Интернет-журнал «Науковедение» ISSN 2223-5167 http://naukovedenie.ru/

Том 9, №3 (2017) http://naukovedenie.ru/vol9-3.php

URL статьи: http://naukovedenie.ru/PDF/36TVN317.pdf

Статья опубликована 20.06.2017

Ссылка для цитирования этой статьи:
Исаева М.Ф., Глухарев М.Л., Ветлугин К.А. Реализация многоуровневой модели разграничения доступа в базах данных под управлением системы управления базами данных Microsoft SQL Server // Интернет-журнал «НАУКОВЕДЕНИЕ» Том 9, №3 (2017) http://naukovedenie.ru/PDF/36TVN317.pdf (доступ свободный). Загл. с экрана. Яз. рус., англ.

УДК 004.056.52

Исаева Мария Феликсовна
ФГБОУ ВО «Петербургский государственный университет путей сообщения императора Александра I»
Россия, Санкт-Петербург[1]
Аспирант кафедры «Информатика и информационная безопасность»
E-mail: isaeva.mf@gmail.com
РИНЦ: http://elibrary.ru/author_profile.asp?id=916016

Глухарев Михаил Леонидович
ФГБОУ ВО «Петербургский государственный университет путей сообщения императора Александра I»
Россия, Санкт-Петербург
Доцент кафедры «Информатика и информационная безопасность»
Кандидат технических наук
E-mail: mlgluharev@yandex.ru
РИНЦ: http://elibrary.ru/author_profile.asp?id=721065

Ветлугин Константин Александрович
ФГБОУ ВО «Петербургский государственный университет путей сообщения императора Александра I»
Россия, Санкт-Петербург
Аспирант кафедры «Информатика и информационная безопасность»
E-mail: k.a.vetlugin@yandex.ru

Реализация многоуровневой модели разграничения доступа в базах данных под управлением системы управления базами данных Microsoft SQL Server

Аннотация. Для удобства хранения и обработки больших объемов информации используются базы данных, находящиеся под управлением специализированного программного обеспечения - систем управления базами данных. В данной статье приведено описание и проанализирована используемая в настоящее время подсистема управления доступом пользователей. Также в работе продемонстрирована разработка программной надстройки, представляющей собой механизм разграничения доступа в базах данных, который может применяться для настройки работы с пользовательскими базами данных. При помощи данной надстройки администратор безопасности может выбрать, какую политику разграничения доступа использовать в работе - меточную или мандатную политику разграничения доступа. При разработке программной надстройки были учтены и исправлены недостатки существующих рекомендаций компании Microsoft по реализации многоуровневой

[1] 190031, Россия, Санкт-Петербург, Московский пр., д. 9

модели доступа, в которой предлагается настраивать безопасность на уровне строк с использованием представлений и меток безопасности. Кроме того, авторами представлена разработанная мандатная модель Белла-ЛаПадулы. Мандатная модель основана на правилах, согласно которым всем участникам процесса обработки критичной информации и документам, в которых она содержится, присваивается специальная метка, которая называется уровнем безопасности. Разработанный механизм может использоваться в качестве учебного материала, а также может быть внедрен в организациях, работающих с базами данных в MS SQL Server.

Ключевые слова: базы данных; управление доступом; ограничение доступа; реляционная база данных; системы управления базами данных; многоуровневая модель доступа; меточная политика; мандатная модель

Базы данных (БД) в современном мире широко используются в прикладном программном обеспечении и web-приложениях, предоставляя удобное решение для хранения информации. В некоторых случаях эта информация может стать объектом кражи, что приведет к серьезным проблемам ее владельца, поэтому она нуждается в защите.

Администраторы безопасности баз данных занимаются настройкой, сопровождением и защитой базы данных. Одной из задач администрирования БД является управление доступом пользователей к данным и другим объектам БД. В ходе управления доступом принимается решение о том, какие действия в системе может выполнять пользователь, а также какие операции разрешено выполнять приложениям, запущенным от имени пользователя. Таким образом, правильное управление доступом базой данных может предотвратить умышленные или неумышленные действия пользователя, которые способны нанести вред хранимой информации [1].

