## Тема: *Угрозы, специфичные для систем управления базами данных*

Существует несколько оснований для классификации угроз, специфичных для систем управления базами данных. Будем использовать упрощенную классификацию угроз по следующим основаниям: угрозы конфиденциальности информации, угрозы целостности информации и угрозы доступности.

**Угрозы конфиденциальности информации.**

К угрозам такого типа можно отнести:

1.  *Инъекция SQL*. Во многих приложениях используется динамический SQL — формирование SQL-предложений кодом программы путем конкатенации строк и значений параметров. Зная структуру базы данных, злоумышленник может либо выполнить хранимую программу в запросе, либо закомментировать «легальные» фрагменты SQL-кода, внедрив, например, конструкцию UNION, запрос которой возвращает конфиденциальные данные. В последнее время даже появились специальные программы, автоматизирующие процесс реализации подобных угроз.

2.  *Логический вывод* на основе функциональных зависимостей. Пусть дана схема отношения: http://www.e-biblio.ru/book/bib/01_informatika/b_baz_dan/sg.files/image002.png. Пусть http://www.e-biblio.ru/book/bib/01_informatika/b_baz_dan/sg.files/image003.png, X, Y — подмножества из U. Говорят, что Х функционально определяет Y, если в любом отношении r со схемой http://www.e-biblio.ru/book/bib/01_informatika/b_baz_dan/sg.files/image002.png не могут содержаться два кортежа с одинаковыми значениями атрибутов из Х, с различными из Y. В этом случае имеет место функциональная зависимость, обозначаемая http://www.e-biblio.ru/book/bib/01_informatika/b_baz_dan/sg.files/image004.png. Пример функциональной зависимости для схемы отношения (фамилия, имя, отчество, должность, зарплата): если должность = менеджер, то зарплата = 1200. В реальных базах данных при наличии сведений о функциональных зависимостях злоумышленник может вывести конфиденциальную информацию при наличии доступа только к части отношений, составляющих декомпозированное отношение.

3.  *Логический вывод* на основе ограничений целостности. Для кортежей отношений в реляционной модели данных (РМД) можно задать ограничения целостности — логические условия, которым должны удовлетворять атрибуты кортежей. Причем ограничение целостности может быть задано в виде предиката на всем множестве атрибутов кортежа. В случае попытки изменения данных в таблице, СУБД автоматически вычисляет значение этого предиката, и в зависимости от его истинности операция разрешается или отвергается. Выполняя многократные изменения данных и анализируя реакцию системы, злоумышленник может получить те сведения, к которым у него отсутствует непосредственный доступ. Также к этому виду угроз конфиденциальности относится анализ значений первичных/внешних ключей.

4.  *Использование oneраmopa UPDATE* для получения конфиденциальной информации. В некоторых стандартах SQL пользователь, не обладая привилегией на выполнение оператора SELECT, мог выполнить оператор UPDATE со сколь угодно сложным логическим условием. Так как после выполнения оператора UPDATE сообщается, сколько строк он обработал, фактически пользователь мог узнать, существуют ли данные, удовлетворяющие этому условию.

Рассмотрим угрозы целостности информации, специфические для систем управления базами данных.

Модификация данных в реляционных СУБД возможна с помощью SQL-операторов UPDATE, INSERT и DELETE. Потенциальная опасность возникает из-за того, что пользователь, обладающий соответствующими привилегиями, может модифицировать все записи в таблице. Ограничить множество записей, доступных для модификации, можно с помощью создания представлений с оператором CHECK, но этот (равно как и любой другой) требует предварительного осмысливания существа задачи и соответствующего проектирования схемы.

**Специфичными для систем управления базами данных угрозами доступности являются:**

1.  Использование свойств первичных и внешних ключей. В первую очередь сюда относится свойство уникальности первичных ключей и наличие ссылочной целостности. В том случае, если используются натуральные, а не генерируемые системой значения первичных ключей, можно создать такую ситуацию, когда в таблицу невозможно будет вставить новые записи, так как там уже будут записи с такими же значениями первичных ключей. Если в базе данных поддерживается ссылочная целостность, можно организовать невозможность удаления родительских записей, умышленно создав подчиненные записи. Важной особенностью реализации ссылочной целостности является вопрос об индексировании внешнего ключа. В том случае, если внешний ключ не проиндексирован, то при обновлении связанных записей, например, в СУБД Oracle возможна организация взаимной блокировки (dead-lock), что приведет к сбою в транзакции.

2.  Блокировка записей при изменении. Заблокировав записи или всю таблицу, злоумышленник может на значительное время сделать ее недоступной для обновления.

3.  Загрузка системы бессмысленной работой. Простейший пример — выполнение запроса, содержащего декартово произведение двух больших отношений. Мощность декартового произведения двух отношений мощности http://www.e-biblio.ru/book/bib/01_informatika/b_baz_dan/sg.files/image005.png, и http://www.e-biblio.ru/book/bib/01_informatika/b_baz_dan/sg.files/image006.png равна http://www.e-biblio.ru/book/bib/01_informatika/b_baz_dan/sg.files/image007.png. То есть при выдаче злоумышленником запроса вида SELECT \* FROM Tab 1, Tab 1 ORDER BY 1, где мощность отношения (количество строк в таблице Tab 1) http://www.e-biblio.ru/book/bib/01_informatika/b_baz_dan/sg.files/image008.png, мощность результирующего отношения будет http://www.e-biblio.ru/book/bib/01_informatika/b_baz_dan/sg.files/image009.png. Вычисление соединения (особенно, если указать в виде подсказки оптимизатору способ соединения, требующий значительных затрат процессорного времени, — сортировку слиянием или хэширование) и сортировка результирующего отношения потребуют значительных ресурсов системы и отрицательно скажутся на производительности операций других пользователей.

4.  В реляционных СУБД возможны реализации и других классических угроз, например атаки типа «троянский конь» — запуска пользователями программ, содержащих выполняющий определенные действия код, внедренный туда злоумышленником.

Практически все современные СУБД имеют встроенный процедурный язык программирования (PL/SQL, Transact SQL и т. п.). Программы на этом языке хранятся внутри базы данных и выполняются исполняющей подсистемой сервера баз данных. Наличие и широкое использование производителями прикладного программного обеспечения механизма скрытия исходных текстов хранимых программ (утилита wrap для СУБД Oracle) затрудняют его обнаружение. Также возможны многочисленные скрытые каналы, обусловленные семантикой данных и необходимостью обеспечения работы в условиях многопользовательского доступа.

На основании проведенного анализа можно сделать вывод, что в современных реализациях реляционных СУБД имеется значительное число уязвимостей, которые могут быть использованы для различных атак на информационные системы, построенные на их базе. Эта проблема отчасти может быть решена использованием специализированных программных средств анализа уязвимостей и защиты от угроз различных типов.