Авторы:

Танаев Иван Владимирович

студент

Швейкин Владислав Витальевич

студент

Дмитриев Егор Андреевич

студент

Завгородний Станислав Дмитриевич

студент

ФГАОУ ВО «Самарский национальный

исследовательский университет

им. академика С.П. Королева»

г. Самара, Самарская область

ПРОЗРАЧНОЕ ШИФРОВАНИЕ В СУБД ORACLE

Аннотация: *статья посвящена методу прозрачного шифрования данных. Авторы отмечают, что данный метод предоставляет возможность в кратчайшие сроки обеспечить информационную безопасность базы данных в соответствии с нормативными документами.*

Ключевые слова: *шифрование, технология прозрачного шифрования, ключи, СУБД, базы данных, информационная безопасность.*

Основные определения

1. СУБД (система управления базами данных) – программно-аппаратный комплекс, позволяющий управлять и манипулировать базами данных.

2. Симметричное шифрование – способ шифрования с использованием одного и того же криптографического ключа для шифрования и расшифрования.

3. Ключ-секретная информация, используемая в криптографических алгоритмах.

Введение

На сегодняшний день вычислительная мощность ЭВМ для обработки данных в СУБД неуклонно растет. Тем не менее для шифрования данных в таблицах применяется симметричное шифрование. При этом у пользователя могут возникнуть определенные трудности, связанные с управлением большим количеством ключей, необходимых для защиты данных. А использование малого числа ключей также нежелательно, потому что приводит к раскрытию большого объема информации при их компрометации.

Прозрачное шифрование

Универсальным решением этой проблемы является использование метода прозрачного шифрования данных (transparent data encryption). Этот метод подразумевает шифрование информации перед записью на диск и дешифрование во время чтения в память, что решает проблему защиты «неиспользуемых» данных, но не обеспечивает защиту информации при передаче по каналам связи или во время обработки. Технология прозрачного шифрования применяется для файлов базы данных на уровне столбцов. Для таблицы, имеющей столбцы, подлежащие шифрованию, создается симметричный ключ, который, в свою очередь, защищен главным ключом, находящимся в специальном wallet-файле за пределами базы данных. Зашифрованные ключи таблицы содержатся в словаре данных (DataDictionary).

Пользователь может задать место хранения wallet-файла (бумажника), указав значение параметра Encryption_wallet_location в файле sqlnet.ora. По умолчанию в СУБД Oracle значение данного параметра равно $ORACLE_BASE/admin/$ORACLE_SID/wallet. Для этого необходимо создать соответствующий каталог вручную, при этом стоит отметить, что крайне нежелательно хранить критические важную информацию в стандартном месте. Для повышения надежности хранения информации лучше использовать внешние носители (например, флеш-память). Чтобы задать место хранения wallet-файла необходимо в файл $ORACLE_HOME/network/admin/sqlnet.ora добавить команду:

ENCRYPTION_WALLET_LOCATION =

(SOURCE =

(METHOD = file)

(METHOD_DATA =

(DIRECTORY = «путь_к_файлу»)

)

)

Чтобы изменения вступили в силу, необходимо перезапустить сервер базы данных.

Следующим шагом необходимо сгенерировать ключевую информацию. Для создания wallet-файла используется команда ALTERSYSTEMSETENCRYPTIONKEYIDENTIFIEDBY «пароль». При этом пользователь должен обладать привилегией ALTERSYSTEM.

Чтобы изменить таблицу, а именно зашифровать столбец, необходимо открыть wallet-файл (предъявить и указать правильный пароль) командой ALTERSYSTEMSETENCRYPTIONWALLETOPENAUTHENTICATED BY «пароль». Для получения информации в зашифрованном столбце также необходимо, чтобы wallet-файл был открыт. После выполнения необходимых операций wallet-файл следует закрыть: ALTERSYSTEMSETENCRYPTIONWALLETCLOSE.

```
ORA-28353: сбой при отрытии футляра LUMN2 encrypt);
alter table tdetable modify (COLUMN2 encrypt)
*
ошибка в строке 1:
ORA-28365: футляр не открыт
SQL> alter system set encryption wallet open authenticated by "walletpassword";

Система изменена.

SQL> alter table tdetable modify (COLUMN2 encrypt);

Таблица изменена.

SQL> desc tdetable
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 COLUMN1                                   NOT NULL NUMBER
 COLUMN2                                            VARCHAR2(20) ENCRYPT

SQL> select * from tdetable;

   COLUMN1 COLUMN2
---------- --------------------
         1 one
         2 two
         3 three
         4 four
         5 five
SQL> alter system set encryption wallet close;
Система изменена.
SQL> select * from tdetable;
select * from tdetable
              *
ошибка в строке 1:
ORA-28365: футляр не открыт

SQL> CONNECT BADUSER/baduserpassword
Соединено.

SQL> select COLUMN2 from tdeuser.tdetable;
select COLUMN2 from tdeuser.tdetable
                           *

ошибка в строке 1:
ORA-28365: футляр не открыт
SQL> alter system set encryption wallet open authenticated by "wrongpassword";
alter system set encryption wallet open authenticated by "wrongpassword";
*
ошибка в строке 1:
ORA-28353: сбой при отрытии футляра
```

