С.Н. Ардатский
О.С. Бартунов
С.Н. Назин
М.Е. Прохоров
Е.Б. Родичев

# УПРАВЛЕНИЕ ДОСТУПОМ В СЛОЖНЫХ ИНФОРМАЦИОННЫХ СИСТЕМАХ НА ОСНОВЕ РОЛЕВОЙ АВТОРИЗАЦИИ

## Введение

Характерной особенностью современных информационных систем является необходимость сочетания богатого информационного наполнения с передовыми технологическими разработками. Это обусловлено тем, что наряду с новыми данными и документами большое значение может иметь наличие особых методов обработки информации, доступа к хранилищу документов, структурных и семантических связей с остальными компонентами информационной системы.

Одним из необходимых условий функционирования такой системы является задача разграничения доступа к ресурсам системы, т.е. поддержание требуемого уровня конфиденциальности, целостности и доступности данных. Отметим, что под ресурсами системы понимаются не только сама информация, но и свободное место на дисках, процессор и т.д.

Спецификой таких систем в науке и образовании является большое количество людей, непосредственно вовлеченных в подготовку содержания (авторы, пользователи системы), и не являющихся сотрудниками владельца системы (институт, управление), которые также принимают участие в подготовке материалов, например редакторская служба. Более того, пользователями системы могут быть другие информационные системы, с которыми осуществляется обмен информацией. При этом обычной практикой в настоящее время является использование web-интерфейсов для доступа к системе, что дает много преимуществ в виде универсальности, возможности удаленной работы, интерактивности.

## Классические модели разграничения доступа к ресурсам

В самом общем виде доступ к ресурсам можно представить в виде матрицы, задающей соотношение субъектов и объектов. К субъектам относятся авторизованные пользователи системы, причем один физический пользователь может соответствовать нескольким субъектам в контексте разграничения доступа (например, если он вошел в систему несколько раз с различными правами или запустил программу от имени одного из виртуальных пользователей). Объектами являются всевозможные ресурсы системы: документы, базы данных, рубрикаторы, файлы, внешние устройства и т.д. К объектам могут относиться и сложные структуры, описывающие пользователей системы, их группы, организации и т.п. В матрице доступа каждому субъекту соответствует строка, а объекту — столбец. На их пересечении указываются права на действия (атрибуты доступа), которые субъект имеет право производить над объектом. (Конечно, для каждого типа объектов набор допустимых действий свой, например, для документов — чтение, запись, модификация, создание, уничтожение, речь идет о подмножестве полного набора действий, права на выполнение которые с данным объектом имеет данный субъект.)

Однако прямой путь реализации полной матрицы доступа в реальных системах неприемлем: в этих системах функционируют сотни и тысячи субъектов, выполняя различные действия над десятками и сотнями тысяч объектов. Полное число элементов типичной матрицы доступа будет составлять миллиарды, что совершенно неприемлемо как с точки зрения производительности, так и с точки зрения поддержания ее целостности и актуальности.

Одним из путей решения указанной проблемы является реализация элементов матрицы доступа по мере надобности на основе атрибутов, приписываемых объектам и субъектам по правилам, определяемым так называемыми «политиками доступа».

Классическими политиками контроля доступа, описанными в «Оранжевой книге» Министерства обороны США и широко используемыми во многих операционных системах, являются ман-

датное управление доступом (MAC) и дискреционное управление доступом (DAC). Принципы, практика использования, преимущества и недостатки этих классических систем управления доступом подробно описаны в [2]. Следует заметить, что прямое применение данных методов не совсем годится для современных информационных систем.

Мандатное или принудительное управление доступом ориентировано в первую очередь на сохранение секретности данных, содержащихся в системе. В самом простом виде этот подход сводится к разделению пользователей (субъектов) и документов (объектов) на несколько уровней секретности (например: открытые материалы — для служебного пользования — секретно — совершенно секретно). Пользователь с заданным уровнем доступа может читать все документы своего или более низких уровней секретности, а на запись ему доступны документы своего и более высоких уровней (при этом после записи информации в документ более высокого уровня она становится недоступной писавшему). Данная система с некоторыми модификациями применяется в военных и государственных системах, но является недостаточно гибкой.

