



**دانشگاه پیام نور تهران**

**واحد شهر ری**

**سمینار، سمینار تحقیق و تتبع نظری**

**عنوان:**

**مدل‌های ایمن و کارآمد برای دريافت و پردازش اطلاعات**

**از پایگاه های رمزگذاری شده در فضاي ابري**

**استاد راهنما:**

**جناب آقای دکتر سید علی رضوی ابراهیمی**

**نگارش:**

**مجيد لطفي**

**تابستان 1400**

چکیده

امروزه بحث حفظ و امنيت بانک‌های اطلاعاتي با داشتن اطلاعات حساس و استراتژيك در سطح يك شركت و حتي بعضا در سطح كشورها، موردبررسی و تحقيق پژوهشگران بخش نرم‌افزار و داده هست.

بخشي از اين امنيت جلوگيري از آسيب و افشاي اطلاعات از سمت هكرها و گروه هاي خرابكاري بوده و بخش نگران كننده ديگر ، افشاي اطلاعات توسط كاربران و ارائه‌دهندگان سرويس هاي ابري هست زيرا گروه دوم از لایه‌های سخت‌افزاري و نرم‌افزاري رايج براي امنيت، گذر كرده و دسترسي آزاد به منابع اطلاعاتي دارند.

سرويس دهندگان ابري و كاربران يك سيستم ، به علت داشتن دسترسي آزاد به اطلاعات، براحتي می‌توانند به اطلاعات طبقه بندي شده و حساس يك بانك اطلاعاتي دسترسي پيدا نمايند و تبعات سنگيني را براي صاحبان بانك اطلاعاتي ايجاد كنند. بهترين و شايد تنهاترين ايده اي كه به ذهن مي رسد رمزگذاري اطلاعات هست تا اين اطلاعات تنها پس از كنترل دسترسي ها داخل نرم‌افزار، به نمايش دربيايند.

اما رمزگذاري و به تبع آن رمزگشايي نياز به محاسبات سنگين و زمان مشغولي سرور دارند و اگر نتوانيم با روش صحيح و مناسب اين روش را جلو ببريم، در حجم اطلاعات بالا و پس از مدتي ، استفاده از سامانه سخت و حتي غيرمقدور می‌شود.

آشنايي با انواع روش‌های رمزگذاري ، ميزان بار و هزينه اجرا آن و تعيين روش صحيح براي رمزگذاري، به ما كمك مي كند به يك راهكار و روش مناسب براي حفظ محرمانگي اطلاعات برسيم.

**کلمات کلیدی:** بانك اطلاعاتي ، رمزگذاري ، رمزگشایی ، سرور ابري ، سرویس‌دهنده ابري

فهرست مطالب

[عنوان](#_Toc413125201) صفحه

فصل اول: مقدمه

[مقدمه 6](#مقدمه)

[1-1 تعريف مسئله و سؤال](#تعريف_مسئله_و_سوال) 6

[1-2 ضرورت هاي تحقيق 7](#ضرورت_تحقيق)

[1-3 اهداف](#اهداف) 7

[1-4 جمع‌بندی 7](#جمعبندي_فصليك)

فصل دوم: آشنايي با رمزگذاري

[مقدمه 8](#مقدمه_رمزگذاري)

[2-1 روش‌های رمزگذاري](#روش_رمزگذاري) 8

[2-2 رمزگذاري متقارن 9](#رمزگذاري_متقارن)

[2-3 رمزگذاري نا متقارن 9](#رمزگذاري_نامتقارن)

فصل سوم: رمزگذاري در پايگاه داده

[مقدمه 11](#مقدمه_فصلسوم)

[3-1 رويكردهای فعلي رمزگذاری در پايگاه داده 11](#رويكردهاي_فعلي)

3-2 معرفي CryptDB 12

[3-3 معرفي FHOPE 12](#معرفي_FHOPE)

3-4 معرفي P-McDb 12

3-5 معرفي SDB 12

[3-6 معرفي CASB](#معرفي_CASB) 13

[3-7 ساير تحقيقات 13](#ساير_تحقيقات)

فصل چهارم: استراتژی‌های رمزگذاري پايگاه داده

[مقدمه 14](#مقدمه_فصلچهارم)

[4-1 استراتژي رمزگذاري 14](#استراتژي_رمزگذاري)

[4-2 روش‌های قابل‌اجراي رمزگذاري](#روشهاي_قابل_اجراي_رمزگذاري) 17

فصل پنجم: آزمايش و ارزيابي

[مقدمه 21](#مقدمه_فصلپنجم)

5-1 آزمايش و ارزيابي 21

[5-2 ارزيابي 22](#ارزيابي)

5-3 بررسي محاسبات Cloud و QM 32

[5-4 بررسي فضاي موردنیاز مدل‌ها 33](#بررسي_فضاي_موردنياز_مدل)

فصل ششم: بحث و پیشنهاد‌ها

[مقدمه 34](#مقدمه_فصلششم)

6-1 بحث و پيشنهاد ها 34

[6-2 نتيجه گيري 36](#نتیجه_گیری)

**فصل اول**

**مقدمه**

امروزه اكثر شرکت‌ها به‌موازات اشتراك فايل در شبکه‌های اجتماعي و فضاي ابري ، به سمت استفاده از فضاي ابري جهت نگهداري اطلاعات و ديتاي خود رفته‌اند زيرا از سوئي می‌توان با اجاره سرویس‌های آماده ، هزينه تهيه سرور را كاهش داد و از طرف ديگر امكان دسترسي هر زمان از ديتاها را فراهم نمود .

به‌موازات گسترش استفاده از اين بستر ، آسیب‌ها و مشكلات آن نيز مطرح‌شده است و بايد بتوان اين سرويس را در امنيت و با كمترين مشكل اجرا نمود . اين مشكلات گاها مربوط به حملات به سرور و سیستم‌های عامل هست با قصد از كار انداختن سرویس‌ها و گاها مربوط به نفوذ غیرمجاز به دیتاها هست پس جدا از بحث‌های امنيت شبكه و سیستم‌عامل ، خود ديتا نيز حفاظت شود.

حفاظت از داده خارج از روش‌های نرم‌افزاری مانند تعريف نقش‌ها در پايگاه داده و كنترل اين نقش‌ها ، بايد با روش‌هایي مانند رمزگذاري حفاظت شود زيرا سرویس‌دهنده‌های و مالكان سرورهاي ابري امكان دسترسي به ديتاي مشتريان خود را داشته می‌توانند از اين اطلاعات حساس استفاده غیرمجاز نمايند.

رمزگذاری در دو بخش ساختار و ديتا قابل‌اجرا هست و اجرا هم‌زمان هر دو بخش می‌تواند بالاترين امنيت را ايجاد نمايد زيرا با فرض امكان و يا افشاي دسترسي به پايگاه داده ، داشتن نام جداول و يا ستون‌ها متداول باعث آسیب‌پذیری بيشتر اطلاعات می‌شود براي مثال اگر جدولي با نام Banks داشته باشيم قطعاً بیان‌کننده نام بانک‌ها هست درحالی‌که اگر همين جدول به‌صورت T12 باشد ، امكان افشاي اطلاعات كاهش و حتي غیرممکن می‌شود.

1-1 تعريف مسئله و سؤال

با توجه به آنچه تا اينجا بيان شد، مسئله اصلي ما حفاظت داده‌ها از بالاترين سطح كه سرویس‌دهنده ابري هست و دسترسي كامل به اطلاعات دارد تا پایین‌ترین سطح كه افراد غیرمجاز و هكرها هست ، تعريف می‌شود. اين حفاظت به‌گونه‌ای هست كه حتي اگر شخص بتواند ديتا را در به‌صورت مستقيم از جداول مشاهده نمايد، باز با خطر افشاي اطلاعات مواجه نشويم.

سؤالاتی كه تا پايان اين تحقيق پاسخ داده می‌شود را این‌گونه بيان می‌کنیم.

1. انواع روش‌های رمزگذاري قابل‌اجرا در پایگاه‌های داده کدم‌اند؟
2. انواع رویکردهای رمزگذاري در پایگاه‌های داده کدم‌اند؟
3. بررسي فرآیندها و عملیات پايگاه داده رمزگذاري شده چگونه هست؟
4. بررسي و مقايسه روش‌ها و رويكردهاي رمزگذاری ازنظر هزينه اجرا و نگهداري چگونه‌اند؟

1-2 ضرورت تحقيق

تا اينجا با مسائل و مشكلات يك پايگاه داده ازنظر افشاي اطلاعات آشنا شديم و رمزگذاري به‌عنوان راهكار آن معرفي گرديد. حال بايد اين موضوع رو در نظر بگيريم كه صرف رمزگذاري و ايجاد لایه‌های رمزگذاري مسئله ما نیست زيرا هرچقدر لایه‌های رمزگذاري اضافه شود ، به همان اندازه لایه‌های به دست آوردن ديتاها و هزينه دريافت اطلاعات براي خودمان بالاتر می‌رود زيرا اين عبارات رمزگذاري شده بايد براي پردازش و استفاده مجدد ، رمزگشايي شوند و ما بايد سعي كنيم رمزگذاري را بر روي ديتاي حساس لحاظ نماييم و نه بر روي تمام پايگاه داده.

ايده و نگرش ديگري كه در اين پایان‌نامه بر آن تأکید شده، كاهش بار محاسبات رمزگذاري و رمزگشایی هست، به اين صورت كه در هنگام دريافت اطلاعات در يك پرس‌وجو، نياز به رمزگشایی اطلاعات نامرتبط نباشد و سعي شود تا حدودي از ميزان اطلاعات واكشي جهت رمزگشایی كاسته شود.

1-3 اهداف

با آنچه تا اينجا در خصوص افشاي اطلاعات و ضرورت حفظ آن بيان شد ، ما در اين تحقيق چند هدف را موردبررسی قرار می‌دهیم و سعي می‌کنیم در انتها با جمع‌بندی روش و رويكردها رايج در اين خصوص بپردازيم . اين اهداف عبارت‌اند از :

1. آشنايي با روش‌ها رمزگذاري بدون در نظر گرفتن پايگاه داده
2. بررسي و اجرا روش رمزگذاري مؤثر و رايج در پايگاه داده
3. آشنايي با مدل‌ها و اصول قابل‌اجرا در رمزگذاري پايگاه داده
4. مقايسه مدل‌ها و انتخاب مدل مناسب

1-4 جمع‌بندی

در اين تحقيق به‌ضرورت حفظ اطلاعات پايگاه داده از افشاي آن و رمزگذاري به‌عنوان بهترين روش جهت انجام آن معرفي می‌گردد. اين موضوع چنانچه قبلاً نيز اشاره شد، صرفاً براي جلوگيري از حملات مخرب و غیرمجاز نیست و بررسی‌های اين تحقيق در خصوص كاربران و اشخاصي هست كه به پايگاه داده ، دسترسي مجاز دارند.