Обеспечение безопасности информации при функционировании любой информационно-управляющей системы (АИУС) является важным вопросом в решении вопросов безопасности. Эффективность работы АУИС может быть значительно снижена при наличии ошибок в процессе проектирования системы защиты информации от несанкционированного доступа. Защита информации от несанкционированного доступа осуществляется для обеспечения целостности данных, исключающее их уничтожение или повреждение; для недопущения несанкционированного изменения данных, представляющего собой умышленные или случайные действия по модификации информации; для недопущения несанкционированного получения данных и для недопущения несанкционированного тиражирования данных. [7]

К современным информационным системам часто предъявляется требование разграничения доступа пользователей не только к объектам БД, но и к отдельным записям, с применением многоуровневых моделей доступа. По этой причине целесообразно реализовывать программный механизм, реализующий многоуровневую модель управления доступом на уровне записей, на стороне систем управления базами данных (СУБД). Большая часть современных реляционных СУБД не имеет в своем составе такого механизма, оставшаяся часть содержит в себе этот механизм в качестве дополнительного инструмента, требующего дополнительной оплаты и имеющего ограниченную функциональность [4].

Механизм управления доступом в базах данных служит для обеспечения информационной безопасности СБД. Программные методы защиты информации позволяют реализовать большинство функций защиты: идентификацию пользователей БД, проверку подлинности пользователей, проверку полномочий пользователей на осуществление того или иного типа доступа к защищаемым данным, реагирование на попытку несанкционированного доступа путём прерывания выполнения запроса или отключения терминала, уничтожение

остаточной информации в оперативной памяти компьютеров после удовлетворения санкционированного запроса пользователя и др. [8, 9]

Кроме того, они могут поддерживать различные стратегии разграничения доступа пользователей к ресурсам БД: разграничение по спискам, разграничение по матрицам установления полномочий, разграничение по кольцам секретности, мандатную модель доступа. Механизмы защиты (МЗ) применяются с целью организации доступа и допуска к защищенной информации только тех групп пользователей, которые обладают соответствующими полномочиями и прошли авторизацию [6].

В качестве базовой модели доступа в СУБД Microsoft SQL Server используется ролевая модель. Правила доступа в ней описываются в виде троек *<субъект, объект, тип доступа>*. *Субъектами* доступа (участниками) могут быть учетные записи серверного уровня, пользователи баз данных, серверные роли и роли баз данных. Учетные записи и серверные роли наделяются привилегиями серверного уровня, каждая из которых дает возможность выполнять то или иное действие на сервере: устанавливать соединение с сервером, просматривать список имеющихся баз данных, создавать новые базы данных, изменять свойства существующих баз данных или удалять их, управлять настройками сервера (в т.ч. настройками безопасности) и т.п. Пользователи и роли баз данных наделяются привилегиями доступа к объектам тех баз данных, в которых они зарегистрированы: привилегиями чтения/записи информации в таблицах, запуска хранимых процедур, изменения свойств объектов базы данных, управления безопасностью на уровне базы данных и т.п. [2].

Недостатком базовой модели доступа является невозможность разграничения доступа пользователей к отдельным строкам и ячейкам таблиц, а также отсутствие многоуровневого управления доступом. Для устранения данного недостатка требуется расширение имеющейся подсистемы управления доступом в виде механизма, поддерживающего указанные функциональные возможности.

Меточная политика доступа как основа механизма многоуровневого управления доступом

Компанией Microsoft представлены рекомендации[2] по созданию механизма многоуровневого управления доступом. В документации предлагается настраивать безопасность на уровне строк с использованием представлений и использовать метки безопасности (security labels).

Под *меткой безопасности* понимается та часть информации, которая описывает «чувствительность» (sensitivity) элемента данных (строки). Строка маркируется меткой, описывающей уровни доступа, классифицируемые по одному или нескольким признакам. Пользователи (субъекты) имеют разрешения, описанные с теми же метками. Разрешения каждого субъекта могут рассматриваться в качестве собственных меток. Метка субъекта сравнивается с меткой строки для определения возможности доступа к этому строке.