Дискреционное или произвольное управление доступом (DAC) основывается на понятиях владельца объекта и списка доступа к объекту. Для получения соответствующих прав для действий над объектом пользователь должен быть указан в списке доступа (ACL) к этому объекту либо персонально, либо в составе группы пользователей, для которой разрешены соответствующие действия над объектом. При отсутствии пользователя или его группы в списке при доступе к объекту используются права, применяемые по умолчанию. Система доступа, близкая к описанной, реализована в файловых системах Windows 2000 и Windows XP, более ограниченная версия встречается в файловых системах Unix, там список доступа ограничен одним пользователем (владельцем файла) и одной группой. Поддержание полномасштабных списков доступа снижает производительность DAC-систем по мере увеличения числа пользователей, а урезанные списки доступа не обеспечивают необходимую гибкость данной схемы.

Одним из недостатков DAC-систем (по сравнению с MAC) является отсутствие средств слежения за передачей информации. Средства произвольного управления доступом не могут помешать авторизованному пользователю законным образом получить секретную информацию и затем сделать ее доступной для других, неавторизованных пользователей.

## Недостатки применения классических моделей разграничения доступа в информационных системах

Использование классических средств разграничения доступа, реализованных в рамках ОС и СУБД, оказывается совершенно недостаточным для эффективного и надежного функционирования информационных систем. Приведем только два фактора, которые объясняют причины указанных явлений.

Во-первых, информационные системы, с точки зрения операционной системы, являются однопользовательскими, т.е. все процессы работают от лица одного, как правило, непривилегированного, пользователя, который ни в коем случае не должен являться владельцем информационного хранилища. Этим достигается определенный уровень безопасности со стороны операционной системы, например гарантированное использование места на дисках, загрузка процессора. Хранилище информации зачастую является гетерогенным — комбинацией файлового хранилища для документов и базы данных для хранения метаданных системы, и располагается на разных серверах. Владельцем хранилища является специальный непривилегированный пользователь, который имеет право на модификацию.

Еще одним фактором, затрудняющим использование классическим схем контроля доступа, является наличие большого количества объектов в ИС, требующих ограничения доступа.

Например, для ведения каталога ресурсов требуется контролировать действия как над отдельными структурными частями каталога, так и над возможностью доступа к ресурсам, приписанных к разным веткам каталога. Как правило, работа над каталогом ведется редак-

торской группой, при этом каждый редактор обладает разной квалификацией, компетенцией и уровнем ответственности. Ограничение доступа здесь преследует несколько целей:

• конфиденциальность, редактор может работать только со своими рубриками;

• целостность, структура каталога может быть изменена только очень компетентными редактором.

Другой пример — регистрация автора в системе.

## Описание модели ролевого разграничения доступа

Начнем с описания терминов, используемых в RBAC. К ним относятся: *пользователи* (users), *объекты* (objects), *роли, привилегии, права* и *сессии*.

**Пользователи и объекты** здесь понимаются в обычном смысле, т.е. как авторизованные пользователи системы и как множество физических и логических объектов, доступ к которым регулируется с помощью RBAC.

**Привилегия** — это минимальное возможное атомарное действие *пользователя*, которое требует того или иного разграничения доступа к этому действию. В принципе возможно объединение нескольких таких атомарных действий в одну *привилегию*, но только при условии, что во всех возможных ситуациях будет требоваться именно такая комбинация простейших действий. Объекты обладают атрибутами, которые могут фигурировать в *правилах*, составляющих ту или иную роль.

**Роль** — это набор *правил* (*прав*), определяющих, какими *привилегиями* и над какими *объектами* будет обладать пользователь, которому присваивается данная роль. *Правила*, составляющие *роль*, могут быть разрешающими (разрешать выполнение какого-то действия над определенной группой объектов) или запрещающими (запрещать выполнение соответствующих действий).

**Права (правила)** — составная часть роли, определяет *привилегию*, подмножество *объектов*, обладающих данной *привилегией*, и разрешение или запрет на выполнение данного действия.

**Сессия** — подмножество активированных в течение некоторого интервала времени *ролей* данного *пользователя*. Установление *сессии* RBAC возможно только после регистрации *пользователя* в системе. Активация каких-либо *ролей* вне *сессии* невозможна. Окончание сессии может производиться явным образом самим *пользователем* или системой RBAC по окончании определенного интервала времени, отсутствии активности и т.д. Одновременно могут выполняться несколько *сессий* для одного и того же *пользователя*.

Система RBAC функционирует следующим образом.

Разработчиком приложений, выполняющих различные операции над объектами системы, совместно с администратором RBAC и конструкторами ролей RBAC составляется список *привилегий*.

Конструкторами ролей разрабатывается *библиотека* ролей данной системы.

Диспетчерами прав пользователей каждому *пользователю* системы статическим образом присваивается набор *ролей*.