ابتداها به روش‌های رمزگذاري ، اصول و قواعد رايج در آن و سپس به رويكردهاي رايج در ايجاد يك پايگاه رمزگذاري شده می‌پردازیم و درنهایت مدل‌ها و روش‌های معرفی‌شده را مقايسه تا بهترين مدل و روش را انتخاب نماییم.

**فصل دوم**

**مقدمه**

رمزگذاري روند رمز کردن پیام‌ها یا اطلاعات است به‌گونه‌ای که تنها افراد مجاز قادر به خواندن آن باشند. پیام یا اطلاعات با استفاده از یک الگوریتم، رمزگذاری شده و علائم رمزی به وجود می‌آید که فقط در صورت رمزگشایی قابل‌خواندن هستند. در رمزگذاری معمولاً یک کلید رمزگذاری شبه تصادفی تولیدشده توسط یک الگوریتم، به کار گرفته می‌شود. اگرچه شاید رمزگشایی پیام بدون در اختیار داشتن کلید ممکن باشد، اما در یک رمزگذاری خوب، منابع محاسباتی زیادی برای این کار لازم است. یک گیرنده مجاز به‌راحتی می‌تواند پیام را با کلید تدارک دیده‌شده توسط صادرکننده پیام، رمزگشایی کند اما گیرنده غیرمجاز نمی‌تواند.

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

**2-1 روش‌های رمزگذاري :**

رمز عبارت است از تبدیل کاراکتر به کاراکتر یا بیت به بیت بدون آن‌که به محتویات زبان‌شناختی (ادبیات) آن پیام توجه شود. رمزگذاری دیجیتال، متن قابل‌خواندن (که اصطلاحاً به آن plaintext می‌گویند) را می‌گیرد و آن را به متني غیرقابل تشخيص تبديل می‌نماید و اصطلاحاً آن را رمزگذاری می‌کند. البته برای انجام چنین کاری، الگوریتم‌های بسیار پیچیده‌ و قدرتمند وجود دارد.

این الگوریتم‌ها از متغیرهایی به نام کلید بهره می‌برند که پیچیدگی رمزگذاری را چندین برابر می‌کنند. این کلیدها به‌طور تصادفی تولید می‌شوند و منحصربه‌فرد هستند. یعنی اگر یک هکر بخواهد بانفوذ به یک بانک داده اطلاعات کارت اعتباری کاربران را بدزدد، نه‌تنها باید از الگوریتم استفاده‌شده خبر داشته باشد بلکه باید بداند کدام کلید استفاده‌شده است. چنین کاری به‌هیچ‌عنوان کار راحتی نیست و به همین خاطر رمزگذاری‌های دیجیتال در برابر حملات جستجوی فراگیر (brute force) مقاومت بسیار بالایی دارند.

دو روش براي رمزگذاري قابل‌اجرا هست : روش متقارن و روش نامتقارن.

**2-2 رمزگذاري متقارن :**

در روش متقارن تنها يك كليد براي رمزگذاري 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 70 تنها نوع رمزنگاری به‌حساب می‌آمد.

برخي از الگوریتم‌های متداول در رمزگذاری بر مبناي روش متقارن به‌صورت ذيل هست:

1. Order-Preserving Encryption: رمزگذاری با حفظ ارزش عبارت رمزگذاري نسبت به اصل عبارت مثلاً اگر رمزگذاری دو عبارت A, B بشود E1, E2 و A>B باشد قطعاً E1>E2 و برعكس و اين مقايسه بدون رمزگشايي معتبر هست و مناسب براي ديتاي عددي و اين مزيت خود منجر به ضعف شده و می‌توان توسط استنباط تا حدودي به اطلاعات اصلي رسيد.
2. Deterministic Encryption: يا همان DES الگوريتم رمزگذاری‌ای هست كه توسط يك كليد ، همواره يك عبارت با به متن مشخصي رمز می‌نماید و متأسفانه ارزش عبارت اصلي را حفظ نمی‌نماید و در صورت افشاي كليد ، به‌راحتی می‌توان ساير عبارات را رمزگشايي كرد! با شکسته شدن الگوریتمDES این استاندارد در سال ۱۹۹۸ تمدید نشد.
3. Advanced Encryption Standard: یک الگوریتم رمزنگاری بلوک جایگزینی شبکه (SPN) است و جايگزين 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-128، AES-192 و AES-256 است. هر رمز، عملیات رمزنگاری و رمزگشایی را بر روی بلوک‌های 128 بیتی داده، با استفاده از کلیدهای 128، 192 و 256 بیتی خود انجام می‌دهد و محبوبيت بالايي دارد.
4. 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.

**2-3 رمزگذاري نامتقارن :**

در روش نامتقارن از دو كليد استفاده می‌نماید ، يك كليد عمومي و يك كليد خصوصي كه كليد عمومي يك عبارت رمزگذاري شده جهت رمزگذاري در سطح كاربران هست كه خود كليد عمومي توسط كليد خصوصي رمزگذاري شده است. اين سطح از رمزگذاري منجر به كاهش سرعت و افزايش هزينه استفاده از سيستم می‌شود.

برخي از الگوریتم‌های متداول در رمزگذاری بر مبناي روش نامتقارن به‌صورت ذيل هست:

1. Homomorphic Encryption: نوعي رمزگذاري براي انجام عملیات جمع و ضرب بدون باز کردن عبارت رمزگذاري شده و يا داشتن كليد و مناسب ديتاي عددي و داراي قابليت اجراي دستورات SQL بر روي عبارات رمزگذاري.
2. Randomized Encryption: اين رمزگذاري يك عبارات را هر بار به عبارت رمزگذاري متفاوتي تبديل می‌نماید و بالاترين سطح امنيت هست اما به علت نوع رمزگذاري قابليت اجراي دستورات SQL بر روي عبارات رمزگذاري را ندارد.
3. Onion Encryption: در اين رمزگذاري از لایه‌های مختلف با متدهاي مختلف رمزگذاري استفاده می‌شود به صورتی كه لايه داخلي با متد رمزگذاري امنيت پايين و لايه بيروني داراي رمزگذاري با امنيت بالا و مشكل آن كند شدن پردازش پرس‌وجو و طولاني شدن عبارات رمزگذاري نسبت به عبارت اصلي هست.
4. RSA: در این‌چنین سیستم‌های رمزنگاری، کلید رمزگذاری عمومی است و از کلید رمزگشایی که مخفی است، جداست. هرکسی می‌تواند از این کلید عمومی برای رمزگذاری یک پیام استفاده کند، اما تنها کسی که آن دو عدد اولی که کلید بر اساس آن‌ها ساخته‌شده را می‌داند، قادر به رمزگشایی پیام است. شکستن رمزگذاری RSA به مسئله‌ی RSA معروف است. تاکنون هیچ روشی برای شکست دادن این سیستم( در صورت استفاده‌ی کلید به‌اندازه‌ی کافی بزرگ) منتشرنشده است. RSA به‌صورت نسبی، الگوریتم کندی است و به همین علت، کمتر برای رمزگذاری مستقیم اطلاعات کاربر استفاده می‌شود.
5. ElGamal: خصوصیت این الگوریتم به‌گونه‌ای است که در هر مرحله از فرایند رمزنگاری یک کلید تصادفی k تولید می‌شود مقدار این کلید کاملاً تصادفی بوده و در هر مرحله از اجرای فرایند رمزنگاری کلید تولیدشده با کلید قبلی متفاوت خواهد بود. این خصوصیت موجب می‌شود در دو مرحله متفاوت خروجی متفاوت تولید شود, که این خصوصیت 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 C1 امکان‌پذیر نباشد.

با توجه به تنوع اين الگوریتم‌ها ما بايد با در نظر گرفتن هزينه اجرا ، سرعت و ميزان حساسيت اطلاعات ، در بخش‌های مختلف پايگاه داده و سيستم نرم‌افزاری ، الگوريتم کاربردی‌تر و مناسب‌تر را انتخاب نماییم .

**فصل سوم**

**مقدمه**

پايگاه داده با توجه به ثبت اطلاعات به‌صورت متن ساده و داشتن اطلاعات حساس ، نيازمند بررسي و مطالعه در جهت حفظ اطلاعات از افشاي آن توسط كاربران و صاحبان سرورها دارد. البته نسخه‌های بالاتر بانک‌های اطلاعاتي داراي دستورات جهت رمزگذاري و رمزگشایی هست اما ما به دنبال رويكرد و الگوریتم‌هایی هستيم تا بتوانيم بر روي تمامي پایگاه‌های داده اجرا نماییم.

امروزه نرم‌افزارهایی به‌صورت سرویس‌دهنده بانک‌های اطلاعاتي كه نقش واسطه بين كاربر و بانك اطلاعاتي دارند، با ايجاد قابليت مديريت کدگذاری بانك اطلاعاتي توانسته‌اند اين موضوع را مديريت كنند اما اگر بخواهيم بانك اطلاعاتي خود را با يك استراتژي قابل تنظيم ، از افشاي اطلاعات محافظت نماييم ، نيازمند آشنايي با رويكردهاي فعلي و تحقيقات انجام‌شده در اين زمینه‌داریم.

**3-1 رويكردهاي فعلي رمزگذاري در پايگاه داده :**

يكي از روش‌های رمزگذاري يك پايگاه داده ، تبديل ركوردها به بلوك رمزگذاری هست به اين صورت كه هر رديف از اطلاعات با هر تعداد از ستون ، تبديل به يك ديتا رمزگذاري شده در جدول رمزگذاري شود اما اين روش امكان اجراي دستورات SQL را بر روي ديتاهاي رمزگذاري شده را از بين می‌برد و حتماً بايد ديتا رمزگشايي و دستور اجرا شود و اين در اطلاعات با ركوردهاي بالا منجر به هزينه و زمان بالا می‌شود.

