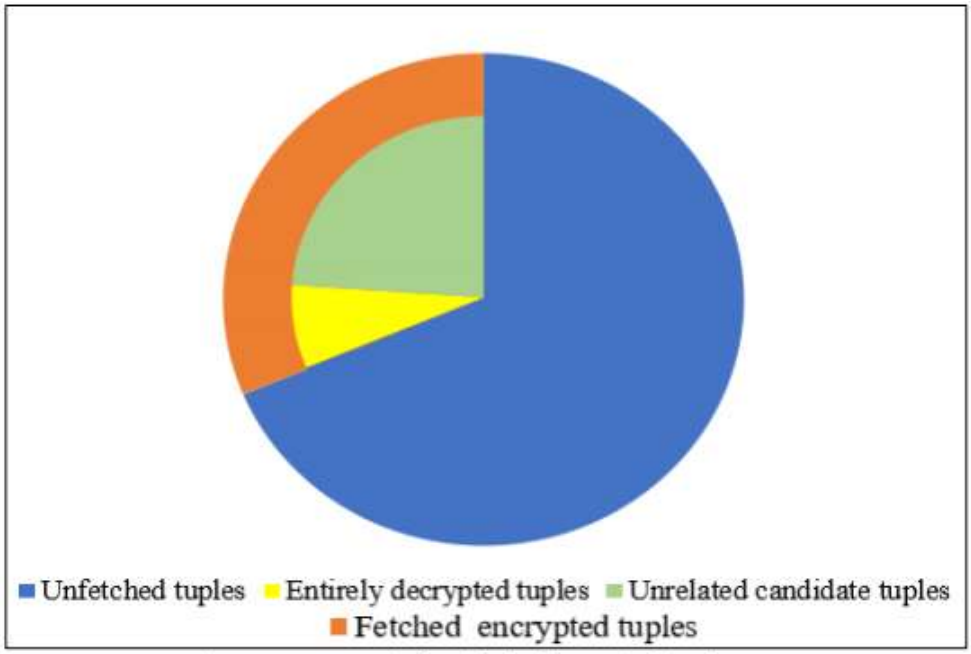
The percentage of retrieved encrypted tuples for all proposed models

چنانچه در نمودار می‌بینید ، بدون استفاده از هیچ‌یک از نمونه‌های اولیه پیشنهادی برای مدیریت پایگاه داده رمزگذاری شده (یعنی بدون Index) باید کل جدول رمزگذاری شده برون‌سپاری شده را بازیابی کنیم.

۵-۲-۲ آزمایش شماره دو :

چنانچه تا اینجا گفته شد هدف ما این است که برای جلوگیری از محاسبات سربار ، که باعث تسریع در زمان اجرای درخواست می‌شود ، محاسبات رمزنگاری را در QM کاهش دهیم.

در تصویر زیر می‌بینیم که از کل مجموعه رمزگذاری شده واکشی شده ، میانگین ردیف‌های کاملاً رمزگشایی‌شده کمتر از ۲۴٪ ردیف‌های واکشی شده است .بر این اساس ، مدل‌های ما کارآمد هستند زیرا ما نه‌تنها دامنه داده‌های برون‌سپاری شده بازیابی شده را محدود می‌کنیم ، بلکه در صورت امکان محاسبات غیرضروری (محاسبات رمزنگاری) را نیز حذف می‌کنیم.



Average percent of entirely decrypted tuples

۵-۲-۳ آزمایش شماره سه :

در این مطالعه ، ما بر روی عبارات *Select* متمرکز شدیم ، زیرا به‌طور گسترده در سیستم‌های پایگاه داده استفاده می‌شود. بنابراین ، ما عبارات *Select* را با تعداد بندهای مختلف و منفرد ، مانند انتخاب با نام ستون و انتخاب همه ستون‌ها ، با مدل‌های معرفی‌شده تا اینجاCryptDB ، OBT و CBF آزمایش و ارزیابی کردیم.

جدول زیر کل زمان اجرا را برای همه سیستم‌ها هنگام اجرای عبارات *Select* برای بازیابی مقادیر با خاصیت ستون حساس ارائه می‌دهد و مدت‌زمانی که اندازه‌گیری کردیم زمان تجزیه پرس‌وجو بود تا نتیجه پرس‌وجو نهایی در میلی‌ثانیه (*ms*)است.

	1 clause						2 clauses						3 clauses					
	BVM	BVSAC	BVSIT	OBT	CBF	CryptDB	BVM	BVSAC	BVSIT	OBT	CBF	CryptDB	BVM	BVSAC	BVSIT	OBT	CBF	CryptDB
10k	242	246	229	200	231	870	136	153	133	98	238	1329	115	132	112	77	324	1558
20k	326	382	352	275	338	1633	259	263	215	179	483	2016	213	224	192	142	542	12626
50k	624	510	783	481	547	10764	479	367	614	289	746	18969	440	303	545	221	893	21634
100k	1107	918	1240	791	970	20135	840	702	1128	562	1478	37078	835	610	1021	493	1604	59910

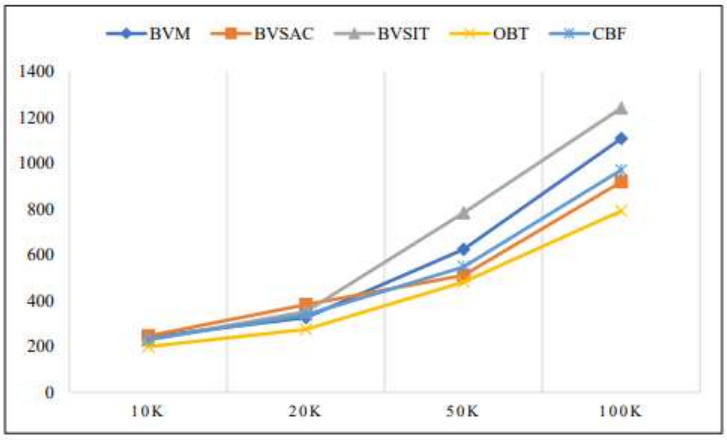
چنانچه مشاهده می‌شود مدل BVSAC به‌طور متوسط کوتاه‌ترین زمان اجرا را از بین تمام مدل‌های پیشنهادی برای همه موارد دارد. این نتیجه به این معنی است که BVSAC ازنظر زمان اجرا ، با پایگاه داده‌های بزرگ‌تر عملکرد بهتری دارد و از زمان شروع بیشتری نسبت به سایر مدل‌های پیشنهادی برخوردار است ، به همین دلیل سرعت آن نسبت به برخی دیگر از سیستم‌ها برای پردازش پایگاه داده با ۲۰هزار و ۱۰هزار ردیف کمتر است. عامل دیگری که سرعت پردازش پرس‌وجو را افزایش می‌دهد این است که ما از عملیات bitwise ارائه‌شده توسط MySQL در فضای ابری بهره‌مند می‌شویم زیرا بیت‌ها را به‌عنوان یک ستون مستقل همراه با هر رکورد رمزگذاری شده ذخیره می‌کنیم.

