ИССЛЕДОВАНИЕ ВЛИЯНИЯ ПРОЗРАЧНОГО ШИФРОВАНИЯ ДАННЫХ НА ПРОИЗВОДИТЕЛЬНОСТЬ СУБД ORACLE

Борисов Алексей Николаевич

студент, Самарский Национальный Исследовательский Университет им. академика С.П. Королева, РФ, г. Самара

Додонов Михаил Витальевич

канд. пед. наук, доцент, Самарский Национальный Исследовательский Университет им. академика С.П. Королева, РФ, г. Самара

Введение.

Защита информации от несанкционированного доступа – один из основных аспектов информационной безопасности. На уровне СУБД это аспект реализуется с помощью стандартных средств: привилегий, ролей, триггеров, хранимых процедур, специализированных пакетов и т.д. Однако данные средства не защищают данные при попытке доступа к ним по иным каналам, к примеру, простого копирования файлов БД. В этом случае приходится прибегать к дополнительным мерам, одной из которых является шифрование.

СУБД Oracle Enterprise Edition 12.2 предоставляет возможность использования прозрачного шифрования данных, при котором все задачи шифрования/дешифрования перекладываются на внутренние механизмы СУБД. Oracle декларирует относительно небольшое снижение производительности при включении прозрачного шифрования, однако не конкретизирует при этом влияние различных конфигураций. В открытых источниках также нет точной информации по данному вопросу. В специализированной литературе так же этот вопрос широко не освещен [2, с. 51].

Описание предмета исследования.

При использовании прозрачного шифрования все работы по шифрованию/ дешифрованию выполняются самой СУБД автоматически. Поддерживаются алгоритмы AES(с длиной ключа 128,192 и 256 бит) и 3DES, алгоритм по умолчанию – AES192.

Доступно 2 типа шифрования: шифрование столбцов таблицы и шифрование табличных пространств. В одной таблице могут быть как шифрованные, так и нешифрованные столбцы, при этом, если шифрованные столбцы не участвуют в запросе, никаких дополнительных операций не выполняется. Столбец может быть шифрован с указанием опции SALT (включена по умолчанию), при этом одинаковые значения в столбце будут давать разный шифротекст. Это увеличивает безопасность, так как делает невозможными статистические атаки, но накладывает ограничение – данный столбец нельзя использовать в индексах. При использовании шифрования табличного пространства никаких ограничений не накладывается, однако любая операция чтения/записи требует определенных дополнительных действий по шифрованию.

Описание методики исследования

Для тестирования влияния шифрования на производительность были созданы таблицы с одинаковым набором полей:

id number,

key varchar2(6),

value varchar2(1024)

и следующими конфигурациями:

Таблица 1.

Конфигурации таблиц

№ конфигурации	Описание конфигурации
1	Нет шифрования
2	Шифруется только столбец value
3	Шифруются столбцы value и key
4	Столбцы value и key шифруются с опцией SALT
5	Value и key шифруются алгоритмом AES256
6	Value и key шифруются алгоритмом 3DES
7	Таблица располагается в зашифрованном табличном пространстве
8	Таблица располагается в зашифрованном алгоритмом AES256 табличном пространстве
9	Таблица располагается в зашифрованном алгоритмом 3DES табличном пространстве

По столбцу key строится индекс(за исключением конфигурации 4), расположенный в одном табличном пространстве с таблицей.

Каждая таблица заполнена 1000000 строк. Key заполняется строковым представлением случайного числа из диапазона [100000, 999999], с целью обеспечения наличия повторений,value заполняется случайной строкой длиной 1024 символа, id заполняется номером строки.

Для тестирования используются следующие запросы:

1. *select count(id) from [table] where value like ‘%DES%’;*

2. *select count(id) from [table] where key like ‘56%’;*

Первый запрос, в силу большого объема охватываемых данных, позволит оценить влияние шифрования на операции ввода/вывода. Второй запрос позволит лучше понять влияние шифрования на индексы. Перед исполнением каждого запроса производилась очистка буфера, с целью предотвращения повторного использования расшифрованных данных. Каждый запрос запускался 10 раз.

Результаты замеров производительности

Таблица 2.

Статистики производительности запроса №1

№ конф.	Среднее время выполнения	Среднее время ЦП	Среднее время ожидания ввода/вывода	Логических чтений	Физических чтений
1	6872579	6727263	206846	330426	166782

2	14512073	14277454	355220	333449	166778
3	13160313	12981451	307990	333433	166762
4	11777079	11615125	252479	333513	166842
5	13417814	13189907	324985	333581	166910
6	68368095	67382595	350827	333437	166765
7	9617632	9511455	306336	333580	166907
8	9237339	9108478	279266	333579	166906
9	75009228	73353389	412685	333451	166778

Медленнее всего исполняются запросы к таблицам, зашифрованным алгоритмом 3DES. Затем идут таблицы, включающие в себя таблицы, включающие шифрованные столбцы.

Запросы к таблицам, находящиеся в табличных пространствах, зашифрованных алгоритмами семейства AES, работают быстрее предыдущих двух вариантов, однако и в этом случае время, затраченное на запрос, в 1,4 раза превышает время запроса к таблице без шифрования. Превосходство над алгоритмом 3DES можно объяснить поддержкой инструкций AES-NI, позволяющих аппаратно производить шифрование. Хотя число раундов шифрования в алгоритме AES256 больше, чем в AES192, видимых различий не наблюдается. Интересно, что шифрование табличного пространства (за исключением алгоритма DES) показывает лучшие результаты. Во всех случаях количество логических и физических операций чтения остается примерно одинаковым, причины некоторой вариации времени ожидания ввода/вывода не ясны, однако могут объясняться особенностями физического расположения блоков в файлах и кешированием данных не только на уровне СУБД, но и на уровне ОС.