براي حل اين مشكل محققان روشي را در نظر گرفتند به اين صورت كه ستوني ديگر به اين جداول رمزگذاري شده اضافه گردد كه در آن دسته‌بندی ردیف‌ها مشخص باشد و در زمان دريافت اطلاعات ، با توجه به دسته‌بندی موردنظر ، ما تعداد ركورد كمتري را دريافت و براي رمزگشايي پردازش نماييم. اين دسته‌بندی‌ها درجایی به‌صورت رمزگذاري شده نگهداري می‌شود.

البته اين روش هم امكان افشاي اين دسته‌بندی‌ها و سنگین‌تر شدن پردازش‌ها را دارد.

در اينجا بعضي سیستم‌های رمزگذاري در پايگاه داده معرفي می‌گردد كه داراي امنيت و كاركرد مناسبي می‌باشند.

**3-2 معرفي CryptDB :**

اين سيستم به‌عنوان اولین سیستم عملی برای اجرای دستورات SQL از طریق پایگاه‌های داده رمزگذاری شده توسعه داد شد كه توسط Onion Encryption طراحي گرديده 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. (نمايي از نوع رمزگذاري و پردازش‌ها در اين سيستم)

اين نوع رمزگذاري هزينه محاسبات رمزگذاري و رمزگشايي را بالا می‌برد البته اين موضوع توسط MONOMI حل‌شده است به اين صورت كه اجراي دستورات به دو بخش تقسیم‌شده‌اند : دستوراتي كه بر روي اطلاعات رمزگذاري شده قابل‌اجرا هست و دستوراتي كه بر روي اطلاعات رمزگشایی‌شده در سمت كاربر اجرا می‌شود. همچنين براي بهبود سرعت رمزگذاري ، از الگوريتم AES-NI بجاي AES استفاده‌شده است كه از روش‌های بهتري براي اجراي رمزگذاري AES استفاده می‌نماید.

**3-3 معرفي FHOPE :**

اين سيستم از الگوريتم homomorphic باقابلیت order-preserving برای اجرای دستورات SQL در اطلاعات عددي استفاده می‌کند و همان‌طور كه قبلاً گفته شد قابليت order-preserving به ما اين امكان را می‌دهد تا بدون رمزگشايي بتوانيم بر روی‌داده‌های رمزگذاري شده کارکنیم. البته اين سيستم بر روي تعداد ركورد كم آزمون شده است و بايد در ديتاهاي بزرگ‌تر آزمون شود.

**3-4 معرفي P-McDb :**

همان‌طور كه گفته شد قابليت اجراي دستورات بر روي ديتاهاي رمزگذاري شده بعضاً منجر به افشاي اطلاعات از طريق استنتاج می‌شود. اين سيستم براي جلوگيري از اين حمله ، توسط دو سرور كه يكي براي ذخیره‌سازی و جستجوي اطلاعات و ديگري براي تصادفي سازي و تغيير شكل پايگاه داده كه اين خود منجر به‌کندی سيستم می‌شود. اين سيستم نيز بجاي جستجوي كلي بر روي ديتاهاي رمزگذاري شده ، از جستجوي جزئي استفاده می‌نماید. همچنين در لغو دسترسي يك كاربر به بانك اطلاعاتي برخلاف ساير طرح‌های SSE (Server Sent Events)، نياز نیست تمامي اطلاعات مجدد رمزگذاری شود بلكه به‌تمامی CSP (Content Security Policy) اطلاع داده می‌شود و دسترسي كاربر بسته می‌شود و حتي اگر كاربر كليد رمزگشایی و دسترسي به يكي از CSP ها را داشته باشد نيز امكان دسترسي به اطلاعات را نخواهد داشت.

**3-5 معرفي SDB :**

يكي از راه‌های افزايش سرعت و کارایی سيستم در مواجهه با رمزگذاري داده‌ها ، تقسيم داده به حساس و غير حساس و رمزگذاري بر روي ديتاهاي حساس هست و اين سيستم از اين روش استفاده می‌نماید. داده حساس نيز به دو بخش اشتراكي تقسيم می‌شود، مالك داده‌ها بخش اول اشتراك و بخش دوم اشتراك را سرویس‌دهنده ابري نگهداري می‌کند. در اين حالت سرویس‌دهنده ابري تا زماني كه مالك داده بخش اشتراكي خود را اجرا ننمايد ، هيچ دسترسی‌ای به داده‌ها نخواهد داشت.

**3-6 معرفي CASB:**

اين سيستم توسط SafeBox تحت عنوان Access Security Broker طراحي شد و به‌صورت نرم‌افزار یا سخت‌افزار داخلی که به‌عنوان واسطه بین کاربران و ارائه‌دهندگان خدمات ابری عمل می‌کند اجراشده است. اين ايده امكان جستجو و اشتراك داده‌ها و حتي فایل‌ها با حفظ امنيت آن را فراهم می‌کند. يكي از امكانات اين سيستم جستجوي كلمات كليدي بر روي اطلاعات رمزگذاري شده را فراهم می‌کند.

**3-7 ساير تحقيقات :**

برخی از محققان از تکنیکی به نام "Bucketization" استفاده کردند که در آن ردیف‌ها داده به بیش از یک منبع متصل می‌شود. اين فنّاوری امكان استفاده از پايگاه داده به‌عنوان يك سرویس‌دهنده براي اجراي دستورات بر روی‌داده‌های رمزگذاري شده را فراهم می‌کند (Database as service) .

برخي از محققان نيز از فضاي ابري تركيبي براي تقسيم داده‌ها به دو بخش حساس و غير حساس استفاده نموده كه بخش غير حساس را در فضاي عمومي و بخش حساس را فضاي خصوصي كاربران قرار می‌دهند. يكي از اشكالات اين روش به اين صورت هست كه كاربران اکثراً داراي اطلاعات حساس می‌باشند و از سوي ديگر یکپارچه‌سازی بين دو بخش مجزا داراي سختی‌های خود هست.

گروهي ديگر از تکنیکی برای جلوگیری از افشاي اطلاعات توسط ارائه‌دهندگان خدمات ابري 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 اين روش داراي تأخیر براي ارتباط بين بخش‌های مختلف هست خصوصاً اگر دستور SQL پيچيده باشد! واسط اين كار يك پراكسي هست كه عمليات رمزگذاري و رمزگشايي را انجام می‌دهد.

با توجه به اينكه تقسيم و برون‌سپاری داده باعث ايجاد حساسیت‌ها و نگرانی‌هایی نسبت به افشاي اطلاعات می‌شود ، گروهي از محققين یک سیستم سخت‌افزاری / نرم‌افزاری را برای رفع مشکل نشت محرمانه بودن در پایگاه‌های داده برون‌سپاری شده معرفی کرده‌اند. روش كار به اين 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 اينجا با انواع رمزگذاري و سیستم‌های رايج رمزگذاري آشنا شديم و اين رمزگذاري را در پايگاه داده موردبررسی قراردادیم اما در يك بانك اطلاعاتي صرف رمزگذاري اطلاعات تنها می‌تواند در ذخيره اطلاعات مؤثر باشد و ما در يك پايگاه اطلاعاتي عملیات‌های متنوعي مانند دريافت ، انتخاب و محاسبه در اطلاعات داريم.

براي ايجاد يك پايگاه داده رمزگذاري شده بايد تمامي عملیات‌های قابل‌اجرا در بانك اطلاعاتي را بررسي نماييم و فرآيندي را پیش‌بینی كنيم تا از ابتداي ذخیره‌سازی ، تغييرات و دريافت اطلاعات به مشكل نخوريم. در اين فرآيند قصد داريم چنانچه قبلاً گفتيم از يك سيستم واسطه بنام QM (Query Manager) استفاده نماييم كه بين كاربران و بانك اطلاعات قرارگرفته و از یک‌سو به كاربران خدمات‌رسانی كند و از سوي ديگر با رمزگذاري اطلاعات از افشاي آن‌ها جلوگيري كنيم.

چنانچه بخواهيم تمامي اطلاعات اعم از حساس و غير حساس را رمزگذاري كنيم، ناچار هزينه بالاي ذخيره و خواندن اطلاعات را قبول كرديم اما اگر تنها اطلاعات حساس را رمزگذاري كنيم كمي از بار هزينه كم می‌شود. در اين رويكرد يا اطلاعات رمزگذاري شده را در يك ستون نگهداري می‌کنیم و يا در جدولي مجزا.

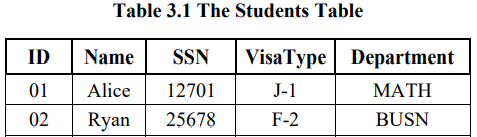
**4-1 استراتژي رمزگذاري :**

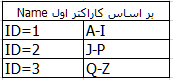
برای تأمین حریم خصوصی و سطح بالاتری از امنیت ، ما از AES-CBC برای رمزگذاری داده‌های حساس استفاده می‌نماییم فقط مدیر پرس‌وجو (QM) کلیدهای مخفی (SK) را نگهداری می‌کند. ما برای دستیابی به سطح بالاتری از امنیت و پردازش سریع‌تر رمزنگاری ، از الگوریتم متقارن به‌جای الگوریتم نامتقارن استفاده کردیم.

مدیر پرس‌وجو (QM) به‌عنوان یک سرور قابل‌اعتماد در فضای خصوصی یک سازمان یا شرکت قرار می‌گیرد و به‌عنوان یک واسطه بین کاربران و سرويس ابري 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 ابري بازنویسی می‌شود. مقادیر موجود در هر ستون يك 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 زير ، بر اساس حرف اول Name می‌تواند چندين Range تعريف نمود مثال A-I و J-P و براي هرکدام از اين محدوده‌ها يك شناسه اختصاص دهيم و مجموع اين محدوده‌ها می‌تواند يك PT را تشكيل دهد .





Alice : 12111

Ryan : 3312

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

براي آشنايي بهتر می‌خواهیم عملیات اصلي و جبر رابطه‌ای يك پايگاه داده رابطه‌ای را توسط QM توضيح دهيم.

**4-1-1** عملیات اصلي پايگاه داده **:**

* دستور ***Insert*** يك عمليات مستقيم 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 فضاي ابري 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 درخواستي در عمليات Select داراي اطلاعات حساس نیست ، پس رمزگذاری‌ای در ذخيره انجام‌نشده است و مستقیماً اطلاعات از پايگاه داده دريافت و نمايش داده می‌شود.