برای پایگاه داده‌های کوچک‌تر (۱۰هزار یا کمتر)، مدل BVSIT سریع‌ترین زمان اجرا را دارد زیرا عملیات bitwise به‌صورت ستون پایه انجام می‌شود ، که امکان جستجوی سریع در جدول بیت‌ها را فراهم می‌کند .بااین‌حال ، وقتی پایگاه‌های داده رشد می‌کنند ، با رشد مقدار شاخص‌های رمزگشایی در QM ، عملکرد BVSIT کاهش می‌یابد.

مدل BVM در جایگاه دوم برای پایگاه داده با بیش از ۵۰هزار ردیف است و اصلی‌ترین عاملی که عملکرد BVM را تحت تأثیر قرار می‌دهد زمان بارگیری BV ها از دیسک سخت به حافظه اصلی و سپس جستجوی آن‌هاست.

ازآنجاکه مدل CryptDB دارای تایم خیلی بالاتری نسبت به سایر مدل‌ها هست، ما آن را حذف و میانگین سایر مدل‌ها در هر سه حالت را درآورده‌ایم.

نمودار روبرو حاصل مقایسه مدل‌ها هست.



(b) Average total processing time for all models, excluding CryptDB.

با توجه به نمودار و جدول بالا مشاهده می‌شود که مدل OBT دارای سرعت بهتری نسبت به سایر مدل‌ها هست زیرا مقدار داده‌های رمزگشایی کمتر از پیشنهاد ما بود (همه مقادیر به‌عنوان یک بلوک ذخیره می‌شوند که منجر به رمزگشایی کمتر برای هر سطر می‌شود) البته این برای زمانی است که دستور Select با تعداد ستون بیشتر انجام شود زیرا در دریافت تک‌ستونی ، سایر مدل‌ها از سرعت بهتری برخوردار هستند.

۵-۲-٤ توان عملیاتی :

توان عملیاتی به‌عنوان مقدار داده منتقل‌شده در یک‌زمان معین تعریف می‌شود .با اندازه‌گیری توان عملیاتی ، می‌توان فهمید که کدام سیستم با افزایش داده‌های درخواستی ، پاسخ بیشتری به دستورات کاربر می‌دهد ، زیرا کاربر نهایی کسی است که تحت تأثیر کاهش سرعت سیستم قرار خواهد گرفت.

برای اندازه‌گیری توان عملیاتی ، مجموعه‌ای از پرس‌وجوها را برای بازیابی٪۲۵ و سپس ٪۵۰ رکوردها از جدول دارای ۱۰۰۰۰۰ رکورد اجرا کردیم .سپس ، ما مقدار داده‌های ساده (داده‌های سوابق پس از رمزگشایی) را اندازه‌گیری می‌کنیم و آن‌ها را بر زمان موردنیاز هر سیستم تقسیم می‌کنیم تا داده‌های موردنیاز را به کاربر تحویل دهیم

$$\text{Throughput} = \frac{\sum_{k=0}^{n} \text{unencrypted record's size (in bytes)}}{\text{Total time taken to deliver the data (MS)}}$$

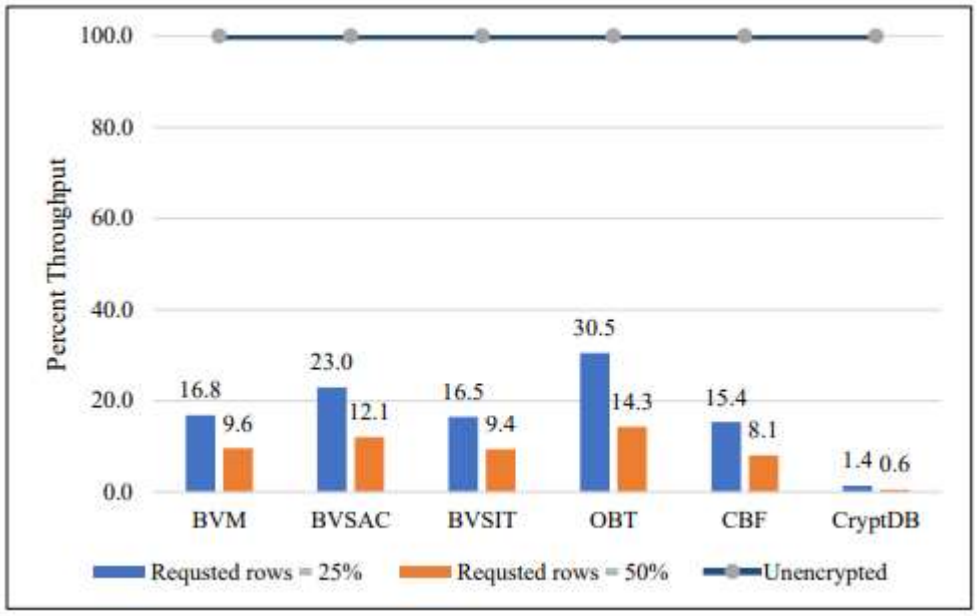
جدول زیر این محاسبات را برای تمامی مدل‌ها نمایش می‌دهد

	kilobytes per second (kB/s)						
	BVM	**BVSAC**	**BVSIT**	**OBT**	**CBF**	**CryptDB**	**MySql**
Requested rows = 25%	794	1083	778	1440	727	66	4720
Requested rows = 50%	971	1221	947	1447	816	57	10114

The amount of plain data in kB that each system can deliver to the user per second.

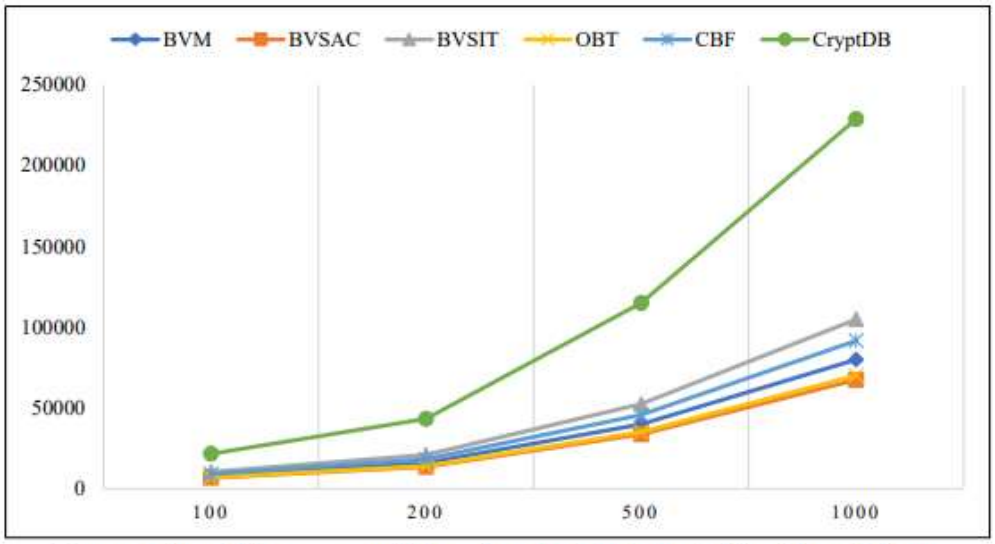
چنانچه مشاهده می‌شود مدل BVSAC توان عملیاتی بالاتری را در بین سیستم‌های پیشنهادی دارا هست که آن را بهترین گزینه برای کاربران نهایی می‌کند که به دنبال یک سیستم پاسخگویی سریع‌تر هستند. همچنین مدل OBT

