Математические структуры и моделирование 2020. № 4(56). С. 126–135

УДК 004.056
DOI 10.24147/2222-8772.2020.4.126-135

О ПРЕДСТАВЛЕНИИ НЕКОТОРЫХ РОЛЕВЫХ МОДЕЛЕЙ РАЗГРАНИЧЕНИЯ ДОСТУПА ОБЪЕКТНО-ОРИЕНТИРОВАННОЙ МОДЕЛЬЮ HRU: НАСЛЕДОВАНИЕ ТАКСОНОМИЧЕСКИМ ЛИСТОВЫМ МЕТОДОМ

С.В. Усов
к.т.н., доцент, e-mail: raintower@mail.ru

Омский государственный университет им. Ф.М. Достоевского, Омск, Россия

Аннотация. Рассматривается возможность представления ролевых политик безопасности с помощью объектно-ориентированных дискреционных моделей разделения доступа. Рассмотрена ролевая модель с иерархией, в которой эксплуатируется таксономический листовой метод наследования. Полномочия ролевой модели представлены в виде наборов пар «объект-право доступа». Построена иерархия классов модели OOHRU, отражающая ролевую политику разграничения доступа. Описаны команды модели OOHRU, соответствующие переназначению ролей в исходной модели.

Ключевые слова: ролевые модели разграничения доступа, OOHRU, иерархия ролей.

Введение

Данная работа является прямым продолжением работы «О представлении некоторых ролевых моделей разграничения доступа объектно-ориентированной моделью HRU» [1], опубликованной двумя годами ранее. По этой причине позволим себе сократить как историческую справку, так и описание рассматриваемых моделей до минимума.

Рассматривается объектно-ориентированное развитие [4] дискреционной модели разграничения доступа, предложенной Харриссоном, Руззо и Ульманом [2]. Объектно-ориентированный подход позволил как расширить охват компьютерных систем, которые можно описать с помощью модели HRU, так и сделал исходную модель совместимой с рядом актуальных задач.

Напомним, что компьютерная система в OOHRU рассматривается в виде множества объектов $\mathbf{O}$, разбитых по множеству классов $\mathbf{K}$ (все объекты одного класса имеют одинаковый набор полей и методов), обладающих открытыми полями $f \in \mathbf{F}$ и скрытыми полями $p \in \mathbf{P}$, а также методами обработки полей $s \in \mathbf{S}$. Здесь $F = \bigcup_{k \in \mathbf{K}} k.\mathbf{F}$ — множество всевозможных открытых полей всех объектов и классов, $k.\mathbf{F}$ — множество открытых полей класса k (каждый объект класса k обладает тем же набором $k.\mathbf{F}$ открытых полей), аналогично

определяются $\mathbf{P}$ и $\mathbf{S}$. Причём если поле $k.f$ наследуется классом k у класса k', то соответствующее поле класса k' мы будем для удобства обозначать именно $k'.f$, подчёркивая тем самым их взаимосвязь (таким образом, $f \in k.\mathbf{F}$ и $f \in k'.\mathbf{F}$). Пусть $\mathbf{O}^k \in \mathbf{O}$ — множество объектов класса $k \in \mathbf{K}$. В случае, если требуется уточнить класс объекта, поле f объекта $o^k \in \mathbf{O}^k$ будем обозначать $o^k.f$, поле f класса k — $k.f$. Для скрытых полей класса будем использовать аналогичные обозначения.

1. Построения модели дискреционного разделения доступов

Для построения модели дискреционного разделения доступов для каждого объекта и для каждого класса вводится дополнительное скрытое поле M, содержащее локальную матрицу доступов, и методы работы с матрицей доступов. Строки матрицы доступов объекта o соответствуют объектам и классам системы, столбцы – полям и методам объекта o, а в ячейке, находящейся на пересечении строки, соответствующей объекту o', и столбца, соответствующего полю либо методу $x \in \mathbf{X} = \mathbf{F} \cup \mathbf{S}$, находится подмножество множества R прав доступа, определённых в системе, которое обозначается $o.M[o', x]$. Модификация матриц доступа производится посредством выполнения команд системы безопасности, о которых будет сказано ниже.