ب) تمامي ستون‌ها داراي اطلاعات حساس هست پس اين اطلاعات به‌صورت رمزگذاري شده ذخیره‌شده‌اند. در اين حالت QM براي هرکدام از ستون‌هايي كه در دستور وجود دارد ، موقعيت BV ها از BVM دريافت 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-2' " باشد ، QM هر BV را پیدا می‌کند که بیتی را نشان می‌دهد که مقادیر "QZ" را به گره نشان می‌دهد زيرا Mark زیرشاخه اين گره در PT هست و تمامي BV ها به‌دست‌آمده در يك آرايه ثبت می‌شود. سپس عملیات منطقی AND رابین دو لیست انجام می‌دهد و برای دستور را مطابق رکوردهای يافت شده اصلاح می‌نماید.

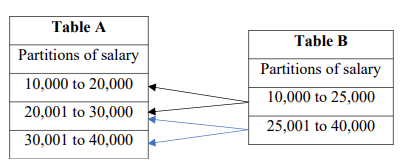
* دستور ***Update***نيز يكي از عملیات QM هست. براي این کار ابتدا QM توسط فرآيند توضيح داده‌شده Select ، ركورد را دريافت و رمزگشایی می‌نماید، سپس اطلاعات جديد را در ركورد يافت شده اصلاح می‌نماید و سپس آن را به سرويس ابري ارسال می‌نماید.
* دستور ***Delete***نيز مانند Update عمل می‌کند با اين تفاوت كه پس از يافتن ركورد موردنظر ديگر نياز به رمزگشایی نیست

در سه دستور Update و Delete و Insert تمامي تغييرات بر روي BVM نيز اعمال می‌گردد تا در مراجعات بعدي معتبر باشد.

* در دستور Alter نيز در صورت حذف يك ستون ، تمامي BV ها از BVM حذف می‌شود و سپس دستور حذف ستون ارسال می‌شود و چنانچه ما بخواهيم ستوني را اضافه نماييم و تعيين كنيم داراي اطلاعات حساس هست ، تمامي ركوردهاي موردنیاز BV در BVM ساخته و آماده می‌شود و اگر ستون داراي اطلاعات حساسي نباشد ، عمليات بدون پردازش خاصي انجام می‌شود.

**4-1-2** عملیات جبر رابطه‌ای **:**

* در دستور ***Join*** شايد بهترين و تنهاترين راه‌حل ، دريافت كل اطلاعات رمزگذاري شده جدول و انجام اين عمليات بر روي اطلاعات رمزگشایی‌شده باشد اما در حجم بالاي اطلاعات ، اين راهكار مناسب نیست براي همين بايد تا جايي كه امكان دارد، ستون‌های مورد پيوند Join بدون رمزگذاري در پايگاه ابري ثبت شود مگر آنكه ستون‌های پيوند داراي اطلاعات حساس باشد.

زماني كه پيوند ما با اطلاعات حساس روبرو باشد ، ما عملكردي مانند Select را خواهيم داشت يعني QM ابتدا به ازاي جداول پيوند فهرستی از PT را ساخته كه شامل محدوده مقادير مربوط به ستون‌های پيوند هست مثل تصوير روبرو و سپس ردیف‌هايي كه داراي شرط پيوند هست را تعيين می‌کند و سپس دستور را بازنويسي كرده و ارسال می‌نماید. درواقع این‌گونه ليست سازي منجر می‌شود كه ما بتوانيم اطلاعات كمتري جهت رمزگشايي دريافت و محاسبات كمتري انجام دهيم.

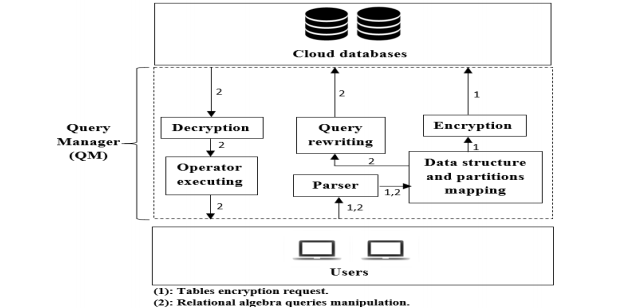
* در دستور ***Union*** هدف يكي كردن خروجي دو يا چند دستور هست كه در آن بايد مشخصات خروجی‌ها بايد يكي باشد. اين دستور مقادير تكراري را نيز از خروجي نهايي حذف می‌نماید. البته بايد جداول و خروجی‌هايي كه در اين عمليات شركت می‌کنند حتماً با يك كليد يكسان رمزگذاري شده باشد. QM ابتدا ردیف‌هاي شرکت‌کننده در اين عمليات را رمزگشايي می‌کند و آن‌ها را به يك 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 خروجي مشترک از دو یا چند خروجي و اين كار بدون رمزگشايي اطلاعات ، كار سختي هست. حال اگر حجم اطلاعات بالا باشد ، بار سنگيني براي به دست آوردن اطلاعات ايجاد می‌شود و بايد سعي كنيم اين كار را بر روي سرور انجام دهيم چون امكان پردازش از سمت كاربر نیست. در اين دستور هم مانند Union بايد خروجی‌ها داراي مشخصات يكسان باشد و چون دنبال اعضاي مشترك هستیم ، می‌توانیم از فرآيند دستور Join استفاده كنيم و خروجي حاصل‌شده را براي كاربر ارسال نماييم.
* عملگر ***Difference*** فرآیندی برعكس Intersection براي به دست آوردن خروجي خاي غیرمشترک هست. براي انجام اين عمليات از طريق Intersection ابتدا ردیف‌هاي مشترك را به دست آورده و آن را از خروجي اصلي خارج می‌کنیم تا خروجي به دست بيايد.
* عملگر ***Duplication Removal*** يا همان distinct فرآیندی براي حذف مقادير تكراري از خروجی‌ها هست. در فضاي ابري اين عمليات غیرممکن است زیرا ما از الگوریتم رمزگذاری غیرقطعی (AES-CBC) استفاده می‌کنیم. بنابراین ، ماقبل از ارسال خروجي به کاربر ، این عمل را در QM انجام می‌دهیم. در اين مورد نيز مانند Union می‌توانیم از يك LinkedHashSet باقابلیت حذف مقادير تكراري استفاده كنيم.
* عملگرهاي ***Aggregation and Sort*** شامل max, min, and count هست. ما در اينجا ستون‌ها با مقادير دامنه‌ای و غير دامنه‌ای داريم كه در ستون‌های غير دامنه‌ای می‌توانیم پردازش را در QM و در كاربري انجام دهيم. ما از محاسبات بر روي اطلاعات رمزگشایی‌شده به خاطر حجم محاسبات، اجتناب می‌کنیم. چنانچه قبلاً در دستور Select ما ابتدا اطلاعات مربوطه را همراه شرط‌هایی كه لحاظ شده بود را تو BV و BVM يافت می‌کردیم ، این بار هم مرحله اول به اين صورت هست. در مورد دستور count عمليات راحت هست چون ما ردیف‌ها را پيدا كرديم و کافی ست تعداد آن را به دست بياوريم اما در خصوص Sum مجبور به رمزگشايي هستیم و در دستور Avg بايد هم Sum و هم Count را انجام دهيم.
* عملگر ***Project*** يك عمليات بر اساس ستون‌ها هست كه QM تمام ردیف‌ها مطابق ستون‌های درخواستي انتخاب می‌کند و چون انتخاب ستون هست ما نيازي به PT نداريم اما به‌هرحال نياز به رمزگشايي هست و البته نيازي به حفظ اطلاعات تكراري نداريم.

**4-2 روش‌های قابل‌اجراي رمزگذاري** :

ما براي ثبت اطلاعات رمزگذاري شده ، دو روش را موردبررسی قرار می‌دهیم. يك روشي كه در آن اطلاعات رمزگذاري شده به‌صورت يك ستون در همان جدول نگهداري می‌شود و روش دوم، روشي كه اطلاعات رمزگذاري شده را در جدول مجزا نگهداري كنيم.

**4-2-1** استفاده Bit Vectors به‌صورت يك 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. اين مدل در تصوير زير نمايش داده‌شده است و اين مدل براي تمام عملگرهايي كه اينجا تحليل شدند اجرا می‌گردد. در اين مدل ما از رمزگذاري BV كه تا اينجا توضيح داده شد استفاده کرده‌ایم و در آن به‌جای آنكه محاسبات را مداوم توسط QM در سمت كاربر انجام دهيم ، آن را به CP منتقل می‌کنیم.



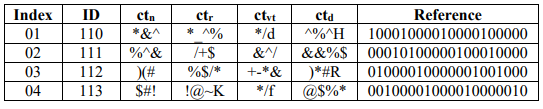
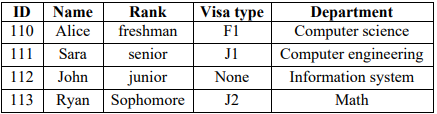
در اين مدل جدول رمزگذاري شده داراي يك ستون است كه حاوي اطلاعات BV هست و چون اطلاعات به‌صورت بيت هست نياز به نگراني بابت حجم آن نيست اما چون نهايت سايز ذخيره براي هر ويژگي 64Bit هست نياز به بیش از یک ویژگی داریم.

حتي اگر يك مهاجم بتواند از طريق استنتاج BV به اطلاعات دست پيدا كند، چون ما نام جداول و ستون‌ها را توسط الگوريتم رمزگذاری قطعی (AES) (که در آن متن‌های رمز برای هر متن ساده همیشه یکسان است) رمزگذاري می‌کنیم، اين احتمال ضعیف‌تر و مشكل مرتفع می‌شود.

به‌عنوان‌مثال ، ستون"Student-Rank" شامل محدوده‌ای از مقادیر مانند senior ، grad،junior و غیره است. یک مهاجم ممکن است بتواند مقادیر احتمالی را استنباط کند ، حتي وقتي آن‌ها به‌طور تصادفی با AES-CBC رمزگذاری شده‌اند.

اشتراک بیش از یک متن رمزگذاري با كليد يكسان باعث نگرانی نمی‌شود زیرا هكرها نمی‌تواند متن اصلي را به دست آورد مگر اینکه کلید رمزگذاري را به دست آورد و براي حل اين نگراني ، كليد به سرور ابری منتقل نمی‌شود.