بالاترین توان عملیاتی را دارد ، به این دلیل که برای رمزگذاری داده‌ها به مقدار کمتری از بایت در میان همه سیستم‌ها نیاز دارد. برای درک بهتر مقادیر را به‌صورت نمودار درصد درآوردیم.



۵-۲-۵ آزمایش شماره چهار (آزمایش دستورات Update و Delete) :

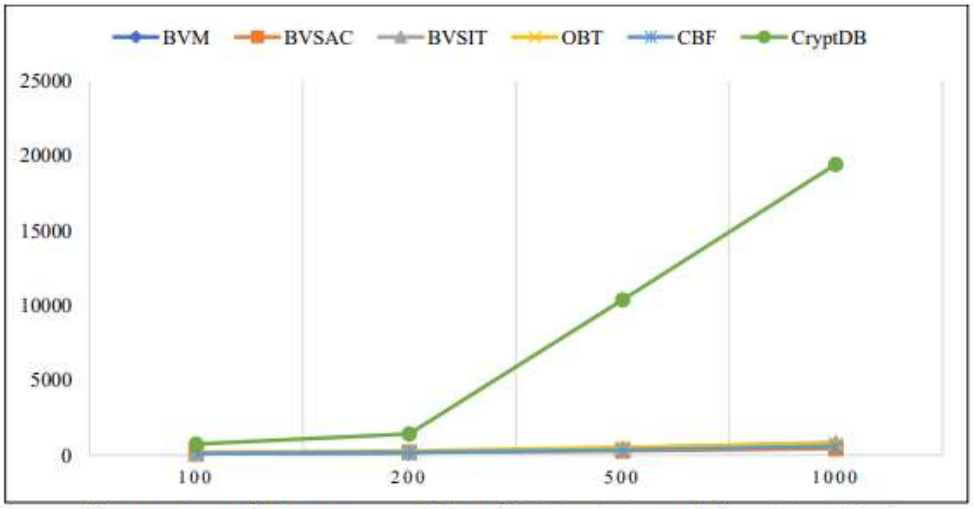
ما این آزمایش را با مدل‌های مختلف انجام و نمودار زیر متوسط زمان هزینه شده توسط هر سیستم برای اجرای عبارات *Update را نشان می‌دهد* .در این آزمایش ، عبارات *Update* را فقط با یک فیلد اجرا کردیم. همان‌طور که در شکل دیده می‌شود ، هزینه زمان به‌روزرسانی در همه سیستم‌ها زیاد است و نتیجه به‌روزرسانی یک فیلد رمزگذاری شده در سیستم‌های پایگاه داده است. محور X نشان‌دهنده تعداد ردیفی است که از دستورات *Update* تأثیر می‌پذیرد ، به این معنی که داده‌های موردنیاز را انتخاب کرده و سپس در حالت درج ۱۰۰ردیف ، در حالت دوم ۲۰۰ ردیف و غیره صادر می‌کند.



Comparison of the average delay of update statements for all models to update a number of existing tuples (100, 200, 500, and 1,000 tuples).

در مدل‌های پیشـنهادی ، BVSIT کندترین سیستم پس از CryptDB است زیرا *Update* باید هم در جدول رمزگذاری شده اصلی و هم در جدول بیت‌ها انجام شود و دومین مدل کند است. علاوه بر این ، BVM کمی تأخیر بالاتر از BVSAC را تجربه کرد اما هنوز هم سریع‌تر از CBF عمل می‌کند ، که سومین مدل کندترین در این مقایسه است .در CryptDB ، هزینه به‌روزرسانی بالاترین هست. به‌طور خلاصه ، می‌توان گفت BVSAC کارآمدترین مدل برای اجرای *Update* است.

آزمـایش را بـا انجام عملیات Delete انجام دادیم و نمودار زیر زمانی اســت که هر مدل برای حذف تعداد متفاوت (۱۰۰ ، ۲۰۰ ، ۵۰۰ و ۱۰۰۰ ردیف) صرف می‌کند ، درحالی‌که شرط حذف فقط یک‌بند دارد.



Comparison of the average delay of delete statements for all models to delete a different number of tuples (100, 200, 500, and 1,000 tuples).

فرآیند حذف ، ردیف‌های موردنیاز را انتخاب کرده و سپس آن‌ها را حذف می‌کند. حذف در همه مدل‌های پیشنهادی کارآمد اسـت زیرا حذف پس از اجرای عملیات *Select* برای بازیابی ردیف‌های موردنیاز انجام می‌شـود. به‌جای ارسال یک پرس‌وجو برای حذف هر رکورد ، ما Index رکورد را به دسـت می‌آوریم و سپس یک دستور صادر می‌کنیم تا هر رکورد از شاخص آن را در دســتور *حذف* ، یعنی *حذف* از TABLE_NAME که در آن index اســت *حذف کند*. CryptDB کندترین سیستم برای اجرای دستورات *حذف* به همان دلایلی است که قبلاً ذکر کردیم.

۵-۲-۶ آزمایش شماره پنج (دستورات محاسباتی) :

مـا این آزمـایش را بـا دو بخش تقســیم و در بخش اول متوســط زمـان هزینـه شــده توســط هر ســیســتم برای اجرای عملیات *Sum* , *Average* , *Count* در مدل‌های مختلف را به دسـت آورده‌ایم. در بخش دوم سایر دستورات مانند Max و Min مورد آزمایش قرارگرفته‌اند.

تمامی آزمایش‌ها برای محدوده مشخصی از اطلاعات هست و چیزی حدود ۶٪ از کل اطلاعات.