Привязка метки безопасности к пользователю определяет, к каким строкам таблицы он может получить доступ.

Пусть имеется множество строк (табл. 1), содержащих метки безопасности в столбце Classification.

[2] Implementing Row- and Cell-Level Security in Classified Databases. [Электронный ресурс]. Режим доступа: [https://technet.microsoft.com/library/Cc966395, 10.05.2017].

Таблица 1

ID	Name	Classification
1	Ivan Ivanov	SECRET
2	Peter Petrov	TOP SECRET
3	Michael Sidorov	UNCLASSIFIED

Составлено автором

Также в базе данных может храниться информация о пользователях (табл. 2):

Таблица 2

User	Clearance
Anna	SECRET
Alex	UNCLASSIFIED (no clearance)

Составлено автором

Оба пользователя, Anna и Alex, могут выполнить запрос на выборку вида SELECT * FROM <tablename> к таблице данных (табл. 1), но результаты данного запроса выглядят неодинаково. Пользователь Anna получает набор данных, представленный в табл. 3, пользователь Alex - представленный в табл. 4.

Таблица 3

ID	Name	Classification
1	Ivan Ivanov	SECRET
3	Michael Sidorov	UNCLASSIFIED

Таблица 4

ID	Name	Classification
3	Michael Sidorov	UNCLASSIFIED

Составлено автором

В метку безопасности могут быть включены уровни доступа по нескольким категориям. Например, в дополнение к классификации секретности (с уровнями SECRET, TOP SECRET и UNCLASSIFIED) может использоваться такая категория, как членство пользователя в определенной проектной команде. Предположим, имеется группа PROJECT Q. Изменим данные в табл. 1 таким образом, чтобы метка для первой строки включала уровень секретности SECRET и задавала разрешение на доступ к этой строке только для участников группы PROJECT Q. Результат этого изменения представлен в табл. 5.

Таблица 5

ID	Name	Classification
1	Ivan Ivanov	SECRET, PROJECT Q
2	Peter Petrov	TOP SECRET
3	Michael Sidorov	UNCLASSIFIED

Составлено автором

Теперь изменим права пользователей: включим пользователя Anna в группу PROJECT Q и добавим пользователя Charlie с уровнем безопасности TOP SECRET (табл. 6).

Таблица 6

User	Clearance
Anna	SECRET, PROJECT Q
Alex	UNCLASSIFIED (no clearance)
Charlie	TOP SECRET

Составлено автором

Выполняя запрос вида SELECT * FROM <tablename>, Anna видит результаты, представленные в табл. 7, а Charlie - в табл. 8. Хотя Charlie имеет уровень доступа TOP SECRET, он не включен в группу PROJECT Q, поэтому он не может видеть строку 1. Anna, однако, имеет все права для того чтобы видеть строку 1. Строка 2, требующая метку TOP SECRET, видна только Charlie.

Таблица 7

ID	Name	Classification
1	Ivan Ivanov	SECRET, PROJECT Q
3	Michael Sidorov	UNCLASSIFIED

Составлено автором

Таблица 8

ID	Name	Classification
2	Peter Petrov	TOP SECRET
3	Michael Sidorov	UNCLASSIFIED

Составлено автором

В некоторых областях применения меточной политики разграничения доступа метки безопасности могут включать в себя несколько классов доступа, относящихся к различным категориям, а число возможных комбинаций этих классов меток может быть достаточно большим [4, 10]. В этой связи уместно рассмотреть рекомендуемую компанией Microsoft архитектуру механизма многоуровневого контроля доступа, ориентированную на использование различных (задаваемых администратором) категорий и классов доступа при определении меток безопасности.

Для настройки безопасности предлагается использовать следующие служебные таблицы базы данных (рис. 1).