براي مثال می‌خواهیم يك جدول را با كليد رمزگذاري كنيم. ابتدا QM نام جدول و ستون‌های حساس را توسط AES-DET رمزگذاري می‌کنیم به اين صورت كه جدول ديگري در فضاي ابري ساخته می‌شود و با آنچه تا اينجا درباره PT و الگوریتم‌های رمزگذاري گفته شد ، QM ردیف‌ها را به‌صورت BV رمزگذاري می‌کنید و جدول رمزگذاري شده دارای یک ستون Index است که در آن هر Index بخشی از BV هر ردیف است.

اين فرآيند براي جدول List در شكل زير توضيح داده‌شده است به اين صورت كه مشاهده می‌کنید ستون‌های Name - Rank - Visa Type و Department در جدول رمزگذاري شده توسط رمزگذاري AES-DET و يك كليد مشترك به‌صورت نام غير آشنا درآمده است.

زماني كه دستوري براي خواندن اطلاعات ارسال می‌شود QM ابتدا ستون‌های استفاده‌شده در دستور به ستون‌های رمزگذاري شده ترجمه می‌کند مانند Select Name From List Where [Visa Type]=? تبديل به Select Ctn From Cta where Ctvt=?

حال با پردازش بخش Where شروع به دست آوردن ردیف‌ها با توجه PT می‌نماید و ردیف‌هاي مربوطه يافت می‌شود و دستور بازنویسی شده را با توجه به ID ها می‌سازد. البته چنانچه قبلاً گفته‌شده PT شامل محدوده مقادير هست و Query بازنویسی شده نيز محدوده‌ای از اطلاعات را می‌آورد و نه ردیف‌هاي قطعي! مثلاً اگر شرط [Visa Type]='F1' باشد با توجه به حرف اول F و خاصیت Bit بودن ستون محدود اي به اين صورت در نظر گرفته می‌شود Reference&524288>0سپس با توجه به حرف دوم كه 1 هست يك شرط تركيبي ساخته می‌شود و درنهایت ممكن است دستوري مانند زير ساخته شود Reference&524288>0 And Reference&65536>0

درنهایت ردیف‌هاي به‌دست‌آمده توسط QM رمزگشایی می‌شود و مجدد شرط بر روي آن اعمال می‌گردد و اطلاعات به خروجي ارسال می‌شود.

عملیات‌هایی مانند Insert و Update و Delete نيز چنانچه قبلاً گفته شد اعمال می‌گردد با اين تفاوت كه این بار از BVM استفاده نمی‌کنیم و اين اطلاعات را در ستون Reference داريم.

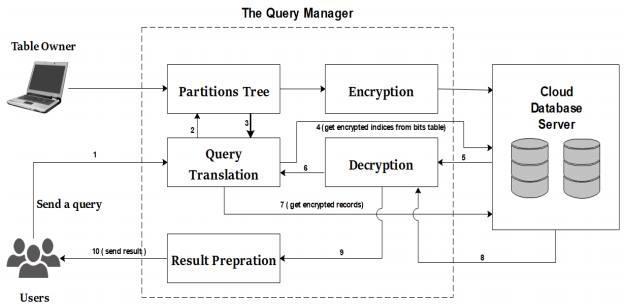
**4-2-2** استفاده Bit Vectors به‌صورت يك جدول مستقل **(BVSIT):**

در اين روش می‌خواهیم BV را در یک جدول مستقل در سرور ابری ذخیره می‌کنیم و بين 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 بيت داده‌ای توسط هكر از طريق استنتاج نشت نمی‌کند، ما Index موجود در جدول بیت‌ها را رمزگذاری می‌کنیم و سعي می‌کنیم حتي ازلحاظ قرارگيري موقعيت ركوردها نيز يكسان نباشد.

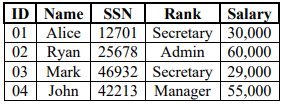
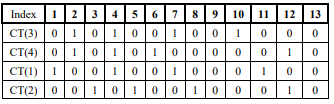
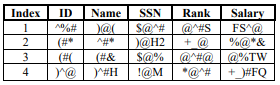
برای امنیت در BVSIT ، داده‌ها در تمام ستون‌های حساس با الگوریتم متقارن (AES-CBC) كه رمزگذاري تصادفی است رمزگذاری می‌شوند.

فرآيند كلي اين روش در شكل زير نمايش داده‌شده است.



در اين عمليات براي پردازش دستورات ، QM دو پرس‌وجو صادر می‌کند. پرس‌وجو اول ، شاخص‌های رمزگذاری شده تمام ردیف‌های داراي شرايط درخواستي را برای پرس‌وجو انتخاب‌شده از جدول بیت‌ها بازیابی می‌کند. پرس‌وجو دوم ، ردیف‌های رمزگذاری شده را مطابق جدول بیت‌ها انتخاب می‌کند و اطلاعات رمزگذاري شده را دريافت و رمزگشایی می‌کند و نتيجه به خروجي ارسال می‌شود. نحوه ساخته‌شدن جدول بیت‌ها و جدول رمزگذاري شده در تصوير زير نمايش داده‌شده است.

اطلاعات قرارگرفته در جدول بیت‌ها همان PT بر اساس محدوده‌هاي كاراكترها هست كه قبلاً توضيح داده‌شده است



< جدول رمزگذاري شده

< جدول بيت ها

عملیات‌هایی مانند Insert و Update و Delete از همان روش‌های گفته‌شده استفاده می‌کند و با اين تفاوت كه براي هرکدام از عملیات‌ها دو دستور ارسال می‌شود ، يكي براي جدول رمزگذاري شده و ديگري جدول بیت‌ها.

**فصل پنجم**

**مقدمه**

هر ايده يا استراتژي در بخش نرم‌افزار بايد در مرحله تست و آزمايش قرار بگيرد تا بتوان نسبت به اجرا و منابع موردنیاز، بررسي و نيازسنجي شود زيرا استفاده از ايده نامناسب و پرهزینه منجر به شكست سيستم می‌شود. با آنچه از رمزگذاري و پايگاه داده گفته‌ایم می‌توان مدل‌هاي معتبر و رايج را بررسي و مقايسه نمود.

در آزمايش بين مدل‌ها و تئوری‌های اجراي پايگاه داده رمزگذاري شده، پارامترهاي متعددي مدنظر قرار می‌گیرد مانند : حجم اطلاعات، زمان اجرا، هزینه‌های سخت‌افزاري مانند حافظه و امنيت مدل‌ها.

مدل‌هاي پايگاه داده رمزگذاري شده با توجه به ايده و استراتژي كه در آن قرار دارد در عملیات‌های مختلف داراي پاسخ و ارزيابي متفاوتي هست مانند آنكه مدلي در هنگام ذخيره اطلاعات به علت طراحي آن باعث هزينه بالا می‌شود و يا مدلي در ذخیره‌سازی اطلاعات داراي نتايج بهتر ولي در بازخواني اطلاعات دچار كندي يا هزينه بالاي اجرا می‌شود. البته عملیات‌های پايگاه داده تنها ذخيره و خواندن اطلاعات نیست و عملیات‌های محاسباتي نيز موردبررسی قرار می‌گیرد پس آن چيزي كه به‌عنوان مدل مناسب انتخاب می‌گردد، مدلي هست كه داراي ميانگين بهتري نسبت به ساير مدل‌ها باشد و درنهایت بايد مدل انتخابي را در اجراي فضاي ابري موردبررسی قرارداد.

**5-1 آزمايش و ارزيابي :**

تا اينجا ، ما به‌طور مفصل مدل‌های پیشنهادی را توضیح دادیم و برای هر مدل ، الگوریتم‌های توسعه‌یافته را برای اجرای عملگرهای مختلف جبر رابطه‌ای ارائه دادیم. همان‌طور که قبلاً گفته شد ، این پایان‌نامه باهدف پیاده‌سازی ، 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 در QM) ارائه داديم.

حال براي مقايسه مدل‌ها و سیستم‌های ارائه‌شده نياز به آزمایش آن‌ها داريم.

برای انجام همه آزمایش‌ها برای همه سیستم‌ها BVM ، BVSAC ، BVSIT ، OBT ، CBF ، CryptDB از يك سيستم با 6 گیگابایت رم ، 1 ترایایت HDD و پردازنده Core i5 با 2.8 گیگاهرتز استفاده کردیم. برای اجرای توابع QM در مدل‌های پیشنهادی ، از Java برای استفاده کردیم. سرور MySQL بر روی دستگاه کاربر استفاده شد و ما از Java Database Connectivity (JDBC) به‌عنوان اتصال‌دهنده به MySQL استفاده کردیم و تمام آزمایش‌ها بر روی دستگاه محلی انجام‌شده است تا تأخیر ارتباط از گزارش‌ها حذف شود.

در تمامي مدل‌های خود ، از نسخه تصادفی AES-CBC برای رمزگذاری داده‌های حساس استفاده کردیم و براي رمزگذاري و رمزگشایی از كتابخانه "javax.crypto" استفاده نموده‌ایم.

ما OBT و CBF را اجرا کردیم زیرا پیاده‌سازی آن‌ها برخلاف CryptDB که برای استفاده عمومی در GitHub در دسترس است به‌صورت آنلاین در دسترس نیست و برای ذخیرهBV 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 ساير مدل‌ها آن‌ها را در داده ابري بارگذاری می‌کنیم.

براي شروع آزمايش ما به‌طور تصادفی 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 فهرست‌ها ، ردیف‌هايي را بسازد. اندازه جداول (تعداد ركوردها) 10هزار – 20هزار – 50هزار و 10هزار رديف بود.

ما در کل 24 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.

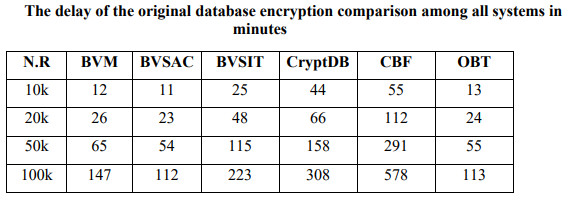
**5-2 ارزيابي** :

ارزیابی شامل موارد زیر هست :

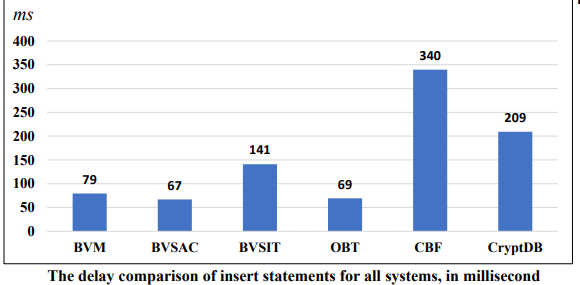
* 1. آزمایش و ارزیابی هزینه اجرای عملیات پایه پایگاه داده (Insert – Update – Deleted - Insert)
  2. آزمایش و ارزیابی هزینه اجراي عملیات محاسباتي (Sum – Min – Max - Avg)
  3. آزمایش و ارزیابی هزینه‌های اجراي عبارات جبر رابطه‌ای (Join – Union - Intersection)
  4. شناسایی نیازهای فضایی برای هر سیستم

در مدل‌های پیشنهادی ، مرحله رمزگذاری پایگاه داده اصلی شامل تجزیه رکوردها ، تولید شاخص‌ها ، ساخت BV ، رمزگذاری داده‌های حساس و قراردادن داده‌های رمزگذاری شده در جدول رمزگذاری شده در سرور ابری است.