	SUM						AVERAGE						COUNT					
	DVM	BVSAC	BVSIT	OBT	CBF	Cryptdb	DVM	BVSAC	BVSIT	OBT	CBF	Cryptdb	DVM	BVSAC	BVSIT	OBT	CBF	Cryptdb
10k	110	121	107	206	143	878	111	121	107	206	147	878	104	115	101	200	131	746
20k	196	212	232	284	212	1718	198	214	234	286	208	1718	189	205	225	277	193	1417
50k	440	304	473	525	361	10762	441	305	474	527	363	10762	431	295	464	516	328	10332
100k	733	460	850	807	645	20185	734	462	853	806	647	20185	722	449	839	796	599	19325

Delay comparison in milliseconds of executing SUM, AVERAGE, AND COUNT functions. (only one predicate in the queries)

در سیستم OBT ، طبق ساختار ، اجباراً کل ردیف‌های رمزگذاری شده واکشی شد ، که نتیجه آن محاسبه هزینه‌های سنگین‌تر در QM است .در CBF و CryptDB ، الگوریتم رمزنگاری مورداستفاده برای رمزگذاری قسمت عددی برای پشتیبانی از عملیات جمع‌آوری روی مقادیر رمزگذاری شده ، رمزگذاری homomorphic است که مدل‌های آن بر ضرب مدو لار برای تولید مجموع رمزگذاری شده استفاده می‌کند، که نیاز به محاسبه سنگین‌تر دارد.

جدول زیر آزمایش را در هر سیستم برای اجرای عملیات *Max* , *Min* نشان می‌دهد.

	Average delay of MAX and MIN					
	BVM	BVSAC	BVSIT	OBT	CBF	CryptDB
10k	112	123	109	208	147	878
20k	201	217	237	289	208	1714
50k	447	311	480	532	363	10762
100k	741	468	858	815	647	20185

Delay comparison in milliseconds of executing MAX and MIN functions. (only one predicate in the queries)

با بررسی این دو جدول در آزمایش‌ها عبارات محاسباتی ، می‌توانیم مدل که دارای تأخیر کمتر هست را به دست آوریم که آن را می‌توان برای پایگاه داده کوچک در نظر گرفت ، اما برای پایگاه‌های داده بزرگ‌تر ، مدل BVSAC برنده است ، درحالی‌که BVSIT محاسبات سنگین‌تری را تجربه می‌کند ، همان‌طور که گفته شد در آزمایش قبلی CryptDB به دلیل عملیات رمزگشایی فشرده ، کندترین مدل هست.

۵-۲-۷ آزمایش شماره شش (دستورات Join , Union) :

در این آزمایش قصد داریم هزینه زمانی دستورات *Union* , *Join* را مقایسه کنیم و چنانچه قبلاً گفتیم در مدل CryptDB و CBF این دو دستور پشتیبانی نمی‌شود، لذا از آزمایش‌ها حذف می‌شود. در این آزمایش از دو جدول که توسط ستون رمزگذاری شده ID باهم Join شده‌اند و با آزمایش‌ها ۵۰۰ و ۱۰۰۰ و ۱۰۰۰۰ ردیف استفاده کرده‌ایم که به ترتیب دارای ۱۰۰ و ۲۰۰ و ۱۰۰۰ رکورد در شرط Join استفاده‌شده هست.

مشخصات دو جدول در تصویر زیر آماده است

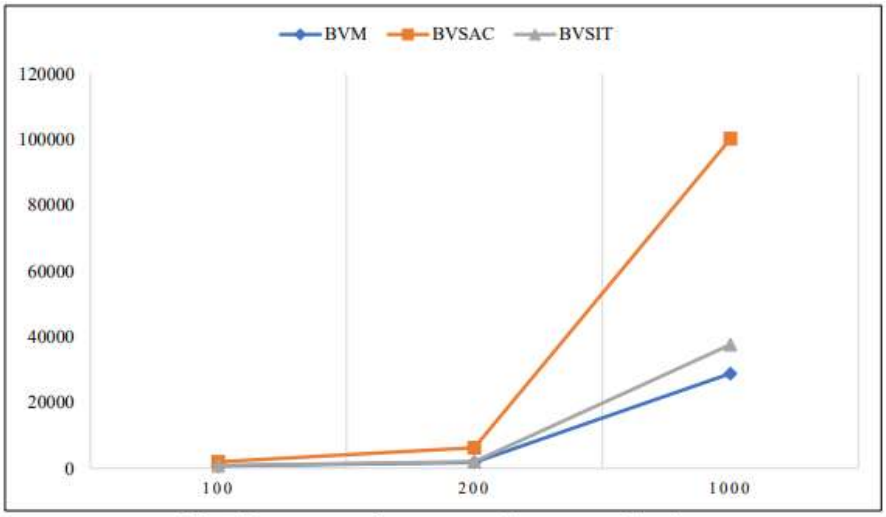
students table.

Name of the Attribute	Data type	Is sensitive?
ID	int	Yes
Name	varchar	Yes
SSN	int	Yes
Visa_Type	varchar	Yes
Salary	int	Yes
Department	varchar	Yes

international students

Name of the Attribute	Data type	Is sensitive?
ID	int	Yes
Name	varchar	Yes
Address	int	Yes
Citizenship	varchar	Yes

نمودار مقایسه مدل‌ها در عملیات Join به شکل زیر آماده است.

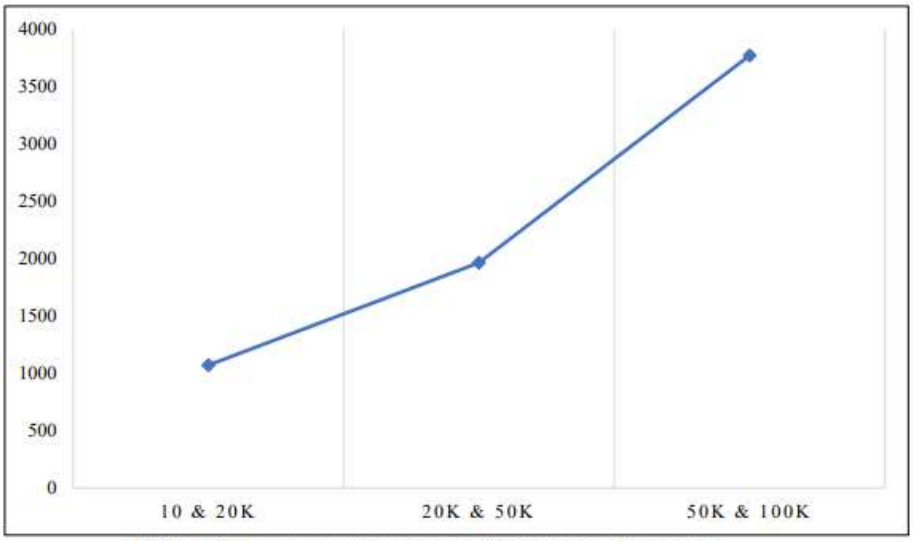


Join delay comparison among the proposed systems

این نمودار نشان می‌دهد که تأخیر BVSAC بالاترین و پس‌ازآن BVSIT است و BVM دارای تأخیر کمتری است. تأخیر زیاد در دستورات Join در همه مدل‌های پیشنهادی به دلیل ماهیت محاسبه Join و مقدار داده‌های رمزگشایی‌شده است. سپس ، پس از اجرای شرایط Join (درQM) ، تاپلهای رمزگشایی‌شده به داخل LinkedHashSet برده شدند تا تکراری را حذف کند.

