В результаті проведених досліджень встановлено, що при застосуванні житніх заквасок виведених на основі створених нових композицій молочнокислих бактерій, покращується якість житньо-пшеничного хліба порівняно з хлібом, у виготовлення якого не використовували закваску.

Список використаних джерел:

1. Афанасьева, О.В. Микробиология хлебопекарного производства / О.В. Афанасьева. – СПб.: Береста, 2003. – 221 с.
2. Дробот, В.І. Довідник з технології хлібопекарського виробництва: навч. посіб. для студ. вищ. навч. закл. / В І. Дробот. – К.: Руслана, 1998. – 415 с.
3. Дробот, В.І. Технологія хлібопекарського виробництва / В.І. Дробот. – К.: Логос, 2002. – 365 с.
4. Квасников, Е.И. Молочнокислые бактерии и пути их использования / Е.И. Квасников, О.А. Нестеренко. – К.: Наука, 1974. – 390 с.
5. Существует огромное количество самых различных заквасок. Вот некоторых из них – Подготовка пресс-служба редакции // Хлібопекарська і кондитерська промисловість України. – 2010. – № 4. – С. 24-26.

Погромська Г.С.
кандидат педагогічних наук, доцент,
Миколаївський національний університет
імені В.О. Сухомлинського

ВБУДОВАНІ МЕХАНІЗМИ СУБД ДЛЯ ВИБІРКОВОГО ЗАХИСТУ ДАНИХ

Інформація в сучасному світі перетворилася в один з найбільш важливих ресурсів суспільства, а інформаційні системи, фундаментальним компонентом яких є бази даних, стали необхідним інструментом практично у всіх сферах діяльності людини, надаючи йому достовірну інформацію для прийняття оптимального рішення. При цьому захист баз даних залишається однією з найскладніших завдань, що стоять перед підрозділами, що відповідають за забезпечення інформаційної безпеки, так як часто чіткої і ясної методики комплексного її вирішення, яку можна було б застосовувати у всіх випадках, не існує.

Дослідження проблем інформаційної безпеки [1, 2] показують, що серед максимальних за обсягом втрат різних організацій від атак різних видів [4] є такі, які безпосередньо є наслідком недостатнього захисту інформації в базах даних, а саме: крадіжка конфіденційних даних та неавторизований доступ до даних.

Проблема захисту даних в СУБД, як і в будь-який інший інформаційній системі, може бути розкладена на три завдання по забезпеченню: конфіденційності даних, цілісності даних, доступності даних.

Розглянемо завдання забезпечення конфіденційності і цілісності даних. Для початку розглянемо, на яких рівнях можна захистити інформацію в БД, тобто

визначимо об'єкти захисту. По-перше, можна захищати дані, що зберігаються в різних структурних елементах БД: цілком бази даних; окремі таблиці БД; записи конкретних таблиць БД; значення конкретних полів таблиць або записів.

По-друге, логічні структурні елементи БД фізично зберігаються на жорстких дисках і передаються по комп'ютерних мережах, отже, можна захищати дані за місцем їх фізичного втілення, наприклад: файли даних; програмні модулі СУБД (від модифікації і впровадження програмних закладок, що змінюють логіку роботи СУБД в будь-яких цілях атакуючого); канали взаємодії між компонентами СУБД (особливо актуально в разі розподілених СУБД) і канали взаємодії між СУБД та користувачами.

Захищати програмні модулі, а також інформацію, що зберігається на жорсткому диску і передану по мережі, можна точно так само, як і будь-яку аналогічну інформацію, що не відноситься до СУБД. Наприклад, за допомогою моделей доступу, а також за допомогою криптографічних методів:

- шифрування файлів і мережевого трафіку допоможе забезпечити конфіденційність;
- методи контролю цілісності (наприклад, хешування) дозволять зберігати цілісність даних і програмних модулів.

Більш цікавою є задача вибіркового захисту даних, що зберігаються в конкретних полях або записах БД. В даному випадку будь-які методи захисту на файловому рівні не допомагають і здається необхідним використовувати специфічні для БД методи захисту, зокрема: уявлення (views); тригери; вбудовані функції шифрування даних.

Розглянемо ці методи. Уявлення – це пойменована динамічно підтримувана сервером вибірка з однієї або декількох таблиць [2]. Іншими словами, уявлення є віртуальною таблицею, всі записи якої формуються в процесі звернення користувача до подання згідно з тим запитом, який був зіставлений даного подання. Таким чином, користувач отримує доступ не до цілої таблиці (або до декількох таблиць), а, в найпростішому випадку, до сукупності стовпців або записів, які визначені в уявленні.

Уявлення є найпростішим у використанні методом захисту як конфіденційності, так і цілісності даних, який дозволяє:

- чітко обмежити дані, доступні користувачеві;
- контролювати той набір даних, який користувач має право модифікувати.

Тригер (trigger) – це збережена процедура особливого типу, яку користувач не викликає безпосередньо, а виконання якої обумовлено настанням певної події (дією) – по суті додаванням INSERT або видаленням DELETE рядків в заданій таблиці, або модифікації UPDATE даних в певному стовпці заданої таблиці реляційної бази даних [5]. Таким чином, тригер – це певна підпрограма, що спрацьовує автоматично в разі модифікації даних в таблиці. Причому, тригер може спрацьовувати як до, так і після настання конкретної події, що визначається ключовими словами AFTER і BEFORE відповідно.