در جدول زير ما زمان صرف شده توسط هر سیستم برای رمزگذاری هر جدول را مقایسه کرده‌ایم و همان‌طور که دیده می‌شود ، مدلBVSAC کمترین تأخیر در رمزگذاری و درج را در بین همه سیستم‌ها تجربه کرده است زیرا QM نیازی به ذخیره و مدیریت BV ها به‌صورت محلی در QM مانند BVM ندارد.



پس از BVSAC مدل OBT است که هر رديف را به‌صورت یک بلوک رمزگذاری می‌کند و آن را در ستوني قرار می‌دهد و درروی دیگر، مدل CBF بالاترین تأخیر فرایند رمزگذاری را دارد زيرا به ازاي هر ستون بايد جداگانه درج انجام شود و رکوردها را به جداول مختلف در سرورهای مختلف ابری قرار می‌دهد. مقايسه مدل‌ها براي عمليات Insert در جدول زير نمايش داده‌شده است

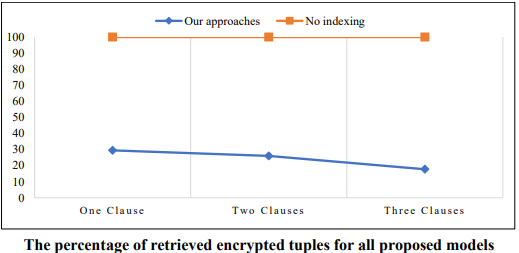


مدل كند بعدي مدل CryptDB هست كه به دلیل محاسبات سنگین رمزگذاري Onion هست.

**5-2-1** آزمايش شماره يك :

ازآنجاکه پراستفاده‌ترین عمليات ، عمل انتخاب (Select) هست ، ما بيشترين تمركز خود را بر روي اين عمليات می‌گذاریم.

در این آزمایش ، ما درصد متوسط ​​ردیف‌هاي 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.

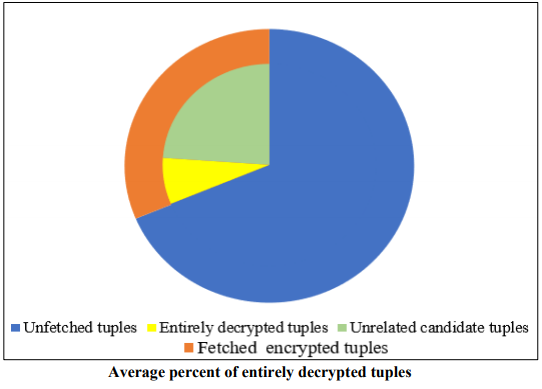


چنانچه در نمودار می‌بینید ، بدون استفاده از هیچ‌یک از نمونه‌های اولیه پیشنهادی برای مدیریت پایگاه داده رمزگذاری شده (یعنی بدون Index) باید کل جدول رمزگذاری شده برون‌سپاری شده را بازیابی کنیم.

**5-2-2** آزمايش شماره دو :

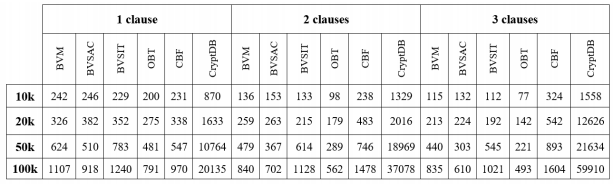
چنانچه تا اينجا گفته شد هدف ما این است که برای جلوگیری از محاسبات سربار ، که باعث تسریع در زمان اجرای درخواست می‌شود ، محاسبات رمزنگاری را در QM کاهش دهیم.

در تصوير زير می‌بینیم که از کل مجموعه رمزگذاری شده واکشی شده ، میانگین ردیف‌های کاملاً رمزگشایی‌شده کمتر از 24٪ 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.



**5-2-3** آزمايش 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 اينجا CryptDB ، OBT و CBF آزمایش و ارزیابی کردیم.

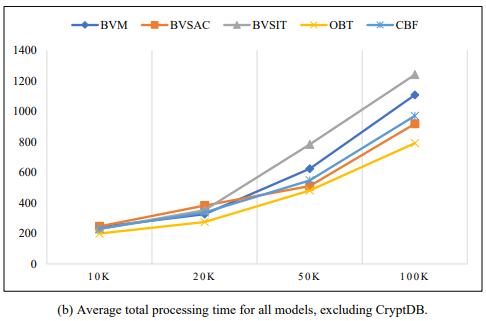
جدول زير کل زمان اجرا را برای همه سیستم‌ها هنگام اجرای عبارات *Select*برای بازیابی مقادیر با خاصیت ستون حساس ارائه می‌دهد و مدت‌زمانی که اندازه‌گیری کردیم زمان تجزیه پرس‌وجو بود تا نتیجه پرس‌وجو نهایی در میلی‌ثانیه (*ms*) است.

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 20هزار و 10هزار رديف 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 (10هزار 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 جايگاه دوم برای پایگاه داده با بیش از 50هزار 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 داراي تايم خيلي بالاتري نسبت به ساير مدل‌ها هست، ما آن را حذف و ميانگين ساير مدل‌ها در هر سه حالت را درآورده‌ایم.

نمودار روبرو حاصل مقايسه مدل‌ها هست.

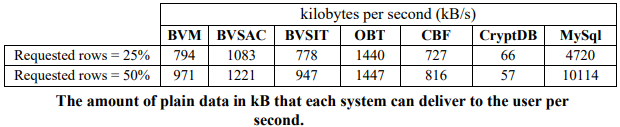
با توجه به نمودار و جدول بالا مشاهده می‌شود كه مدل OBT داراي سرعت بهتري نسبت به ساير مدل‌ها هست زیرا مقدار داده‌های رمزگشایی کمتر از پيشنهاد ما بود (همه مقادیر به‌عنوان یک بلوک ذخیره می‌شوند که منجر به رمزگشایی کمتر برای هر سطر می‌شود) البته اين براي زماني است كه دستور Select با تعداد ستون بيشتر انجام شود زيرا در دريافت تک‌ستونی ، ساير مدل‌ها از سرعت بهتري برخوردار هستند.

**5-2-4** توان عملياتي :

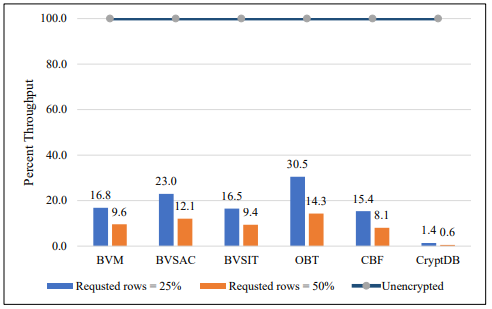
توان عملیاتی به‌عنوان مقدار داده منتقل‌شده در یک‌زمان معین تعریف می‌شود. با اندازه‌گیری توان عملیاتی ، می‌توان فهمید که کدام سیستم با افزایش داده‌های درخواستی ، پاسخ بیشتری به دستورات کاربر می‌دهد ، زیرا کاربر نهایی کسی است که تحت تأثیر کاهش سرعت سیستم قرار خواهد گرفت.

برای اندازه‌گیری توان عملیاتی ، مجموعه‌ای از پرس‌وجوها را برای بازیابی٪25 و سپس ٪50 رکوردها از جدول دارای 100000 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

جدول زير اين محاسبات را براي تمامي مدل‌ها نمايش می‌دهد

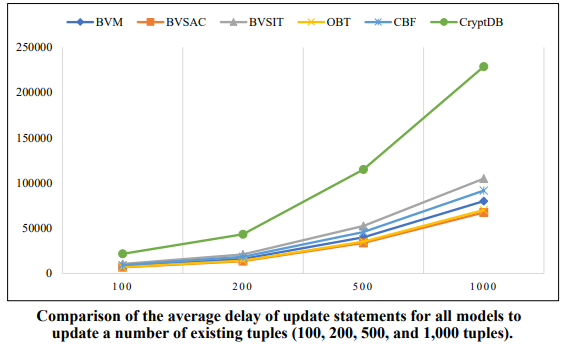


چنانچه مشاهده می‌شود مدل BVSAC توان عملیاتی بالاتری را در بین سیستم‌های پیشنهادی دارا هست که آن را بهترین گزینه برای کاربران نهایی می‌کند که به دنبال یک سیستم پاسخگویی سریع‌تر هستند. همچنين مدل OBT بالاترین توان عملیاتی را دارد ، به این دلیل که برای رمزگذاری داده‌ها به مقدار كمتري از بايت در میان همه سیستم‌ها نیاز دارد. براي درك بهتر مقادير را به‌صورت نمودار درصد درآورديم.



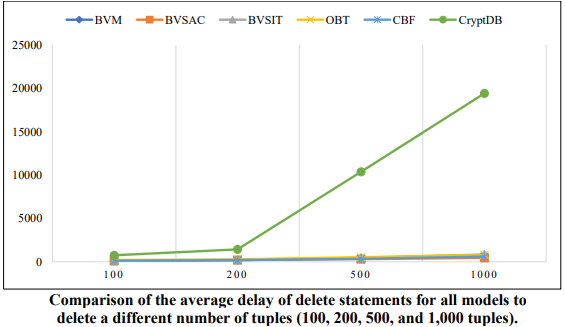
**5-2-5** آزمايش شماره چهار (آزمايش دستورات Update و Delete) :

ما اين آزمايش را با مدل‌های مختلف انجام و نمودار زير متوسط ​​زمان هزینه شده توسط هر سیستم برای اجرای عبارات *Update را نشان می‌دهد*. در این آزمایش ، عبارات *Update*را فقط با یک فيلد اجرا کردیم. همان‌طور که در شکل دیده می‌شود ، هزینه زمان به‌روزرسانی در همه سیستم‌ها زیاد است و نتیجه به‌روزرسانی یک فيلد 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 100رديف ، در حالت دوم 200 رديف و غیره صادر می‌کند.