در پایان ، اگرچه تأخیرهای بالایی را تجربه کردیم ، اما با استفاده از سیستم‌های پیشنهادی بیش از پایگاه داده‌هایی که با الگوریتم رمزگذاری تصادفی رمزگذاری شده بودند ، مانند AES-CBC ، دستورات join را اجرا کردیم که این تأخیر برای امنیت بهتر هست.

نمودار هزینه زمان در تمامی مدل‌ها در دستور Union به شکل زیر آماده است و از میانگین کلیه مدل‌ها استفاده‌شده است زیرا در آزمایش‌ها متوجه شـدیم که زمان مدل‌های خیلی به هم نزدیک هسـت . سعی شده است دستور Union به صورتی انتخاب شـود که بدون تغییر توسـط QM به سـرور ارسـال شـود. تأخیری که اندازه‌گیری کردیم از زمانی اسـت که QM پرس‌وجو را قطع می‌کند تا نتیجه نهایی پرس‌وجو تشکیل شود. برای حذف موارد تکراری ، ما ردیف‌های رمزگشایی‌شده را از هر دو جدول به یک LinkedHashSet هدایت کردیم.



Union delay comparison among the proposed systems

چنانچه مشاهده می‌شود با افزایش رکوردها ، زمان بیشتری صرف می‌شود.

۵-۲-۸ آزمایش شماره هفت (بازنویسی دستورات توسط QM) :

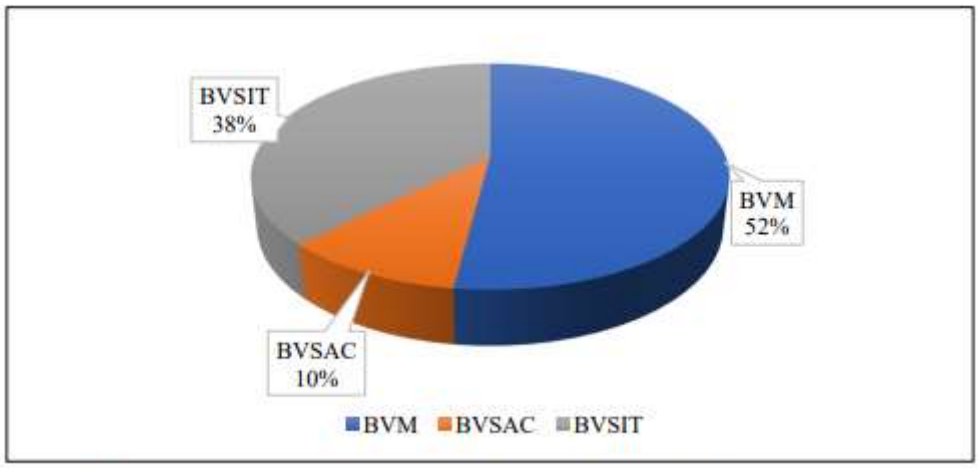
چنانچه قبلاً گفتیم در رمزگذاری ما از QM استفاده می‌کنیم تا دستورات را بازنویسی کند تا به‌سرعت بهتر و تأخیر کمتری در پردازش و نمایش اطلاعات برسیم. این آزمایش میانگین زمان صرف شده توسط هر مدل برای بازنویسی یک پرس‌وجو را نشان می‌دهد.

بازنویسی دستورات شامل چند مرحله هست که در جدول زیر برای مدل‌های انتخابی بررسی‌شده است

	Query parsing	PT search	BVs store and search (at QM)	Indices decryption
BVM	✓	✓	✓	x
BVSAC	✓	✓	x	x
BVSIT	✓	✓	x	✓

برای محاسبه میانگین ، مجموعه‌ای از Query را برای موارد مختلف *Select* با یک ، دو و سه بند اجرا کردیم و جدول زیر حاصل شد.

ما روی عبارات *Select* متمرکز شدیم زیرا این‌ها بخشی از دستورات به‌روزرسانی و حذف هستند ، بنابراین بررسی عبارت *Select* برای اندازه‌گیری هزینه بازنویسی دستورات در هر مدل کافی است همچنین عملیات رمزگذاری را از مقادیر آزمایش حذف کردیم.

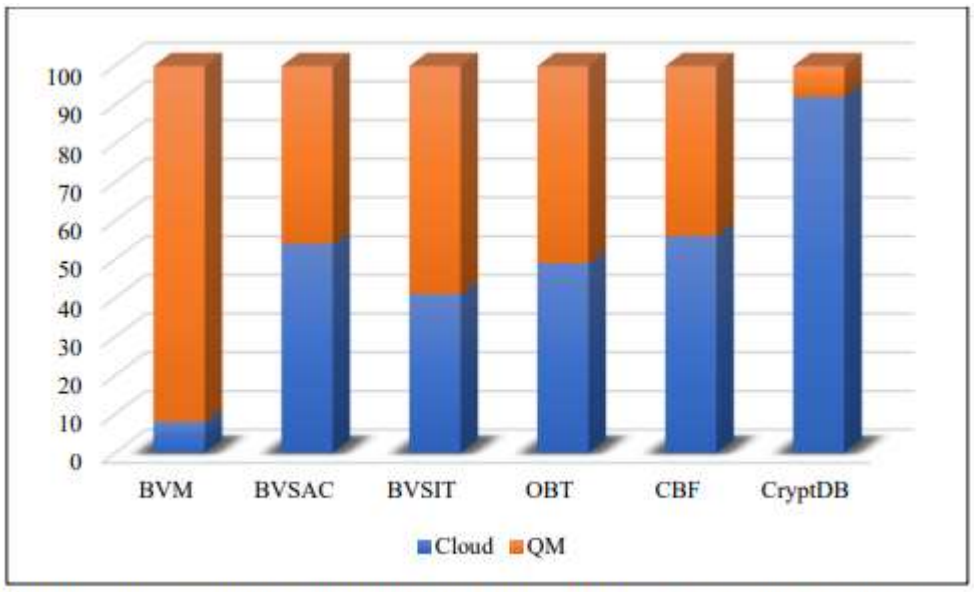


The average query translation cost at the QM for each model

طبق این نمودار ، BVM بیشـترین تأخیر را دارد (حدود ۵۲٪) زیرا بالاترین درصـد تولید و پردازش BV را ارائه می‌دهد ، پس‌ازآن BVSIT (حدود ۳۸٪) ، و BVSAC تنها ۱۰٪ از هزینه کل را گرفته است.

۳-۵ بررسی محاسبات Cloud و QM :

در این بررسـی سـعی شـده اسـت حجم محاسـبات در سرور ابری و QM محاسبه و مقایسه شود . نمودار زیر این مقدار را به‌صورت میانگین درصد زمان محاسبه نشان می‌دهد.