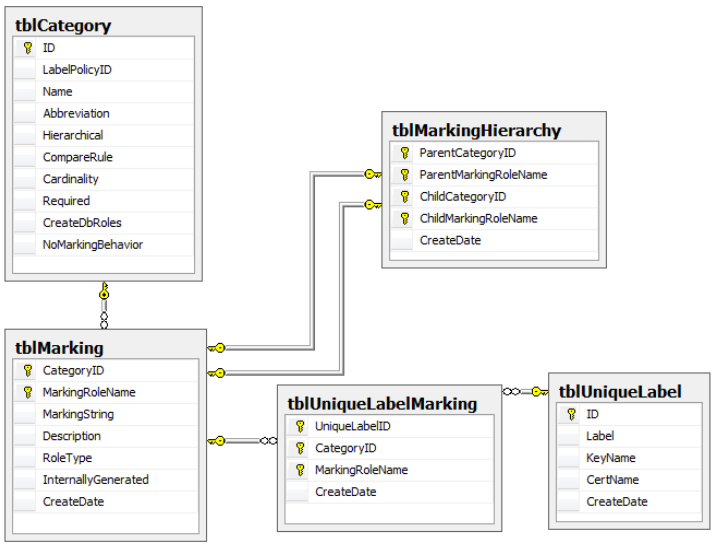

Рисунок 1. *Диаграмма БД (составлено автором)*

В таблице **tblCategory** предлагается хранить категории разграничения доступа. В качестве таковых категорий может выступать уровень защищенности информации, группа пользователей и т.п.

Для каждой категории существует определенный набор возможных маркировок, которые предлагается хранить в таблице **tblMarking.**

Если категория имеет иерархическую структуру, то отношения «родитель-потомок» устанавливаются с соответствующими записями в таблице **tblMarkingHierarchy**.

Таблица **tblUniqueLabel** используется для хранения меток безопасности. Эта таблицу рекомендуется заполнять «по требованию»: при добавлении записей с некоторой меткой безопасности сначала производить поиск этой метки в таблице tblUniqueLabel с помощью специальной хранимой процедуры; при неудачном завершении поиска метка добавляется в таблицу.

В таблице **tblUniqueLabelMarking** отдельные значения меток связываются с экземплярами меток безопасности.

Помимо таблиц, рекомендуется включить в архитектуру служебной БД некоторые программные объекты, поддерживающие контроль безопасности на уровне отдельных записей в автоматическом режиме.

Прежде всего, к таковым объектам относится представление vwVisibleLabels, которое позволяет получить набор меток, доступных для текущего пользователя.

Листинг представления имеет следующий вид:

```
CREATE VIEW vwVisibleLabels
AS
SELECT ID, Label
FROM tblUniqueLabel WITH (NOLOCK)
WHERE
  ID IN --Classification
  (SELECT UniqueLabelID FROM tblUniqueLabelMarking WITH (NOLOCK)
  WHERE CategoryID = 1 AND IS_MEMBER(MarkingRoleName) = 1)
AND --Compartments
  1 = ALL(SELECT IS_MEMBER(MarkingRoleName) FROM tblUniqueLabelMarking
  WHERE CategoryID = 2 AND UniqueLabelID = tblUniqueLabel.ID)
GO
```

Кроме того, требуется обеспечить опосредованный доступ на чтение и запись к пользовательской таблице, чтобы в процессе чтения и записи проводилось сравнение меток безопасности пользователя с метками обрабатываемых строк. Главным объектом здесь является представление для пользовательской таблицы, к которому пользователь должен выполнять запросы на чтение или запись. Представление же построено на запросе к пользовательской таблице, и в этом запросе происходит сравнение меток безопасности, благодаря чему меточная безопасность на уровне строк таблицы контролируется в прозрачном режиме.

Недостатком рассмотренной модели является избыточность (составные внешние ключи в таблицах tblMarkingHierarchy, tblUniqueLabelMarking), дублирование информации о

пользовательских ролях БД (кроме того, что эти роли заранее создаются с помощью встроенных средств SQL Server, информация о них должна быть дополнительно представлена в таблицах tblMarking и tblMarkingHierarchy, что может существенно усложнить настройку механизма управления доступом). Рассмотренное выше представление vwVisibleLabels позволяет определять доступные метки для текущего пользователя только по двум категориям, хотя наличие таблицы tblCategory в архитектуре служебной БД указывает на то, что число категорий не ограничено [3].