در مدل‌های پیشنهادی ، BVSIT كندترين 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 گفتBVSAC کارآمدترین مدل برای اجراي *Update*است.

آزمايش را با انجام عمليات Delete انجام داديم و نمودار زير زمانی است که هر مدل برای حذف تعداد متفاوت (100 ، 200 ، 500 و 1000 رديف) صرف می‌کند ، درحالی‌که شرط حذف فقط یک‌بند دارد.

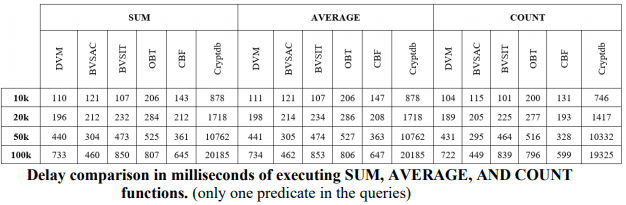


فرآیند حذف ، ردیف‌های موردنیاز را انتخاب کرده و سپس آن‌ها را حذف می‌کند. حذف در همه مدل‌های پیشنهادی کارآمد است زیرا حذف پس از اجرای عملیات *Select*برای بازیابی ردیف‌هاي موردنیاز انجام می‌شود. به‌جای ارسال یک پرس‌وجو برای حذف هر رکورد ، ما Index رکورد را به دست می‌آوریم و سپس یک دستور صادر می‌کنیم تا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

**5-2-6** آزمايش شماره پنج (دستورات محاسباتي) :

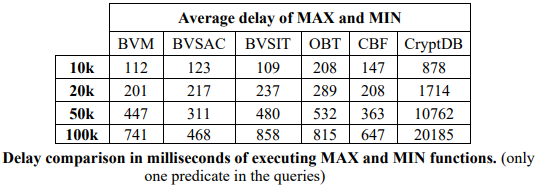
ما اين آزمايش را با دو بخش تقسيم و در بخش اول متوسط ​​زمان هزینه شده توسط هر سیستم برای اجرای عملیات *Count , Average , Sum در* مدل‌های مختلف را به دست آورده‌ایم. در بخش دوم ساير دستورات مانند Max و Min مورد آزمايش قرارگرفته‌اند.

تمامي آزمایش‌ها براي محدوده مشخصي از اطلاعات هست و چيزي حدود 6% از كل اطلاعات.



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 كه مدل‌های آن بر ضرب مدو لار برای تولید مجموع رمزگذاری شده استفاده می‌کند، که نیاز به محاسبه سنگین‌تر دارد.

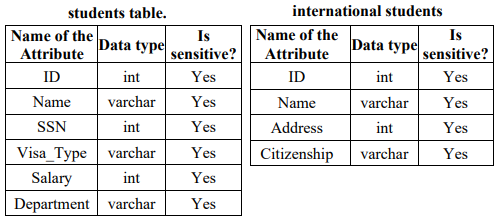
جدول زير آزمايش را در هر سیستم برای اجرای عملیات  *Min , Max* نشان می‌دهد.



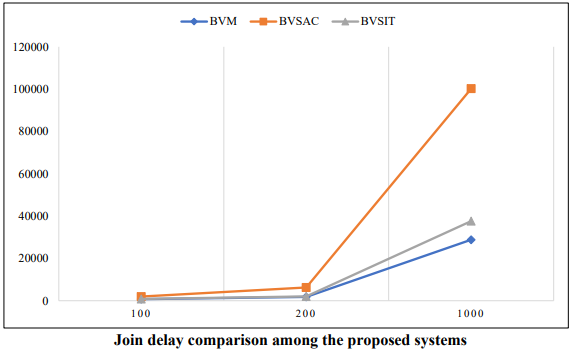
با بررسي اين دو جدول در آزمایش‌ها عبارات محاسباتي ، می‌توانيم مدل كه داراي تأخیر كمتر هست را به دست آوريم كه آن را می‌توان براي پايگاه داده كوچك در نظر گرفت ، اما برای پایگاه‌های داده بزرگ‌تر ، مدلBVSAC برنده است ، درحالی‌که BVSIT محاسبات سنگین‌تری را تجربه می‌کند ، همان‌طور که گفته شد در آزمایش قبلی  CryptDB به دلیل عملیات رمزگشایی فشرده ، کندترین مدل هست.

**5-2-7** آزمايش شماره شش (دستورات Join , Union) :

در اين آزمايش قصد داريم هزينه زماني دستورات *Join , Union* را مقايسه كنيم و چنانچه قبلاً گفتيم در مدل CryptDB و CBF اين دو دستور پشتيباني نمی‌شود، لذا از آزمایش‌ها حذف می‌شود. در اين آزمايش از دو جدول كه توسط ستون رمزگذاري شده ID باهم Join شده‌اند و با آزمایش‌ها 500 و 1000 و 10000 رديف استفاده کرده‌ایم كه به ترتيب داراي 100 و 200 و 1000 ركورد در شرط Join استفاده‌شده هست.

مشخصات دو جدول در تصوير زير آماده است

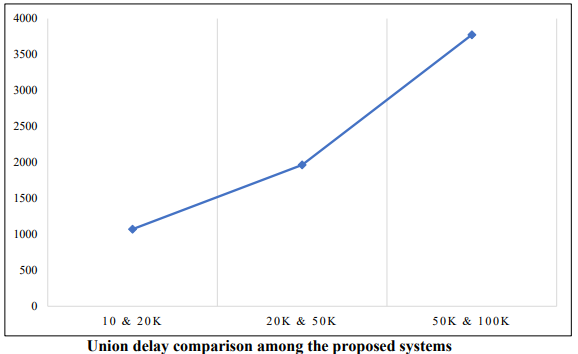
نمودار مقايسه مدل‌ها در عمليات Join به شكل زير آماده است.



اين نمودار نشان می‌دهد که تأخیر BVSAC بالاترین و پس‌ازآن BVSIT است و BVM دارای تأخیر كمتري 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 (در QM) ، تاپلهای رمزگشایی‌شده به داخل LinkedHashSet برده شدند تا تكراري 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 كه اين تأخیر براي امنيت بهتر هست.

نمودار هزينه زمان در تمامي مدل‌ها در دستور Union به شكل زير آماده است و از ميانگين كليه مدل‌ها استفاده‌شده است زيرا در آزمایش‌ها متوجه شديم كه زمان مدل‌های خيلي به هم نزديك هست . سعي شده است دستور Union به صورتی انتخاب شود كه بدون تغيير توسط QM به سرور ارسال شود. تأخیری که اندازه‌گیری کردیم از زمانی است که QM پرس‌وجو را قطع می‌کند تا نتیجه نهایی پرس‌وجو تشکیل شود. برای حذف موارد تکراری ، ما ردیف‌هاي رمزگشایی‌شده را از هر دو جدول به یک LinkedHashSet هدایت کردیم.

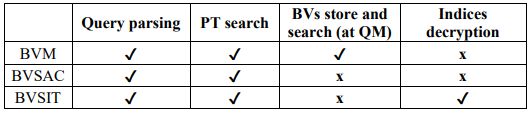


چنانچه مشاهده می‌شود با افزايش ركوردها ، زمان بيشتري صرف می‌شود.

**5-2-8** آزمايش شماره هفت (بازنويسي دستورات توسط QM) :

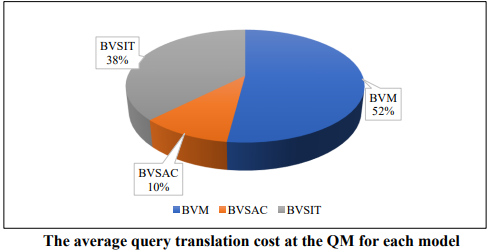
چنانچه قبلاً گفتيم در رمزگذاري ما از QM استفاده می‌کنیم تا دستورات را بازنويسي كند تا به‌سرعت بهتر و تأخیر كمتري در پردازش و نمايش اطلاعات برسيم. اين آزمايش میانگین زمان صرف شده توسط هر مدل برای بازنویسی یک پرس‌وجو را نشان می‌دهد.

بازنويسي دستورات شامل چند مرحله هست كه در جدول زير براي مدل‌های انتخابي بررسی‌شده است



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

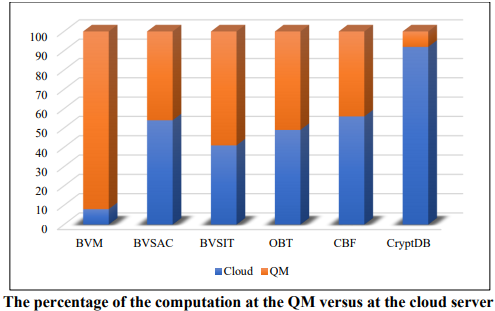
ما روی عبارات *Select*متمرکز شدیم زیرا این‌ها بخشی از دستورات به‌روزرسانیو *حذف هستند*، بنابراین بررسی  عبارت *Select*برای اندازه‌گیری هزینه بازنويسي دستورات در هر مدل کافی است همچنين عملیات رمزگذاری را از مقادير آزمايش حذف کردیم.



طبق این نمودار ، BVM بیشترین تأخیر را دارد (حدود 52٪) زيرا بالاترین درصد تولید و پردازش BV را ارائه می‌دهد ، پس‌ازآن BVSIT (حدود 38٪) ، و BVSAC تنها 10٪ از هزینه کل را گرفته است.

**5-3** بررسي محاسبات Cloud و QM :

در اين بررسي سعي شده است حجم محاسبات در سرور ابري و QM محاسبه و مقايسه شود . نمودار زير اين مقدار را به‌صورت میانگین درصد زمان محاسبه نشان می‌دهد.