Модель безопасности OOHRU называется иерархической (или моделью с иерархией), если на множестве объектов $\mathbf{O}$ задан частичный порядок-иерархия и в любой момент работы системы для любых двух объектов $o, o' \in \mathbf{O}$ таких, что $o' \leqslant o$ для любого поля или метода $x \in \mathbf{X}$, общего для объектов o и o', и для любого поля или метода $x' \in \mathbf{X}$ объекта $o'' \in \mathbf{O}$ верно следующее: $o''.M[o, x'] \subset o''.M[o', x']$ и $o'.M[o'', x] \subset o.M[o'', x]$. Здесь и далее «$\leqslant$» – отношение частичного порядка, задающее иерархию.

Состояние системы в модели HRU изменяется под действием команд, которые состоят из условной части и последовательности элементарных операторов [2], которая выполняется, только если истинна условная часть. Список элементарных операторов в OOHRU включает [4]:

1. $Create(o^k, k)$ — создаёт объект o^k класса $k \in \mathbf{K}$, если $o^k \in \mathbf{O}$.
2. $Destroy(o^k)$ — уничтожает объект $o^k \in \mathbf{O}$.
3. $Enter(r, o^k, o'^{k'}.f)$ — вносит право доступа r в $o'^{k'}.M[o^k, o'^{k'}.f]$, где o^k — объект класса k, $o'^{k'}$ — объект класса k'.
4. $Delete(r, o^k, o'^{k'}.f)$ — удаляет право доступа r из $o'^{k'}.M[o^k, o'^{k'}.f]$.
5. $Grant(r, o^k, o'^{k'}.s)$ — разрешает вызов объектом o^k метода $o'^{k'}.s$.
6. $Deprive(r, o^k, o'^{k'}.s)$ — запрещает вызов объектом o^k метода $o'^{k'}.s$.

Изменения, производимые операторами, отражаются в матрицах доступа объектов системы. Подробное описание модели OOHRU можно найти в [4].

Также рассматриваются ролевые политики разграничения доступа [3].

Компьютерная система в рамках ролевой политики представляется совокупностью следующих множеств: множества пользователей U, множества ролей

Z, множества полномочий **Y** и множества сеансов работы пользователей с системой [3]. Обратим внимание, что использование нестандартных обозначений здесь вызвано необходимостью избежать конфликта с описанием модели HRU, где те же символы несут иную смысловую нагрузку по сравнению с классическими обозначениями сущностей, принятыми в ролевой модели.

Множество объектов системы не задаётся в явном виде, а представляется через множество полномочий **Y**: каждое полномочие заключает в себе отношение на декартовом произведении множества прав доступа и множества объектов системы, однако фактически каждое «элементарное» полномочие можно представить в виде пары $y = (x, r)$, где x отвечает объекту системы (или группе сходных объектов) в рамках субъектно-объектной парадигмы, а r – праву доступа к этому объекту (или группе объектов). В качестве r может также выступать право вызова, если x представляет собой активную сущность системы. Сформированное таким образом множество всех объектов системы обозначим через X, а множество всех прав доступа — через R.

Далее, пусть $z.y$ — обозначение элементарного полномочия y, относящегося к роли z (поскольку каждая роль задаётся именно набором полномочий), соответственно, $z.Y$ — полный набор полномочий роли z, $z.Y \subset Y$.

Управление доступом осуществляется на основе изменения множества активных сеансов системы. Каждому сеансу $c \in C$ сопоставляется пользователь $u_c \in U$, подмножество доступных пользователю в рамках данного сеанса ролей $Z_c \subset \mathbf{Z}$ и подмножество допустимых полномочий $Y_c \subset \mathbf{Y}$. Считается, что система функционирует безопасно, если пользователь u_c может осуществлять действия только в рамках полномочий из множества $Y_c = \bigcup_{z \in Z_c} z.Y$ во время сеанса $c \in C$.

На множестве ролей в реальных системах обычно возникает иерархия, индуцированная организационно-управленческими схемами организаций, в которых эксплуатируется система. Как правило, подчинённость ролей в иерархии включает наследование прав и полномочий, которое может быть направлено как «снизу», так и «сверху». При наследовании «сверху» подчинённый субъект наследует права родительских субъектов. Однако базовые ролевые модели эксплуатируют наследование «снизу». Здесь можно выделить три ключевых подхода к построению иерархии.

1. Строгий таксономический листовой подход. Всё множество полномочий разбивается на непересекающиеся подмножества, каждое из подмножеств приписывается листу дерева иерархии ролей. Роль в нелистовой вершине наделяется множеством полномочий, являющимся объединением множеств полномочий непосредственно подчинённых ролей (соответствие между ролями и полномочиями, таким образом, индуцируется полномочиями листовых вершин и устанавливается при движении по дереву ролей снизу вверх).