Таким образом, после работы с предложенной схемой построения служебных таблицы была получена оптимизированная схема служебных таблиц для организации меточной модели разграничения доступа (рис. 2).

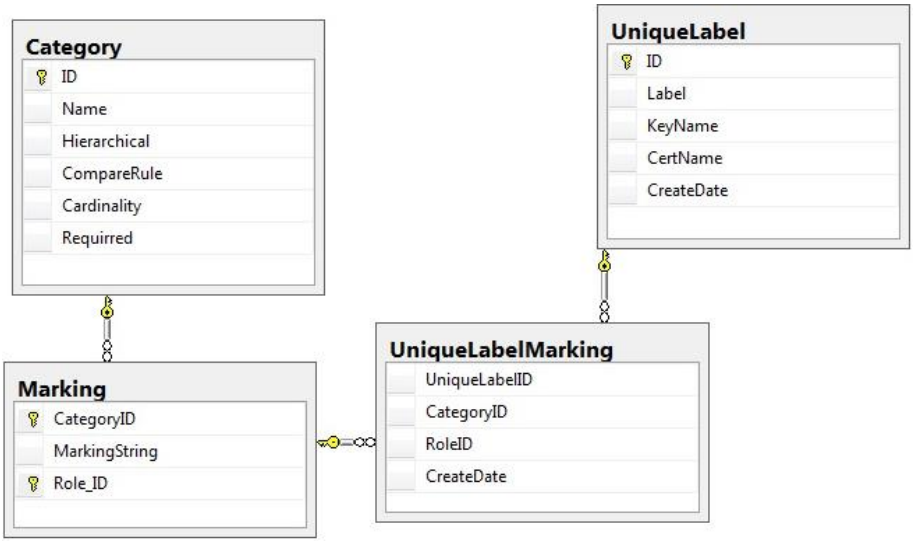

Рисунок 2. *Оптимизированная схема служебных таблиц (составлено автором)*

Данная оптимизация была получена путем устранения составных внешних ключей в таблицах tblMarkingHierarchy, tblUniqueLabelMarking, а также за счет устранения дублирования информации о пользовательских ролях БД в таблицах схемы tblMarking и tblMarkingHierarchy. Как можно заметить, была убрана таблица tblMarkingHierarchy, хранящая информацию об иерархии меток ролей, т.к. вся иерархия хранится в системных таблицах и может быть получена при правильно составленном запросе с использованием служебных представлений select * from sys.Database_Role_Members и select * from Sys.Database_Principals.

Для автоматизации процесса настройки разграничения доступа к пользовательским таблицам был написан ряд программных объектов.

Для автоматизации процесса заполнения служебных таблиц была создана процедура заполнения таблицы UniqueLabelMarking, в которой в качестве параметров передаются идентификатор метки, уровень безопасности и имя роли:

```
create procedure fillingULM (@LabelNumber int, @CategoryName varchar(30), @RoleName varchar(30))

as

begin

declare @CategoryNumber int

declare @RoleNumber int

set @CategoryNumber = (select Category.ID from Category where Category.Name = @CategoryName)
```

set @RoleNumber = (select Marking.Role_ID from Marking,sysusers where Marking.CategoryID=@CategoryNumber and sysusers.name = @RoleName and sysusers.uid=Role_ID)

insert into UniqueLabelMarking values (@LabelNumber,@CategoryNumber,@RoleNumber," ")

end

Заполненные служебные таблицы имеют следующий вид и представлены на рисунках 3-6.