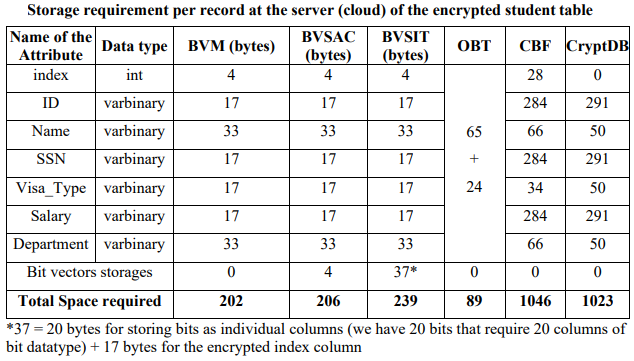
چنانچه مشاهده می‌شود که CryptDB بیش از 90٪ تأخیر محاسبات پردازش پرس‌وجو را به سرور منتقل کند ، و کمتر از 10٪ تأخیر محاسبه را به QM واگذار می‌کند (QM معادل پروکسی در CryptDB و CBF است) .

همچنين می‌توانیم ببینیم که BVM بیشترین هزینه محاسبات را در QM دارد زیرا علاوه بر عملیات رمزنگاری ، BV ها به‌صورت محلی در QM پردازش می‌شوند. مدل BVSIT به دلیل رمزگشایی شاخص‌های رمزگذاری شده BV پس از واکشی آن‌ها از سرور حدود 60٪ محاسبات در QM را دارد. مدل OBT حدود 53٪ محاسبه در QM را دارد زیرا QM به‌طور کامل رمزگشایی می‌کند ، حتی برای اگر دستور خواندن تمامي ستون‌ها نباشد.

از طرف دیگر ، BVSAC و CBF درصد نزديك به هم دارند و حدود 50٪ 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.

**5-4** بررسي فضاي 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 4 بایت نیاز دارد ، درحالی‌که بقیه ستون‌ها (ستون‌هایی که رمزگذاری می‌شوند) به 33 بایت نیاز دارند (32 بایت ازآنجاکه کلید مخفی 32 بایت است و 1 بایت برای ضبط طول نوع داده) علاوه بر این ، ما فضای موردنیاز برای ذخیره BV ها را در سرور برای هر مدل ارائه می‌دهیم.



همان‌طور که در دیده می‌شود ، OBT کمترین نیاز ذخیره‌سازی را در سمت سرور دارد زيرا كل ديتا به يك بلوك تبديل 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 1046 بایت نیاز دارد که بالاترین میزان فضای موردنیاز در بین سیستم‌ها است. CryptDB دومین بالاترین میزان نیاز به فضا را کسب می‌کند ، حدود 1023 بایت / رکورد.

**فصل ششم**

**مقدمه**

بحث در خصوص بانک‌های اطلاعاتي و دغدغه‌های افشاي اطلاعات همواره پژوهشگران و صاحبان داده را به دنبال بررسي و انتخاب راهكارهاي امن و مناسب كشانده و در اين مسير ایده‌های متفاوتي مطرح می‌شود چنانچه تا اينجا مدل‌ها و استراتژی‌های مختلف عنوان گرديد و موردبررسی قرار گرفت.

مدل‌ها و ایده‌هایي كه در اين پایان‌نامه مطرح گرديد را می‌توان به‌عنوان راهكارهاي اوليه دانست زيرا داراي نقص و کمبودهایی هست، علی‌الخصوص كه بانک‌های اطلاعاتي مانند SQL با فراهم آوردن گزينه رمزگذاري و رمزگشایی به‌عنوان دو ابزار دروني، اين دغدغه را كاهش داده است.

در اين تحقيق، تمامي تمركز بر يك سيستم به‌عنوان واسط بنام QM بوده تا بتواند با ترجمه دستورات و درخواست‌ها به بانك اطلاعاتي و نگهداري اطلاعات به‌صورت رمزگذاري شده، از افشاي آن توسط كاربران غیرمجاز و يا صاحبان سرورها جلوگيري نمايد اما آنچه به‌عنوان راهكار در اين QM در نظر گرفته‌شده داراي اشكالات و نقص‌های فراواني هست و در بعضي بخش‌ها حتي به بن‌بست خواهيم رسيد.

البته ايده و استراتژي كاهش حجم اطلاعات در فراخواني و انتخاب اطلاعات را می‌توان ايده مناسبي دانست و در راهكارهاي بعدي استفاده نمود. در ادامه راهكارها و ايده بهتري را جهت تحقيقات آتي اعلام می‌داریم.

**6-1 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 زيرا 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 ردیف‌هاي رمزگذاری شده بازیابی شده از ابر کمک می‌کند. البته داشتن تعداد پارتيشن متعادل خود نيز چالشي هست كه طراح و مالك داده‌ها با آن روبرو هست.

**6-1-1 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 35 درصد کاهش دهند و همان‌طور كه ديده 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. درBVSAC نيز ما BV ها را به‌عنوان یک ستون اضافی در همان جدول رمزگذاری شده در ابر ذخیره كرديم.

برای سیستم‌های تك كاربر 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 مشكلاتي 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 كه از روش BVSIT استفاده می‌نماید. همچنينBVSAC کمترین زمان اجرا ، دومین کمترین میزان فضای موردنیاز در ابر و کمترین سيستم 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 سرويس ابر تبانی نکنند ، امنیت بالابود. به‌هرحال تضمین‌شده نیست و رتبه سطح امنیت پاييني دارد .

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

مدل OBT در صورت استفاده از الگوریتم رمزگذاری تصادفی ، ایمن بود وگرنه سطح امنیت کاهش پيدا 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 سرويس ابري 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 .

**6-1-2 پيشنهادها** :

آنچه در اين پایان‌نامه بررسي و موردبحث قرار گرفت راهكارها و مدل‌هاي ايجاد و ارتباط با يك بانك اطلاعاتي رمزگذاري شده، بوده و به‌عنوان راهكار كلي براي تمامي انواع پايگاه اطلاعاتي رابطه‌ای قابل انجام هست. امروزه شرکت‌های ارائه‌دهنده پایگاه‌های داده رابطه‌ای مانند SQL با ايجاد دستورات و ابزارهاي آسان و قابل‌فهم‌تر، قدم بزرگي در ايجاد بانک‌های اطلاعاتي رمزگذاري شده برداشته‌اند به صورتی كه ديگر نياز به يك نرم‌افزار واسط نباشد.

آن چيزي كه در اين پایان‌نامه ایدئال بود، ايده كاهش تعداد رديف پردازش رمزگذاري هست زيرا حتي با داشتن ابزارهايي مانند آنچه بيان شد، باز بايد تمامي ردیف‌ها مورد پردازش رمزگشایی قرار گيرد ، پس ايده كاهش تعداد رديف جهت پردازش نهايي به‌عنوان ايده مناسبي می‌تواند در كنار اين ابزارها و امكانات منجر به ايجاد يك بانك و سرويس اطلاعاتي امن گردد.

اين نكته را بايد در نظر بگيريم كه عملیات‌های پايگاه داده خيلي گسترده‌تر و متنوع‌تر از آن چيزي است كه در اين پایان‌نامه موردبررسی قرار گرفت و در مدل‌هاي مختلف اجرا گرديد مثلاً عملیات‌هایی مانند Like كه قابليت جستجو در محتواي فیلدهای كاراكتري هست در اين پایان‌نامه قرار نگرفته و عملاً براي اجراي آن نياز به رمزگشايي در تمامي اطلاعات هست كه نا گریز منجر به هزينه بالاي اجرا می‌گردد.

همچنين ايده نرم‌افزار واسط به‌عنوان يك ايده مناسب براي كار با بانك اطلاعاتي رمزگذاري شده را می‌توان به‌صورت يك حافظه موقت رمزگشایی‌شده طراحي و اجرا نمود چنانچه اين نرم‌افزار هر زمان كه اطلاعاتي از جدولي درخواست شود آن را به‌طور كامل رمزگشایی و در يك حافظه موقت قرار دهد و تمامي تغييرات هم‌زمان بر حافظه موقت و بانك اطلاعاتي لحاظ گردد. در اين حالت بار هزينه تنها یک‌بار براي مدت اجراي حافظه موقت اتفاق ميفتد و در تمامي ارجاعات، اين نرم‌افزار با اطلاعات رمزگشایی‌شده كار خواهد كرد و دسترسي به آن نيز پس از گذراندن مجوزهای لازم هست.

**6-2** 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 رديف از داده را به‌صورت بیتی رمزگذاری می‌کنیم .

از بیت‌ها برای بازیابی ردیف‌هاي انتخابي 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**

1. R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan, “CryptDB: Processing queries on an encrypted database.” Commun. ACM, vol. 55, no. 9, pp. 103–111, 2012.
2. Y. D. Jang and J. H. Kim, “A comparison of the query execution algorithms in secure database system,” Int. J. Electr. Comput. Eng., vol. 6, no. 1, pp. 337–343, 2016.
3. W. Wang, Y. Hu, L. Chen, X. Huang, and B. Sunar, “Exploring the feasibility of fully homomorphic encryption,” IEEE Trans. Comput., vol. 64, no. 3, pp. 698–706, 2013.
4. A. Alsirhani, P. Bodorik, and S. Sampalli, “Improving database security in cloud computing by fragmentation of data,” in 2017 International Conference on Computer and Applications (ICCA), 2017, pp. 43–49
5. 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 2002 ACM SIGMOD international conference on Management of data, 2002, pp. 216–227.
6. 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 2002 ACM SIGMOD international conference on Management of data, 2002, pp. 216–227.
7. A. Boicea, F. Radulescu, C.-O. Truica, and C. Costea, “Database encryption using asymmetric keys: a case study,” in 2017 21st International Conference on Control Systems and Computer Science (CSCS), 2017, pp. 317–323.
8. O. M. Ben Omran and B. Panda, “A new technique to partition and manage data security in cloud databases,” in The 9th International Conference for Internet Technology and Secured Transactions (ICITST-2014), 2014, pp. 191–196.
9. S. Cui, M. R. Asghar, S. D. Galbraith, and G. Russello, “P-McDb: Privacy-preserving search using multi-cloud encrypted databases,” in 2017 IEEE 10th International Conference on Cloud Computing (CLOUD), 2017, pp. 334–341
10. O. M. Omran, “Data Partitioning Methods to Process Queries on Encrypted Databases on the Cloud,” 2016.
11. V. H. Hacigumus, B. R. Iyer, and S. Mehrotra, “Query optimization in encrypted database systems.” Google Patents, Mar-2010.
12. V. H. Hacigumus, B. R. Iyer, and S. Mehrotra, “Query optimization in encrypted database systems.” Google Patents, Mar-2010.
13. “Package javax.crypto.” [Online]. Available: https://docs.oracle.com/javase/7/docs/api/javax/crypto/packagesummary.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.