Рис. 1. Демонстрация технологии прозрачного шифрования

Отметим, что загрузка данных в таблицу и их последующее шифрование при помощи команды ALTERTABLE не может обеспечить высокую степень защищенности БД. Потому что сканирование файла табличной области, которая

содержит таблицу, показывает, что после завершения процесса прозрачного шифрования, исходные данные в незашифрованном виде не стираются. При анализе дампа табличной области, посторонний пользователь может получить важную информацию о содержании зашифрованных данных. Особенно, если в качестве исходных данных, используются символьные строки. Любой пользователь, который может выполнять запрос к словарю DBA_ENCRYPTED_COLUMNS, способен получить перечень зашифрованных столбцов и узнать исходный тип данных.

Допустим, злоумышленнику известен пароль, указанный при создании бумажника для шифрования столбца. Тогда попытка создать «логическую копию» wallet-файла получить доступ к данным.

```
SQL> connect BADUSER/baduserpassword
Соеденино.

SQL> alter system set encryption key identified by "walletpassword";

Система изменена.

SQL> select * from tdeuser.tdetable;
select * from tdeuser.tdetable;
                      *

ошибка в строке 1:
ORA-28362: главный ключ не найден
```

Рис. 2. Ошибка чтения данных при использовании копии бумажника

Существенным недостатком данной технологии является невозможность её реализации на системах с повышенными требованиями к информационной безопасности, так как при открытии бумажника, таблица также становится открытой для всех пользователей, имеющих право доступа к ней.

Недостатком данного способа для систем, обладающих высокими требованиями к безопасности, является ограниченная область применения технологии прозрачного шифрования. Кроме того, будет весьма проблематично работать с данными, если пользователь, обладающий привилегией ALTERSYSTEM, выполнит команду закрытия wallet-файла

(ALTERSYSTEMSETENCRYPTIONWALLETCLOSE), потому что для выполнения данной команды не требуется указывать пароль.

Шифрование данных с неявным заданием ключа позволяет изменять ключевую информацию. Если ключевая информация скомпрометирована, необходимо изменить ключевую информацию (ALTERTABLEс параметром REKEY) и выполнить новую процедуру шифрования данных. Но если злоумышленник получил копию бумажника и пароль, то несмотря на заявленное в документации перешифрование данных, изменение бумажника не произойдет и данные читаются с использованием старого бумажника.

Преимущества и недостатки

Одним из главных преимуществ является отсутствие необходимости использовать дополнительные решения для защиты информации при передаче по каналам связи, так как она передается в зашифрованном виде. Также злоумышленнику необходимо иметь доступ к приложению, чтобы расшифровать данные, хранящиеся в БД, что существенно уменьшает возможность кражи конфиденциальной информации.

Тем не менее, для использования технологии прозрачного шифрования на уровне приложений возникает необходимость во внесении изменений не только в приложение, но и в базу данных. Наряду с этим могут возникнуть проблемы с производительностью БД, индексированием и поиском. Еще одним недостатком является управление ключами в системе с таким шифрованием, так как несколько приложений могут использовать базу данных, и ключи хранятся в разных местах, что при неправильном управлении ими может привести к потере доступности, целостности, конфиденциальности. Кроме того, при возникновении необходимости смены ключа, требуется расшифровать данные старым ключом и потом зашифровать, использую новый ключ.

Заключение

В заключение следует отметить, что на сегодняшний день известно множество способов защиты информации, каждый из которых имеет свои преимуще-

ства и недостатки. Технология прозрачного шифрования предоставляет возможность в кратчайшие сроки обеспечить информационную безопасность базы данных в соответствии с нормативными документами.

Список литературы

1. Смирнов С.Н. Безопасность систем баз данных. – М.: Гелиос АРВ, 2007. – 352 с.

2. Дейт К.Дж. Введение в системы баз данных. – 8-е изд. – М.: Вильямс, 2005. – 1328 с.