The percentage of the computation at the QM versus at the cloud server

چنانچه مشاهده می‌شود که CryptDB بیش از ۹۰٪ تأخیر محاسبات پردازش پرس‌وجو را به سرور منتقل کند ، و کمتر از ۱۰٪ تأخیر محاسبه را به QM واگذار می‌کند (QM معادل پروکسی در CryptDB و CBF است) .

همچنین می‌توانیم ببینیم که BVM بیشـترین هزینه محاسبات را در QM دارد زیرا علاوه بر عملیات رمزنگاری ، BV ها به‌صـورت محلی در QM پردازش می‌شـوند. مدل BVSIT به دلیل رمزگشـایی شـاخص‌های رمزگذاری شـده BV پس از واکشـی آن‌ها از سـرور حدود ۶۰٪ محاسـبات در QM را دارد. مدل OBT حدود ۵۳٪ محاسبه در QM را دارد زیرا QM به‌طور کامل رمزگشایی می‌کند ، حتی برای اگر دستور خواندن تمامی ستون‌ها نباشد.

از طرف دیگر ، BVSAC و CBF درصد نزدیک به هم دارند و حدود ۵۰٪ محاسبات به سرور منتقل شد زیرا در BVSAC همان‌طور که قبلاً توضیح داده شد ، BV ها در سرور جستجو می‌شوند و ما از پردازش آن‌ها در QM جلوگیری می‌کنیم. بعلاوه ، در CBF ، ترتیب مجموعه‌های مختلف شاخص‌ها و بازنویسی پرس‌وجو پس از انجام عملیات پیوندی بین مجموعه شاخص‌ها ، از علل اصلی به دست آوردن درصد محاسبات بالاتر در QM است.

۵-٤ بررسی فضای موردنیاز مدل‌ها :

در این بخش ، ما هر سیستم را ازنظر فضای موردنیاز در سرور ابری و QM ارزیابی می‌کنیم. ستون شاخص یک عدد صحیح است که به ۴ بایت نیاز دارد ، درحالی‌که بقیه ستون‌ها (ستون‌هایی که رمزگذاری می‌شوند) به ۳۳ بایت نیاز دارند (۳۲ بایت ازآنجاکه کلید مخفی ۳۲ بایت است و ۱ بایت برای ضبط طول نوع داده) علاوه بر این ، ما فضای موردنیاز برای ذخیره BV ها را در سرور برای هر مدل ارائه می‌دهیم.

Storage requirement per record at the server (cloud) of the encrypted student table

Name of the Attribute	Data type	BVM (bytes)	BVSAC (bytes)	BVSIT (bytes)	OBT	CBF	CryptDB
index	int	4	4	4		28	0
ID	varbinary	17	17	17		284	291
Name	varbinary	33	33	33	65	66	50
SSN	varbinary	17	17	17	+	284	291
Visa_Type	varbinary	17	17	17	24	34	50
Salary	varbinary	17	17	17		284	291
Department	varbinary	33	33	33		66	50
Bit vectors storages		0	4	37*	0	0	0
Total Space required		**202**	**206**	**239**	**89**	**1046**	**1023**

*37 = 20 bytes for storing bits as individual columns (we have 20 bits that require 20 columns of bit datatype) + 17 bytes for the encrypted index column

همان‌طور که در دیده می‌شود ، OBT کمترین نیاز ذخیره‌سازی را در سمت سرور دارد زیرا کل دیتا به یک بلوک تبدیل می‌شود و پس‌ازآن BVM و سپس BVSAC قرار دارد. BVSIT به فضای بیشتری نسبت به BVSAC نیاز دارد زیرا ما BV ها را به‌عنوان یک جدول مستقل و نه ستونی در همان جدول رمزگذاری شده مانند BVSAC ذخیره می‌کنیم. CBF برای هر ضبط حدود ۱۰۴۶ بایت نیاز دارد که بالاترین میزان فضای موردنیاز در بین سیستم‌ها است. CryptDB دومین بالاترین میزان نیاز به فضا را کسب می‌کند ، حدود ۱۰۲۳ بایت / رکورد.

فصل ششم

مقدمه

بحث در خصوص بانک‌های اطلاعاتی و دغدغه‌های افشای اطلاعات همواره پژوهشگران و صاحبان داده را به دنبال بررسی و انتخاب راهکارهای امن و مناسب کشانده و در این مسیر ایده‌های متفاوتی مطرح می‌شود چنانچه تا اینجا مدل‌ها و استراتژی‌های مختلف عنوان گردید و موردبررسی قرار گرفت.

مدل‌ها و ایده‌هایی که در این پایان‌نامه مطرح گردید را می‌توان به‌عنوان راهکارهای اولیه دانست زیرا دارای نقص و کمبودهایی هست، علی‌الخصوص که بانک‌های اطلاعاتی مانند SQL با فراهم آوردن گزینه رمزگذاری و رمزگشایی به‌عنوان دو ابزار درونی، این دغدغه را کاهش داده است.

در این تحقیق، تمامی تمرکز بر یک سیستم به‌عنوان واسط بنام QM بوده تا بتواند با ترجمه دستورات و درخواست‌ها به بانک اطلاعاتی و نگهداری اطلاعات به‌صورت رمزگذاری شده، از افشای آن توسط کاربران غیرمجاز و یا صاحبان سرورها جلوگیری نماید اما آنچه به‌عنوان راهکار در این QM در نظر گرفته‌شده دارای اشکالات و نقص‌های فراوانی هست و در بعضی بخش‌ها حتی به بن‌بست خواهیم رسید.

البته ایده و استراتژی کاهش حجم اطلاعات در فراخوانی و انتخاب اطلاعات را می‌توان ایده مناسبی دانست و در راهکارهای بعدی استفاده نمود. در ادامه راهکارها و ایده بهتری را جهت تحقیقات آتی اعلام می‌داریم.

٦-١ بحث و پیشنهادها:

عواملی که باعث افزایش کارایی سیستم می‌شوند:

تعداد پارتیشن‌های یک ستون در کارایی مدل‌های پیشنهادی ما نقش بسزایی دارد. هرچه تعداد پارتیشن‌ها بیشتر باشد ، رکوردهای کمتری از ابر گرفته می‌شود زیرا کارایی مدل‌ها تحت تأثیر تعداد رکوردهای رمزگذاری شده واکشی شده از ابر است.

ازآنجاکه ما ردیف‌ها را بر اساس تقسیم ویژگی‌ها به بیت کد می‌کنیم ، جستجوی بیت‌ها سریع است و ما نگران طولانی بودن PT نیستیم ، زیرا این امر به کاهش تعداد ردیف‌های رمزگذاری شده بازیابی شده از ابر کمک می‌کند. البته داشتن تعداد پارتیشن متعادل خود نیز چالشی هست که طراح و مالک داده‌ها با آن روبرو هست.