2. Нестрогий таксономический листовой подход. Единственное отличие от предыдущего подхода — отсутствие требования на запрет пересечения подмножеств полномочий, сопоставленных листовым вершинам.

3. Иерархически охватный подход. Граф такой иерархии ролей не обязан быть деревом (но, конечно, не может содержать сильных циклов). При таком

подходе считается, что если пользователю сопоставлена некоторая роль $r \in \mathbf{R}$, то ему также должны быть сопоставлены и все роли, подчинённые r. Что позволяет исключить из роли r полномочия, уже содержащиеся в подчинённых ей ролях.

В работе [4] была предложена иерархическая модель OOHRU, устройство которой подразумевает, что объект o, находящийся на более низком уровне иерархии, чем объект o', обладает меньшим набором прав (как в отношении доступа к другим объектам, так и в отношении ограничения доступа других объектов по отношению к себе) по сравнению с объектом o'. Такая структура напоминает решётку иерархии ролей, что позволяет предположить структурную близость данных моделей.

Целью данной работы является именно моделирование ряда ролевых политик безопасности средствами OOHRU. Связь между субъектно-объектной ролевой моделью без иерархии на множестве ролей и OOHRU, равно как и связь между субъектно-объектной ролевой моделью с иерархией с наследованием «сверху» на множестве ролей и OOHRU, были изучены в работе [1]. Здесь будет разобран случай ролевой модели с таксономическим листовым подходом к организации иерархии ролей.

2. Связь между субъектно-объектной ролевой моделью с иерархией на множестве ролей и OOHRU: случай таксономического листового подхода к наследованию

В этом разделе нас будет интересовать ролевая политика безопасности с таксономическим листовым подходом к наследованию.

Базовая ролевая политика разграничения доступа подразумевает статичность подсистемы безопасности в рамках одного сеанса, то есть полномочия неизменны, к каждой роли относится фиксированный набор полномочий, и каждому пользователю назначен неизменный в течение сеанса набор ролей. Таким образом, перераспределения прав доступа в смысле дискреционной политики безопасности не происходит. Построим иерархическую объектно-ориентированную модель HRU $\Sigma' = (O(t), M(t), K, R)$, соответствующую данной субъектно-объектной ролевой модели $\Sigma = (U, Z, Y, C)$.

Во-первых, сформируем множество R прав доступа модели Σ', вычленив права доступа r из элементарных полномочий, а также множество X объектов системы.

Во-вторых, зададим классы объектов, а также вспомогательные класс пользователей и класс сеансов: в реализующей ролевую политику с иерархией модели OOHRU они задаются так же, как и в случае реализации ролевой политики без иерархии на множестве ролей [1].

В частности, пассивной сущности $x \in X$ субъектно-объектной модели будет соответствовать объект o, имеющий поле f, содержащее информацию объекта x, и приватное поле M — матрицу доступов. Активной сущности x' субъектно-объектной модели будет соответствовать объект o', содержащий метод s, вы-

полняющий активные функции субъекта x', и приватное поле M — матрицу доступов. Объекты будут в дальнейшем разбиты по классам исходя из их структуры и возможностей доступа к ним. Каждой роли $z \in Z$ также будет сопоставлен класс $k_z \in K$ в модели OOHRU.

Заполнение матриц доступов объектов происходит следующим образом: если роль $z \in Z$ обладает элементарным полномочием $y = (x, r) = (o.f, r)$, где x — объект субъектно-объектной модели, объектно-ориентированной интерпретацией которого является объект o, то в матрицу доступов объекта o в ячейку на пересечении строки поля f и столбца класса k_z заносится право доступа r: $o.M[k_z, f] := o.M[k_z, f] \cup r$. Если же роль $z \in Z$ обладает элементарным полномочием $y = (x', r) = (o'.s, r)$, где x' — субъект субъектно-объектной модели, объектно-ориентированной интерпретацией которого является объект o', а r — право вызова процедуры, то в матрицу доступов объекта o в ячейку на пересечении строки метода f и столбца класса k_z заносится 1: $o.M[k_z, f] := 1$.

Осталось построить лишь иерархию классов ролей.

Иерархия классов ролей в OOHRU будет повторять иерархию ролей в исходной ролевой модели, только в отражённом виде. Каждой роли z будет сопоставлен класс k_z, кроме того, введём класс k_0, отвечающий пустому множеству ролей. Его наследниками будут классы, соответствующие листовым вершинам в дереве ролей. Далее, если роль z является непосредственным потомком роли z' в таксономической модели (при этом набор полномочий $z.Y$ роли z будет содержаться в наборе полномочий $z'.Y$ роли z'), то класс $k_{z'}$ будет непосредственным потомком класса k_z в OOHRU.