З точки зору захисту даних в БД тригери дозволяють:

- перевірити повноваження користувача – чи має користувач право на модифікацію конкретних даних; важливо, що така перевірка може бути

виконана до модифікації даних, яку тригер може скасувати при недостатності прав користувача на дану операцію;

- протоколювати (наприклад, в окремій таблиці аудиту) всі події, пов'язані з модифікацією даних в таблиці, що захищається – це є зручним, наприклад, при пошуку користувача, який виконав несанкціоновану модифікацію даних.

В [5] наведено приклад найпростішого тригера, який записує в таблицю аудиту (info) інформацію про те, що сталися зміні рядка таблиці district, для якої визначається тригер (нотація СУБД Oracle):

```
CREATE OR REPLACE TRIGGER DistrictUpdatedTrigger AFTER UPDATE
ON district FOR EACH ROW
BEGIN
INSERT INTO info VALUES («one string in table «district» has changed»);
END;
```

Відзначимо, що тригери вкрай прості у використанні, але при цьому дозволяють ефективно контролювати цілісність даних, а також протоколювати події їх модифікації.

Варто зазначити, що вбудовані функції шифрування присутні далеко не у всіх СУБД. Отже, універсальним даний метод назвати не можна.

Розглянемо функції шифрування на прикладі СУБД MySQL 5.5 [7]. Дана СУБД пропонує два однотипних набори функцій шифрування, в одному з яких реалізований алгоритм DES, а в іншому – AES. Крім того, в MySQL реалізовано декілька алгоритмів хешування. Набір криптографічних функцій даної СУБД виглядає так (див. табл. 1):

Таблиця 1

Набір криптографічних функцій

№	Функція	Призначення
1	AES_ENCRYPT()	Зашифрування даних алгоритмом AES
2	AES_DECRYPT()	Розшифрування даних алгоритмом AES
3	DES_ENCRYPT()	Зашифрування даних алгоритмом DES
4	DES_DECRYPT()	Розшифрування даних алгоритмом DES
5	ENCRYPT()	Зашифрування даних функцией crypt()
6	MD5()	Хешування даних алгоритмом MD5
7	SHA1() или SHA()	Хешування даних алгоритмом SHA-1

Функції шифрування даних алгоритмом AES використовують 128-бітний ключ шифрування, тобто шифрування ключами розміром 192 і 256 біт, передбаченими стандартом AES [3; 6], в MySQL не реалізоване. Ключ шифрування задається явно як один з параметрів функції.

На відміну від них, функції DES_ENCRYPT () і DES_DECRYPT (), які шифрують алгоритмом TripleDES, крім явного завдання ключа шифрування, допускають найпростіший варіант управління ключами у вигляді ключового файлу, що містить пронумеровані значення ключів. Однак, дані функції за замовчуванням вимкнені, для їх використання необхідно включити підтримку протоколу SSL в конфігурації СУБД.

Функція ENCRYPT () може бути використана тільки в операційних системах сімейства Unix, оскільки вона шифрує дані за допомогою системного виклику crypt ().

Що стосується використовуваних функцій хешування, то в документації на MySQL [7; 8] міститься попередження про те, що алгоритми, які лежать в їх основі, зламані. Однак, MySQL поки не пропонує більш стійких функцій хешування замість існуючих.

В MySQL за замовчуванням не використовується шифрування при з'єднаннях, так як це значно уповільнює обмін даними між клієнтом і сервером. Будь-які додаткові функції призводять до додаткового навантаження для комп'ютера, а шифрування даних вимагає інтенсивної роботи процесора, що може викликати затримку виконання основних завдань MySQL. За замовчуванням MySQL налаштований на максимально швидку роботу.

Розглянуті вище методи захисту даних вбудованими засобами СУБД прості у використанні, але можуть досить ефективно забезпечити конфіденційність і цілісність даних. Їх об'єднує один недолік – захист з їх допомогою досить складно накласти на вже існуючу систему. Проте, на етапі проектування захищених БД завжди варто передбачити додатковий рубіж захисту у вигляді сукупного використання уявлень, тригерів і вбудованих криптографічних функцій СУБД.

Список використаних джерел:

1. Есин В. И. Безопасность информационных систем и технологий / В. И. Есин, А. А. Кузнецов, Л. С.Сорока. – Х.: ООО «ЭДЭНА», 2010. – 656 с.

2. Лихоносов А. Безопасность баз данных. Учебное пособие / А. Лихоносов. – МФПА, 2010. – 390 с.

3. Мельников В. В. Безопасность информации в автоматизированных системах / В. В. Мельников. – М.: Финансы и статистика, 2003. – 368 с.

4. Погромська Г.С. Розробка програмного забезпечення моделювання атак типу SQl-ін'єкцій до баз даних // Zbiór artykułów naukowych Konferencji Miedzynarodowej Naukowo Praktycznej organizowanej dla pracowników naukowych uczelni, jednostek naukowo-badawczych «Obiecujące osiągnięci a naukowe Inżynieria i technologia» (30.09.2017) – Warszawa: Wydawca: Sp. z o.o. «Diamond trading tour», 2017. – 40 str. – S. 8-10.

5. Триггер (базы данных) //Википедия. Свободная энциклопедия [Електронний ресурс].– Режим доступу: http://ru.wikipedia.org.

6. FIPS Publication 197. Specification for the Advanced Encryption Standard. – November 26, 2001 [Електронний ресурс].– Режим доступу: http://csrc.nist.gov.

7. MySQL Documentation: MySQL 5.5 Reference Manual /Офіційний сайт MySQL [Електронний ресурс]. – Режим доступу: https://dev.mysql.com/doc/.

8. Panasenko S. SHA Hash Functions: History & Current State / S. Panasenko [Електронний ресурс].– Режим доступу: http://www.panasenko.ru.