Сразу после авторизации *пользователя* в системе для него создается *сессия* работы с RBAC. Новые *сессии* создаются явным образом по команде *пользователя* или неявным образом по окончании предыдущей *сессии*, если работа *пользователя* с системой продолжается.

При попытке пользователя выполнить какое-либо действие над объектом, контролируемым RBAC, подсистема RBAC выполняет следующие действия:

- проверяет, является ли данное действие допустимым для данного пользователя в рамках текущей сессии RBAC с учетом атрибутов объекта на момент выполнения обращения;
- если действие входит в одну из ролей, присвоенных данному пользователю, — оно выполняется;
- в противном случае пользователь получает сообщение о недопустимости данного действия над объектом.

Функционирование системы RBAC иллюстрирует следующая схема.

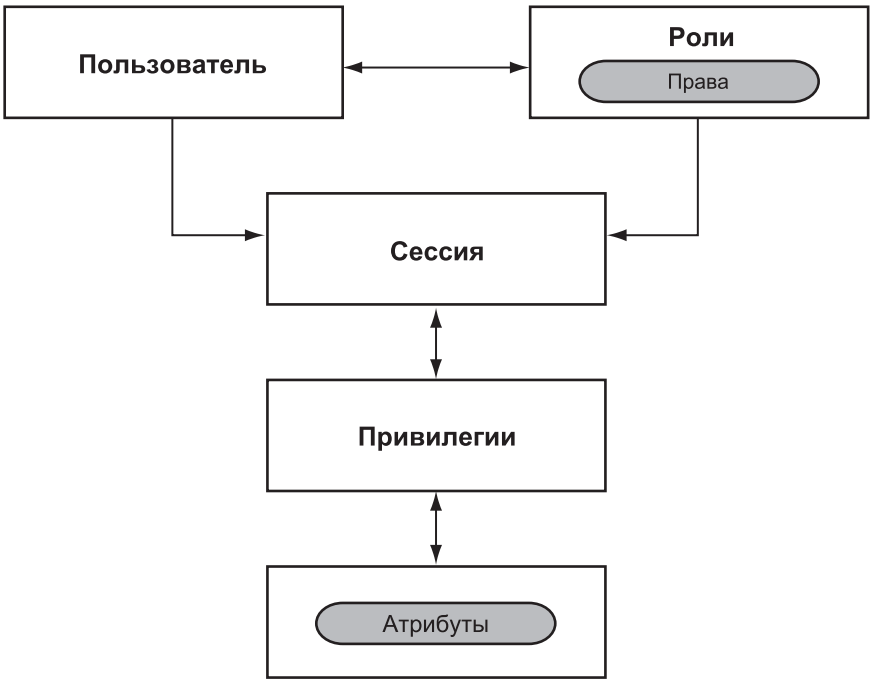


**Рис. 1.** *Схема функционирования системы RBAC*

Легко видеть, что в рамках системы ролей можно воспроизвести как мандатную, так и дискретную политику разграничения доступа к ресурсам.

Стандарт [4—6], регламентирующий терминологию, принципы и функциональность ролевой системы доступа, разработан Национальным институтом стандартов и технологий США и находится на стадии рассмотрения. Тем не менее он уже широко используется разработчиками информационных систем.

## Разделение ответственности в системе RBAC

Естественным образом система ролевого разграничения доступа позволяет распределять проблемы разделения ответственности между администраторами и привилегированными пользователями системы. Например, в правилах присвоения ролей может быть указано, что в системе может быть определен только один администратор верхнего уровня — в этом случае попытка присвоения данной роли второму субъекту завершится неудачей. Другой пример — может быть определено количество пользователей, играющих определенную роль. Данные ограничения производятся статическим образом, т.е. в момент присвоения роли пользователю, до того, как он может воспользоваться данными правами.

Самым важным примером статического разделения ответственности является запрет на совмещение определенных ролей, например, запрет совмещения функций конструктора ролей и диспетчера (распределителя) ролей. Аналогичный пример из системного администрирования операционных систем, оснащенных системой RBAC, — обязательное разделение функций системного администратора и администратора RBAC.

Система статического разделения ответственности дополняется более гибкой системой динамических ограничений. Например, в системе могут отсутствовать ограничения на число пользователей, которым присвоены те или иные административные роли, но может существовать ограничение на число одновременно функционирующих администраторов. Так, в систему может войти только один пользователь с правами главного администратора, редактора раздела документов и даже с правами на редактирование конкретного документа. Статические и динамические правила разделения ответственности могут сосуществовать друг с другом.