٦-١-١ بحث :

همان‌طور که در آزمایش‌های انجام‌شده مشاهده شد ، ما هر مدل پیشنهادی را ارزیابی کردیم. سپس ، ما مدل‌ها را ازنظر سرعت پردازش پرس‌وجو ، محاسبات سربار ، نیازهای ذخیره‌سازی در QM و سرور ابری مقایسه کردیم.

ازنظر سرعت ، مهم‌ترین دلیل تأخیر ، مقدار داده‌هایی است که باید رمزگشایی شود. بااین‌حال ، تمام نمونه‌های پیشنهادی در نظر دارند مقدار داده‌های رمزگذاری شده بازیابی شده از ابر را به کمتر از ۳۵ درصد کاهش دهند و همان‌طور که دیده شد علت ایجاد تغییر در سرعت پردازش پرسش به نحوه و محل ذخیره‌سازی BV مرتبط است.

تأخیر BVM عمدتاً تحت تأثیر انتقال BV از هارددیسک سخت به حافظه است ، درحالی‌که در BVSIT ، ما با ذخیره‌سازی BV ها به‌عنوان یک جدول رابطه‌ای مستقل در ابر ، از این موضوع جلوگیری می‌کنیم. در BVSAC نیز ما BV ها را به‌عنوان یک ستون اضافی در همان جدول رمزگذاری شده در ابر ذخیره کردیم.

برای سیستم‌های تک کاربر که توسط فضای ابری محدودشده‌اند ، BVM مدل بهینه است زیرا برای مدیریت و پردازش جداول رابطه‌ای رمزگذاری شده به حداقل فضای اضافی نیاز دارد . بااین‌حال ، اگر بیش از یک کاربر منبع محاسبات مشترک را به اشتراک بگذارد ، محاسبه سربار محاسبه و میزان مصرف حافظه در QM دچار مشکلاتی خواهد بود ، زیرا برای اجرای یک پرس‌وجو ، کل مجموعه BV های جدول باید در حافظه اصلی بارگیری شوند . بنابراین ، BVM برای یک سیستم چندکاربره توصیه نمی‌شود ، و مدل مطلوب BVSAC است که از روش BVSIT استفاده می‌نماید. همچنین BVSAC کمترین زمان اجرا ، دومین کمترین میزان فضای موردنیاز در ابر و کمترین سیستم موردنیاز برای ذخیره‌سازی را در QM دارا هست ، به این معنی که برای سیستم‌های تک کاربرِ و چندکاربره مناسب است.

ازنظر امنیت ، همه مدل‌های پیشنهادی در بالاترین سطح امنیتی بودند. در CBF ، اگر ارائه‌دهندگان سرویس ابر تبانی نکنند ، امنیت بالابود. به‌هرحال تضمین‌شده نیست و رتبه سطح امنیت پایینی دارد .

در CryptDB ، نویسندگان پیشنهاد نکردند لایه‌های رمزگذاری را عقب بگذاریم که باعث کاهش سطح امنیت شود.

مدل OBT در صورت استفاده از الگوریتم رمزگذاری تصادفی ، ایمن بود وگرنه سطح امنیت کاهش پیدا می‌کند. برای جداول با ستون‌های حساس زیاد ، مانند هزاران ، عملکرد OBT به خطر می‌افتد زیرا داده‌های تمام زمینه‌های هر سطر در یک بلوک به‌هم‌پیوسته‌اند.

در CBF ، هزاران سرویس ابری برای ذخیره ستون‌های توسعه‌یافته موردنیاز بود ، که این سیستم را غیرعملی می‌کند. در مقابل ، سیستم‌های پیشنهادی چنین مشکلی ندارد زیرا ما رمزگذاری مبتنی بر سلول را انجام دادیم. ما تمام داده‌ها را در همان سرور ابری ذخیره کردیم که باعث تسریع در زمان پردازش جستجو برای پایگاه‌های مقیاس‌پذیر می‌شود .

٦-١-٢ پیشنهادها :

آنچه در این پایان‌نامه بررسی و موردبحث قرار گرفت راهکارها و مدل‌های ایجاد و ارتباط با یک بانک اطلاعاتی رمزگذاری شده، بوده و به‌عنوان راهکار کلی برای تمامی انواع پایگاه اطلاعاتی رابطه‌ای قابل انجام هست. امروزه شرکت‌های ارائه‌دهنده پایگاه‌های داده رابطه‌ای مانند SQL با ایجاد دستورات و ابزارهای آسان و قابل‌فهم‌تر، قدم بزرگی در ایجاد بانک‌های اطلاعاتی رمزگذاری شده برداشته‌اند به صورتی که دیگر نیاز به یک نرم‌افزار واسط نباشد.

آن چیزی که در این پایان‌نامه ایدئال بود، ایده کاهش تعداد ردیف پردازش رمزگذاری هست زیرا حتی با داشتن ابزارهایی مانند آنچه بیان شد، باز باید تمامی ردیف‌ها مورد پردازش رمزگشایی قرار گیرد ، پس ایده کاهش تعداد ردیف جهت پردازش نهایی به‌عنوان ایده مناسبی می‌تواند در کنار این ابزارها و امکانات منجر به ایجاد یک بانک و سرویس اطلاعاتی امن گردد.
این نکته را باید در نظر بگیریم که عملیات‌های پایگاه داده خیلی گسترده‌تر و متنوع‌تر از آن چیزی است که در این پایان‌نامه موردبررسی قرار گرفت و در مدل‌های مختلف اجرا گردید مثلاً عملیات‌هایی مانند Like که قابلیت جستجو در محتوای فیلدهای کاراکتری هست در این پایان‌نامه قرار نگرفته و عملاً برای اجرای آن نیاز به رمزگشایی در تمامی اطلاعات هست که نا گریز منجر به هزینه بالای اجرا می‌گردد.
همچنین ایده نرم‌افزار واسط به‌عنوان یک ایده مناسب برای کار با بانک اطلاعاتی رمزگذاری شده را می‌توان به‌صورت یک حافظه موقت رمزگشایی‌شده طراحی و اجرا نمود چنانچه این نرم‌افزار هر زمان که اطلاعاتی از جدولی درخواست شود آن را به‌طور کامل رمزگشایی و در یک حافظه موقت قرار دهد و تمامی تغییرات هم‌زمان بر حافظه موقت و بانک اطلاعاتی لحاظ گردد. در این حالت بار هزینه تنها یک‌بار برای مدت اجرای حافظه موقت اتفاق میفتد و در تمامی ارجاعات، این نرم‌افزار با اطلاعات رمزگشایی‌شده کار خواهد کرد و دسترسی به آن نیز پس از گذراندن مجوزهای لازم هست.