	ID	Name	Hierarchical	CompareRule	Cardinality	Required
1	1	Classification	1	Any	One	1
2	2	Compartment	0	All	Many	0
3	3	Nationality	0	Any	Many	0
4	4	Need-to-Know	0	Any	Many	0

Рисунок 3. *Служебная таблица Category (составлено автором)*

	CategoryID	MarkingString	Role_ID	Mark
1	1	SECRET	5	0
2	1	CONFIDENTIONAL	6	1
3	1	UNCLASSIFIED	9	2
4	1	TOPSECRET	10	3

Рисунок 4. *Служебная таблица Marking (составлено автором)*

	ID	Label	KeyName	CertName	CreateDate
1	1	L1	K1	Cert1	2014-06-12 00:00:00.000
2	2	L2	K2	Cert2	2014-06-12 00:00:00.000
3	3	L3	K3	Cert3	2014-06-12 00:00:00.000
4	4	L4	K4	Cert4	2014-06-12 00:00:00.000

Рисунок 5. *Служебная таблица UniqueLabelMarking (составлено автором)*

	UniqueLabelID	CategoryID	RoleID	CreateDate
1	1	1	10	2014-06-12 15:00:00.000
2	2	1	5	2014-06-12 15:00:05.000
3	3	1	6	1900-01-01 00:00:00.000
4	4	1	9	2014-06-12 15:00:00.000
5	4	1	5	2014-06-12 00:00:00.000
6	3	1	6	1900-01-01 00:00:00.000

Рисунок 6. *Служебная таблица UniqueLabel (составлено автором)*

Кроме того, на основе рекомендаций было написано и модифицировано представление VisibleLabels, которое обеспечивает получение набора меток, доступных текущему пользователю. Главное отличие данного представления от оригинального заключается в использовании в качестве параметра определения принадлежности пользователя к роли специальной функции getRoleName (RoleID), где RoleID - идентификатор роли.

Функция getRoleName (RoleID) имеет следующую структуру:

create function getRoleName (@ID int) returns varchar(max)

as

begin

return (select name from sysusers,Marking where uid=@ID and Marking.Role_ID=@ID)

```
end
```

Представление имеет вид:

```
create view VisibleLabels
as
select ID, Label
from UniqueLabel with (nolock)
where
  ID in --Classification
  (select UniqueLabelID from UniqueLabelMarking with (nolock)
  where CategoryID = 1 and IS_MEMBER(dbo.getRoleName(RoleID)) = 1)
and --Compartments
  1 = ALL(select IS_MEMBER(dbo.getRoleName(RoleID)) FROM UniqueLabelMarking
  where CategoryID = 2 and UniqueLabelID = UniqueLabel.ID)
```

Результат работы представления, когда к нему обращаются пользователи с разными уровнями доступа, выглядит следующим образом:

Для пользователя UserS с уровнем доступа «Секретно».

	ID	Label
1	2	L2
2	3	L3
3	4	L4

Рисунок 7. *Результат представления VisibleLabels для Пользователя UserS (составлено автором)*

Для пользователя UserC с уровнем доступа «Конфиденциально».

	ID	Label
1	3	L3
2	4	L4

Рисунок 8. *Результат представления VisibleLabels для Пользователя UserC (составлено автором)*

Мандатная модель Белла-ЛаПадула. Под ***мандатным механизмом управления доступом*** понимается способ обработки запросов, основанный на формальном сравнении меток безопасности субъектов и объектов доступа в соответствии с заданными правилами.

Мандатная модель Белла-ЛаПадулы основана на правилах секретного документооборота, которые приняты в государственных и правительственных учреждениях большинства стран. Согласно этим правилам всем участникам процесса обработки критичной информации и документам, в которых она содержится, присваивается специальная метка, которая называется уровнем безопасности. Все уровни безопасности упорядочиваются с помощью установленного отношения доминирования [5, 9].

Для настройки модели Белла-ЛаПадулы были написаны функции получения идентификатора роли по её имени, функция получения ранга роли по её идентификатору, а также функция получения уровня безопасности текущего пользователя по номеру категории. Листинг данных программных объектов представлен ниже:

1. Функция получения идентификатора роли по её имени.

```
create function getRoleID (@Rolename varchar(max)) returns int
as
begin
    return (select uid from sysusers where name=@Rolename)
end
```

2. Функция получения ранга роли по её идентификатору.

```
create function getRoleMark (@RoleID int) returns int
as
begin
    return (select Mark from Marking where Role_ID=@RoleID)
end
```