В дальнейшем мы будем называть такую иерархию в OOHRU таксономической иерархией ролей.

Если пользователь $u \in U$ при авторизации на сеанс $c \in C$ получает роль z, то в модели Σ' в классе k_z создаётся новый объект o_u. Каждый такой объект содержит в качестве private-поля собственную матрицу доступов (заполняется при создании объекта на основе совокупности всех элементарных полномочий, которыми обладает пользователь u во время сеанса c), поле идентификатора и методы, необходимые для работы с объектами системы. Так, например, каждому полномочию может соответствовать свой метод.

Дополним OOHRU командой старта сеанса с пользователем u с авторизацией его на роль z и командой завершения сеанса:

$CommandStartSession\_z(o_u : k_z)$
  $Create(o_u, k_z)$.

Команда завершения сеанса этим же пользователем:

$CommandEndSession\_z(o_u : k_z)$
  $Destroy(o_u)$.

Тот факт, что перед стартом сеанса выполняется проверка возможности пользователю u назначить роль z, может быть также отражён исключительно средствами модели OOHRU. Для этого вне иерархии введём два дополнительных класса: класс сеансов k_C и класс пользователей k_U. Для каждого сеанса c в класс k_C помещается объект o_c, содержащий поле-идентификатор сеанса,

метод $user$ инициализации пользователя и матрицу доступов. Для каждого пользователя u в класс k_U помещается объект o_u^*, содержащий метод $user$ и матрицу доступов. Аналогичный метод $user$ будут содержать все классы и объекты естественной иерархии подмножеств ролей. Роль этого метода формальна, нам потребуется лишь соответствующая ему ячейка в матрице доступов содержащего его объекта (класса).

Положим, что сеанс c подразумевает назначение пользователю u роли z. В этом случае при создании объекта o_c происходит модификация матриц доступа объекта o_u^* и класса k_z следующей командой:

$CommandCreateSession\_z(o_c : k_C; o_u^* : k_U; k_z)$
$Create(o_c, k_C)$,
$Grant(o_c, o_u^*.user)$,
$Grant(o_c, k_z.user)$.

При выполнении этой команды матрицы доступов модифицируются следующим образом: $o_u^*.M[o_c, user] = 1$, $k_z.M[o_c, user] = 1$ (объект o_c получает право запускать метод $user$ объекта o_u^*, относящегося к пользователю u, и аналогичный метод класса k_z). Команда может быть выполнена, только если пользователь u зарегистрирован в системе, при регистрации был создан соответствующий ему объект o_u^*.

Теперь в команду сеанса мы можем включить проверку возможности пользователю u назначить роль z:

$CommandStartSession\_z(o_c : k_C; o_u^* : k_U; o_u : k_z)$
If
$o_u^*.M[o_c, p_u] = 1$ and $kz.M[o_c, p_u] = 1$
then
$Create(o_u, k_z)$.

Существенное отличие от случая ролевой модели без иерархии заключается в том, что каждый пользовательский объект в индуцированной иерархии ролей соответствует одной роли из числа назначенных субъекту. В информационных системах, эксплуатирующих таксономический листовой подход к построению иерархии ролей, можно выделить два ключевых случая сочетания назначенных пользователю ролей.

Первый случай подразумевает назначение пользователю некоторого множества ролей из числа листовых (на прочие роли пользователь авторизован быть не может). В такой ситуации достаточно построить естественную иерархию подмножеств листовых ролей, игнорируя таксономическую листовую иерархию либо дополняя её. Фактически работа со множеством листовых ролей сводит данный случай к случаю ролевой модели, свободной от иерархии. Работа с созданием и удалением пользовательских объектов, переназначение ролей и полномочий происходит точно так же, как и в случае модели, свободной от иерархии [1].

Второй случай подразумевает назначение пользователю одной роли (не обязательно листовой), реже нескольких, из иерархии. В такой ситуации работа с созданием и удалением пользовательских объектов, переназначение ролей и

полномочий происходит аналогично случаю иерархии с наследованием «сверху».