۶-۲ نتیجه‌گیری :

رایانش ابری یک محیط محاسباتی جذاب برای انواع کاربران و شرکت‌ها است. اما ، نقض حریم خصوصی ، نه‌تنها توسط مهاجمان مخرب بلکه توسط ارائه‌دهندگان کنجکاو ، نقطه‌ضعف این نوع خدمات است ، زیرا کاربران کنترل دسترسی بر داده‌های خارج از منابع را از دست می‌دهند. راه‌حل‌های زیادی برای این مشکل وجود دارد و رمزگذاری داده‌ها یک راه‌حل مؤثر است. بااین‌حال ، اجرای نمایش داده‌های SQL بر روی داده‌های رمزگذاری شده چالش‌برانگیز است ، به‌خصوص اگر از الگوریتم رمزگذاری تصادفی مانند AES-CBC برای رمزگذاری استفاده شود. در این تحقیق ، ما ابتدا QM را معرفی می‌کنیم ، یک سرور قابل‌اعتماد ، که به‌عنوان واسطه بین سرور ابری و کاربر (ها) کار می‌کند و همه فرایندهای رمزنگاری را انجام می‌دهد. علاوه بر این ، ما یک تکنیک جدید برای نمایه‌سازی بر اساس پارتیشن‌های از پیش تعریف‌شده برای هر ویژگی حساس طراحی می‌کنیم ، و سپس هر ردیف از داده را به‌صورت بیتی رمزگذاری می‌کنیم .
از بیت‌ها برای بازیابی ردیف‌های انتخابی برای یک پرس‌وجو خاص استفاده می‌شود که دامنه تاپل های رمزگذاری شده بازیابی شده را به حداقل می‌رساند. بر اساس این طرح رمزگذاری ، ما سه سیستم ایمن مختلف را پیشنهاد داده‌ایم که هر یک از آن‌ها برای ذخیره و نگهداری داده‌های شاخص (به‌عنوان‌مثال ، BV) یا به‌صورت محلی در QM یا با انتقال BV ها به فضای ابری ، از روش متفاوتی استفاده می‌کنند.
برای هر نمونه اولیه پیشنهادی ، ما الگوریتم‌های مختلفی را برای انجام پرس‌وجو از عملگرهای مختلف جبر رابطه‌ای SQL طراحی می‌کنیم و آن را در برابر سناریوهای حمله ، مانند حملات استنتاج ، مقاوم می‌کنیم. ما مدل‌های خود را با پیاده‌سازی آن‌ها و مقایسه عملکرد آن‌ها با سیستم‌های پیشرفته مانند CryptDB ، آزمایش می‌کنیم.
ما آن‌ها را ازنظر نیاز به زمان اجرا و فضای موردنیاز ارزیابی می‌کنیم. درمی‌یابیم که سیستم‌های پیشنهادی در مقایسه با اکثر سیستم‌های رقیب ، به زمان اجرا و فضای کمتری نیاز دارند.

منابع و مراجع

فهرست منابع غیرفارسی و اینترنتی

۱) R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan, “CryptDB: Processing queries on an encrypted database.” Commun. ACM, vol. ۵۵, no. ۹, pp. ۱۰۳–۱۱۱, ۲۰۱۲.

۲) Y. D. Jang and J. H. Kim, “A comparison of the query execution algorithms in secure database system,” Int. J. Electr. Comput. Eng., vol. ٦, no. ۱, pp. ۳۳۷–۳٤۳, ۲۰۱٦.

۳) W. Wang, Y. Hu, L. Chen, X. Huang, and B. Sunar, “Exploring the feasibility of fully homomorphic encryption,” IEEE Trans. Comput., vol. ٦٤, no. ۳, pp. ٦۹۸–۷۰٦, ۲۰۱۳.

٤) A. Alsirhani, P. Bodorik, and S. Sampalli, “Improving database security in cloud computing by fragmentation of data,” in ۲۰۱۷ International Conference on Computer and Applications (ICCA), ۲۰۱۷, pp. ٤۳–٤۹

٥) H. Hacigümü\cs, B. Iyer, C. Li, and S. Mehrotra, “Executing SQL over encrypted data in the database-service-provider model,” in Proceedings of the ۲۰۰۲ ACM SIGMOD international conference on Management of data, ۲۰۰۲, pp. ۲۱٦–۲۲۷.

٦) H. Hacigümü\cs, B. Iyer, C. Li, and S. Mehrotra, “Executing SQL over encrypted data in the database-service-provider model,” in Proceedings of the ۲۰۰۲ ACM SIGMOD international conference on Management of data, ۲۰۰۲, pp. ۲۱٦–۲۲۷.

۷) A. Boicea, F. Radulescu, C.-O. Truica, and C. Costea, “Database encryption using asymmetric keys: a case study,” in ۲۰۱۷ ۲۱st International Conference on Control Systems and Computer Science (CSCS), ۲۰۱۷, pp. ۳۱۷–۳۲۳.

۸) O. M. Ben Omran and B. Panda, “A new technique to partition and manage data security in cloud databases,” in The ۹th International Conference for Internet Technology and Secured Transactions (ICITST-۲۰۱٤), ۲۰۱٤, pp. ۱۹۱–۱۹٦.

۹) S. Cui, M. R. Asghar, S. D. Galbraith, and G. Russello, “P-McDb: Privacy-preserving search using multi-cloud encrypted databases,” in ۲۰۱۷ IEEE ۱۰th International Conference on Cloud Computing (CLOUD), ۲۰۱۷, pp. ۳۳٤–۳٤۱

۱۰) O. M. Omran, “Data Partitioning Methods to Process Queries on Encrypted Databases on the Cloud,” ۲۰۱٦.

۱۱) V. H. Hacigumus, B. R. Iyer, and S. Mehrotra, “Query optimization in encrypted database systems.” Google Patents, Mar-۲۰۱۰.

۱۲) V. H. Hacigumus, B. R. Iyer, and S. Mehrotra, “Query optimization in encrypted database systems.” Google Patents, Mar-۲۰۱۰.

۱۳) “Package javax.crypto.” [Online]. Available: https://docs.oracle.com/javase/۷/docs/api/javax/crypto/package-summary.html#package_description

ABSTRACT

Nowadays, the issue of maintaining and securing databases with sensitive and strategic information at the level of a company and sometimes even at the level of countries, is studied and researched by software and data researchers. Part of this security is the prevention of damage and disclosure of information by hackers and sabotage groups, and another worrying part is the disclosure of information by users and providers of cloud services because the second group of hardware and software layers common for security , Have and have free access to information resources.

Cloud servers and users of a system, due to their free access to information, can easily access the classified and sensitive information of a database and have serious consequences for the owners of the database. The best and perhaps the only idea that comes to mind is to encrypt information so that it can only be displayed after controlling access within the software.

But encryption and consequently decryption require heavy calculations and server busy time, and if we can not advance this method with the right and appropriate method, in the high volume of information and after a while, the use of the system becomes difficult and even impossible.

Familiarity with the types of encryption methods, the amount of load and the cost of its implementation and determining the correct method for encryption, helps us to reach a suitable solution and method to maintain the confidentiality of information.