3. Функция получения уровня безопасности текущего пользователя по номеру категории.

```
create function getCurrentMark (@Category_ID int) returns varchar(max)
as
begin
return (select name from sysusers, Marking where Marking.Role_ID = sysusers.uid and
Marking.CategoryID = @Category_ID
intersect select name from Sys.Database_Principals,sys.Database_Role_Members where
principal_id=role_principal_id
and member_principal_id=(select principal_id from Sys.Database_Principals where
name=SESSION_USER))
end
```

Далее при помощи комбинации данных процедур и функций были написаны представление и триггеры для работы с пользовательскими таблицами. Сначала при помощи процедуры addlabelcolumn к пользовательской таблице добавляется меточный столбец, в котором администратор безопасности задает метки безопасности для каждой строки.

```
create procedure addlabelcolumn (@tablename nvarchar (30), @labelcolumnname nvarchar (30))
as
begin
declare @s nvarchar(max)
set @s = 'alter table ' + @tablename + ' add ' + @labelcolumnname + ' int NULL'
execute sp_executesql @s
end
```

После этого, при вызове представления или триггера метка безопасности той или иной строки сравнивается с меткой безопасности роли, к которой относится текущий пользователь. В зависимости от результатов сравнения триггер либо выполняется, либо происходит откат транзакции, приведшей к запуску триггера, т.к. уровень доступа не позволяет его выполнить.

В процессе работы с базой данных пользователь обрабатывает таблицу через разработанное представление. При создании представления имя базовой таблицы изменяется, а представлению присваивается прежнее имя защищаемой таблицы. В результате пользователь не чувствует изменений в структуре БД и по-прежнему может обращаться с запросами к таблице, используя известное ему имя таблицы. При этом оригинальная таблица становится защищенной от несанкционированных изменений. Разработанные представление и триггеры в общем виде представлены ниже:

1. Представление для просмотра таблицы.

```
create view @tablename
as
select * from @table where (@table.labels >= dbo.getRoleMark(dbo.getRoleID(dbo.getCurrentMark(1))))
```

2. Триггер на добавление новых строк.

```
create trigger adding
on watching
INSTEAD OF insert
as
insert into @table
select @tablecolumn1, @tablecolumn2, @tablecolumn3, dbo.getRoleMark(dbo.getRoleID(dbo.getCurrentMark(1)))
from inserted
```

3. Триггер на обновление / удаление строк в таблице.

```
alter trigger updatingdelet
on @table
after update, delete
as
begin
declare @roleMark int = dbo.getRoleMark(dbo.getRoleID(dbo.getCurrentMark(1)))
print @roleMark
select * from deleted
if exists (select * from deleted where deleted.labels > @roleMark)
rollback
end
```

После создания всех процедур и функций полученный код был скомпонован и представлен в виде сценария установки в любую базу данных под управлением MS SQL Server. Администратору безопасности следует выполнить данных сценарий, чтобы в БД оказались автоматически встроены описанные ранее служебные объекты. Две разработанные модели управления доступом были сохранены как хранимые процедуры - MsLabelSecurity.sql и BellaLaPadula.sql, при запуске которых в базу данных устанавливаются служебные таблицы, процедуры, функции и триггеры, при помощи которых администратор безопасности сможет осуществить настройку разграничения доступа к базе данных [3].

Выводы

Таким образом, была проанализирована используемая в настоящее время подсистема управления доступом пользователей к объектам баз данных в СУБД Microsoft SQL Server. При разработке программной надстройки были учтены и исправлены недостатки существующих рекомендаций компании Microsoft по реализации многоуровневой модели доступа. Разработан механизм разграничения доступа к базам данных под управлением MS SQL Server, который может использоваться в качестве учебного материала, а также может быть внедрен в организациях, работающих с базами данных в MS SQL Server.