Стоит отметить, что план выполнения запроса для таблиц с шифрованием столбца отличался от стандартного: предикатом поиска является *INTERNAL_FUNCTION(VALUE) LIKE ‘%DES’*;

Таблица 3.

Статистики производительности запроса №2

№ конф.	Среднее время выполнения	Среднее время ЦП	Среднее время ввода/вывода	Логических чтений	Физических чтений
1	85836	83136	7353	3399	3382
2	94953	90050	7357	3399	3382
3	5655801	5513493	23226	8858	8832
4	6628353	6507118	360989	333513	166842
5	5601956	5522640	24676	8858	8832
6	14820753	14597817	17756	7563	7540
7	151440	146685	7550	3339	3321
8	142594	135666	7436	3399	3382
9	1582232	1561655	7878	3355	3337

В данном случае таблицы, использующие алгоритмы 3DES, так же являются аутсайдерами. Между таблицами одного типа, использующими разные алгоритмы семейства AES, также нет существенной разницы.

Различий между конфигурациями 1 и 2 практически нет, что неудивительно – согласно описанию технологии, доступ к нешифрованным столбцам (а следовательно, и значениям индекса) не влечет никаких дополнительных операций.

Интересен скачок времени выполнения между конфигурациями 2 и 3. Кроме того, возрастает число чтений. Согласно плану выполнения запроса для №3, сканирование индекса производится по выражению *INTERNAL_FUNCTION(KEY) LIKE ‘56%’*, вместо *KEY LIKE ‘56%’*. Из этого можно сделать вывод, что в индексе хранятся зашифрованные значения полей, что логично, т.к. бессмысленно было бы шифровать столбец, если бы его значения хранились в

открытом виде в другом месте БД. Отсюда следует, что сам по себе индекс(являющийся B-деревом), во многом теряет позиции как средство ускорения производительности. Поскольку результат шифрования статистически не зависит от шифруемого значения, структура B-дерева нарушается: если для значений A, B выполняется A>B, то для их шифров выполнение условий ШИФР(A)>ШИФР(B) и ШИФР(A)<ШИФР(B) одинаково вероятно, и потому неизвестно, где в итоге эти значения окажутся в дереве. Однако гарантируется, что ШИФР(A)=ШИФР(B) тогда и только тогда, когда A=B, что может быть полезно при поиске. Отсюда же очевидно и ограничение на создание индексов по столбцам с использованием опции SALT – её указании нарушается последнее равенство.

Хотя различий по времени выполнения между конфигурациями 3 и 4 немного, на два порядка возрастает число чтений – по сути, происходит обычный поиск по таблице.

Интересно резкое падение времени выполнения при переходе от шифрования столбца к шифрованию табличного пространства. Ответ находится в плане исполнения – предикатом поиска снова становится *KEY LIKE '56%'*. Отсюда можно сделать предположение о том, что данные расшифровываются непосредственно при операциях чтения (что логично – на уровне табличного пространства СУБД оперирует отдельными блоками, а не конкретными значениями столбцов и индексов), а не обращения к значениям индекса, что ведет к уменьшению загрузки ЦП.

Способы увеличения производительности.

Важным моментом является то, что индексы в данной работе явно создаются в том же табличном пространстве, что и таблицы, как следствие, доступ к ним порождает лишние операции дешифрования. Возможно перенести индекс в нешифрованное табличное пространство. Вопрос о переносе решается согласно секретности данных в столбце – если приемлемо хранить их незашифрованными, то лучше произвести перенос.

Все замеры производились при постоянных сбросах буфера, с целью исследовать производительность шифрования при операциях ввода-вывода. Для небольших объектов (таблиц и индексов) Oracle автоматически кеширует считанные блоки, как следствие, при использовании шифрования табличных пространств в буфер будут попадать уже расшифрованные блоки. В случае шифрования отдельных столбцов вопрос о величине пользы кеширования остается открытым, поскольку во всех планах исполнения используется сравнение с результатом INTERNAL_FUNCTION() – функции расшифрования.

Из результатов замеров очевидно, что алгоритмы семейства AES превосходят 3DES по скорости работы, их использование предпочтительно. Поскольку между собой алгоритмы по скорости работы различаются слабо, вопрос о выборе алгоритма решается сообразно с требуемой стойкостью шифрования.

Выводы:

В данной работе были рассмотрены различные конфигурации прозрачного шифрования. Установлено, что алгоритмы семейства AES на порядок превосходят алгоритм 3DES по скорости работы. Установлено, что использование шифрования табличных пространств работает быстрее, чем отдельных столбцов в однообразных операциях с колонками больших объемов данных. Вопрос об влиянии шифрования на производительность в комплексных реальных запросах подлежит дальнейшему рассмотрению. Исследовано влияние шифрования на индексы. Согласно результатам, предпочтительнее использовать индексы в сочетании с шифрованием табличных пространств и, если возможно, переносить их в нешифрованные табличные пространства. Установлено, что шифрование отдельных индексируемых столбцов крайне отрицательно влияет на производительность индекса вследствие фактического нарушения логики работы B-дерева. По результатам работы даны рекомендации по повышению производительности.

Список литературы:

1. Алапати, С.Р. Oracle Database 11g: руководство администратора баз данных.: Пер с англ. – М. : ООО «И.Д. Вильямс», 2010. – 1440 с.

2. D.C.Knox. Applied Oracle Security: Developing Secure Database and Middleware Environments / D.C.Knox, S.G.Gaetjen, H.Jahandir et al. – N.Y.: Oracle Press, 2009. – P. 640