## Конфликты ролей в системе RBAC

Конфликты ролей в системе RBAC вызываются присвоением одному *пользователю ролей*, содержащих *правила* противополож-

ного знака относительно одних и тех же *привилегий*. Возможны следующие подходы к разрешению или недопущению подобных конфликтов.

**Недопущение конфликтов**

Конфликты ролей в системе RBAC становятся невозможны в принципе, если все роли строятся из правил одного знака: только разрешающих или только запрещающих. (Система ролей, построенная только из запрещающих правил внутренне не противоречива, но функционирование в ней пользователя выглядит совершенно неестественно.) В библиотеке ролей, составленной только из разрешающих правил, присвоение пользователю еще одной роли приводит только к расширению его прав доступа и не накладывает ограничений на уже имеющиеся.

**Разрешение конфликтов** возможно статическими и динамическими способами. Перечислим возможные пути их разрешения:

*статические*

• разделение пользователя на несколько субъектов (с независимыми логинами), с бесконфликтными наборами ролей;

• введение статической (на момент присвоения роли) процедуры отмены одного из конфликтующих правил;

*динамические*

• активация в рамках одной сессии только не противоречивого подмножества присвоенных данному пользователю ролей;

• переключение между сессиями — при активации роли, содержащей противоречие с одной из уже активированных, текущая сессия завершается, из нее исключается роль с противоречием, включается новая роль, и для получившегося набора ролей создается новая версия RBAC.

## Уровни систем с ролевым разграничением доступа

На данный момент существуют 4 различных уровня систем RBAC [4].

1. Базовый уровень (Core RBAC). В системах этого уровня:

• *пользователи* получают *права* через присвоенные им *роли*;

• поддерживается отношение многое-ко-многим между *пользователями* и *ролями*;

• поддерживается отношение многое-ко-многим между *правами* и *ролями*;

• выполняется активация *ролей* на *сессию*.

2. Иерархический RBAC. Данный вариант содержит базовый уровень RBAC плюс поддержку иерархической системы ролей.

3. RBAC с поддержкой статического разделения конфликтов. Состоит из базового уровня и системы разрешения конфликтов между ролями.

4. RBAC с поддержкой динамического разделения конфликтов: наиболее полная система, состоящая из RBAC базового уровня, системы поддержки иерархических ролей и системы динамического разрешения конфликтов между ролями.

**Примеры реализации систем с ролевым разграничением доступа**

Большинство подобных примеров относятся к операционным системам, в которых с помощью RBAC реализуются более гибкие правила доступа пользователей ОС к ресурсам системы (файлам и внешним устройствам), чем в рамках стандартной дискретной модели:

• элементы RBAC были введены с ОС Solaris начиная с версии 8.0;

• на основе RBAC был разработан защищенный серверный дистрибутив ОС Linux — ALT Linux «Castle».

Примеры применения системы ролевого разграничения доступа в общедоступных информационных системах встречаются гораздо реже. Базовая модель RBAC (Core RBAC) была разработана в рамках грантов РФФИ 99-07-90068 и 02-07-90222 на основе базы данных PostgreSQL. На ее основе были реализованы информационные серверы «Научной Сети» (www.nature.ru, phys.web.ru, students.web.ru, www.astronet.ru) и Федеральные образовательные порталы (www.humanities.edu.ru, www.en.edu.ru и www.ecsocman.edu.ru).

## Литература

[1] RBAC NIST http://csrc.nist.gov/rbac/.

[2] Sandhu R., Samarati P. Access Control: Principles and Practice, IEEE Communications. 1994. Vol. 32. N 9.

[3] Osborn S., Sandhu R., Munawer Q. Configuring Role-Based Access Control to Enforce Mandatory and Discretionary Access Control Policies // ACM Transactions on Information and Systems Security (TISSEC). 2000. Vol. 3. N 2.

[4] Proposed voluntary consensus standard for role based access control http://csrc.nist.gov/rbac/rbac-std-ncits.pdf.

[5] A Proposed Standard for Role Based Access Control (PDF) / D. Ferraiolo, R. Sandhu, S. Gavrila et al. // ACM Transactions on Information and System Security. 2001. Vol. 4. N 3. Draft of a consensus standard for RBAC http://csrc.nist.gov/rbac/rbacSTD-ACM.pdf.

[6] RBAC LIST (Laboratory for Information Security Technology, George Mason University) http://www.list.gmu.edu/.