ЛИТЕРАТУРА

1. Ададуров С.Е., Диасамидзе С.В., Корниенко А.А., Сидак А.А. Международная кибербезопасность на железнодорожном транспорте: методологические подходы и нормативная методическая база // Вестник ВНИИЖТ. - 2015, № 6. С. 9-15.
2. Глухарев М.Л. Методы и механизмы обеспечения информационной безопасности в СУБД «Microsoft SQL Server»: учеб. пособие по дисциплине «Безопасность систем баз данных» / М.Л. Глухарев. - СПб.: Петербургский государственный университет путей сообщения, 2010. - 46 с.
3. Глухарев М.Л., Исаева М.Ф. Расширение механизма управления доступом в базах данных под управлением СУБД Microsoft SQL Server, Интеллектуальные системы на транспорте, 2014, с. 485-491.
4. Корниенко А.А. и др. Информационная безопасность и защита информации на железнодорожном транспорте: учебник. В 2 ч. / Корниенко А.А. и др.: Под ред. А.А. Корниенко, ч.2 Программно-аппаратные средства обеспечения информационной безопасности на железнодорожном транспорте - М.: ФГБОУ «Учебно-методический центр по образованию на железнодорожном транспорте», 2014. - 448 с.
5. Корниенко А.А., Глухарев М.Л. Формальная верификация ограничений целостности и триггеров реляционных баз данных при наличии избыточных требований целостности // Известия Петербургского университета путей сообщения. - СПб: ПГУПС, 2012. - №2 (31). - с. 112 - 115.
6. Корниенко А.А., Глухов А.П. Система обеспечения информационной безопасности на железнодорожном транспорте: современное состояние и задачи совершенствования, 2013, СПб: СПОИСУ Информационная безопасность регионов России (ИБРР-2013). VIII Санкт-Петербургская межрегиональная конференция. Санкт-Петербург, 23-25 октября 2013 г.: Материалы конференции - с. 105-106.
7. Кульба В.В., Курочка Н.П. Математическая модель обеспечения безопасности информации в базах данных // Интернет-журнал «НАУКОВЕДЕНИЕ» Том 7, №2 (2015) http://naukovedenie.ru/PDF/108TVN315.pdf (доступ свободный). Загл. с экрана. Яз. рус., англ. DOI: 10.15862/108TVN315.
8. Мамиконов А.Г., Кульба В.В., Косяченко С.А., Ужастов И.А. Оптимизация структур распределенных баз данных в АСУ. М.: Наука, 1990. - 235 с.
9. Хомоненко А.Д., Цыганков В.М., Мальцев М.Г. - Базы данных. Учебник для высших учебных заведений (6-е изд.) - СПб: КОРОНА-Век, 2009 - 734 с.
10. Шаньгин В.Ф. Защита компьютерной информации: Эффективные методы и средства. - М.: ДМК Пресс, 2010. - 544 с.

Isaeva Mariia Feliksovna
Emperor Alexander I St. Petersburg state transport university, Russia, Saint-Petersburg
E-mail: isaeva.mf@gmail.com

Gluharev Michael Leonidovich
Emperor Alexander I St. Petersburg state transport university, Russia, Saint-Petersburg
E-mail: mlgluharev@yandex.ru

Vetlugin Konstantin Alexandrovich
Emperor Alexander I St. Petersburg state transport university, Russia, Saint-Petersburg
E-mail: k.a.vetlugin@yandex.ru

Realization of the multilevel-based model of access distinction in databases under the management of the database control system Microsoft SQL Server

Abstract. For the convenience of storing and processing of the large amount of information databases are used, which are under the control of specialized software - database control systems. This article describes and analyzes the currently used user access control subsystem. In addition, the work demonstrates the development of software add-on, which is a mechanism for delineating access in databases, which can be used to customize work with user databases. With the help of this add-on, the security administrator can choose which access control policy to use in his work - a label or mandate access policy. During the developing of the software add-on, it was taken into account and corrected the shortcomings of Microsoft's existing recommendations for implementing a multilevel access model, and it is proposed to configure security at the row level using representations and security labels. In addition, the authors presented the developed mandate model of Bell-LaPadula. The mandate model is based on the rules according to which all participants of the processing of process the critical information and the documents, in which this information is contained. are assigned a special label, which is called the security level. The developed mechanism can be used as educational material, and it can be implemented in organizations working with databases in MS SQL Server.

Keywords: databases; access control; access restriction; relational database; database control system; level-based access model; marking policy; mandate model