Например, если необходимо назначить пользователю u роли z и z' в рамках уже построенной системы безопасности, достаточно создать при назначении пользователя на эти роли два отвечающих пользователю u объекта как в классе k_z, так и в классе $k_{z'}$:

$CommandStartSession\_\{z, z'\}(o_u : k_z, o'_u : k_{z'})$
  $Create(o_u, k_z)$,
  $Create(o'_u, k_{z'})$.
$CommandEndSession\_\{z, z'\}(o_u : k_z, o'_u : k_{z'})$
  $Destroy(o_u)$,
  $Destroy(o'_u)$.

Аналогичные команды можно использовать для произвольного набора ролей α.

Как и в случае ролевой модели, свободной от иерархии, для смены набора назначенных ролей α на набор β пользователю необходимо завершить текущий сеанс и начать новый, в котором ему будет назначен требуемый набор ролей.

Рассмотрим, как может осуществляться модификация множества полномочий, назначенных роли z.

1) Только листовые роли могут быть назначены пользователю. Ролевая модель реализована с помощью естественной иерархии модели OOHRU. В этом случае наблюдается полное соответствие случаю ролевой модели без иерархии, случай которой был изучен в работе [1].

2) Только листовые роли могут быть назначены пользователю. При этом естественная иерархия в модели OOHRU не построена. В этом случае под каждую роль z, назначенную пользователю u, был создан отдельный объект o_z класса k_z. Модификация каждой отдельной роли осуществляется независимо с помощью команд вида:

$CommandAddPermission\_z\_r\_f(k_z, o : k_o)$
  $Enter(r, k_z, o.f)$.

Данная команда не нарушает наследование прав доступа в иерархии, поскольку в OOHRU оператор $Enter$ может быть выполнен, только если соблюдены так называемые условия целостности:

$$(r \in o.M[k_{z_1}, f])\& \ldots \&(r \in o.M[k_{z_s}, f])[4].$$

При необходимости добавить более широкий набор полномочий, расширяем список элементарных операторов $Enter$ в команде, добавляющих требуемые права доступа. Если элементарное полномочие содержит право вызова метода s объекта o некоторого класса k_o, используем команду несколько иного вида:

$CommandAddPermission\_z\_s(k_z, o : k_o)$
  $Grant(k_z, o.s)$.

Удаление элементарного полномочия осуществляется аналогичным способом, только условия целостности будут наложены уже на родительские классы:

$CommandDeletePermission\_z\_r\_f(k_z, o : k_o)$

$Delete(r, k_z, o.f)$.

Отметим, что данный подход применим и в том случае, когда возможно назначение пользователю не только листовых ролей. Важно лишь, чтобы для каждой роли пользователя создавался отдельный пользовательский объект. В не подпадающих под это правило ситуациях необходимо воспользоваться рецептом, предложенным в одном из трёх оставшихся случаев.

3) Случай таксономически строгого подхода. Добавим в систему дополнительно роли z_y для каждого элементарного полномочия $y = (x, r)$, где $x \in X$ — объект, $r \in R$ – допустимое право доступа к этому объекту. Если $y \in z.Y$, то соответствующий этой роли класс k_y будет являться непосредственным наследником класса k_0 и непосредственным родителем класса k_z. При этом $r \in o.M[k_y, o.f]$ либо $o.M[k_y, o.p] = 1$, в зависимости от того, представляет ли элементарное полномочие y возможность доступа r к полю f объектно-ориентированной интерпретации o объекта x или же возможность вызова метода p объектно-ориентированной интерпретации o субъекта x. Далее рассмотрим случай с полем объекта (случай с методом субъекта аналогичен).

Во-первых, согласно условиям целостности все потомки класса k_y (то есть классы, соответствующие ролям, содержащим полномочие y) будут обладать правом r в соответствующей ячейке права доступа. При необходимости отозвать полномочие y у роли z (а значит и у всех ролей, ранее его содержавших, поскольку мы работаем в рамках таксономического листового подхода), необходимо применить команды вида:

$CommandDeletePermission\_\alpha\_r\_f(k_\alpha, o : k_o)$

$Delete(r, k_\alpha, o.f)$

ко всем классам таксономической иерархии, начиная с классов, содержащих (по включению) наимощнейшие наборы полномочий (классы, соответствующие самым верхним ролям или наборам ролей α в иерархии ролей), и заканчивая классами листовых ролей и далее классами листовых полномочий (на самом деле, среди последних удаление права пройдёт успешно только для класса k_y, поскольку остальные классы полномочий изначально не содержали этого права в соответствующей ячейке матрицы доступов). При таком просмотре графа ролей, «сверху вниз», условия целостности будут выполнены на всех этапах.

Для возвращения отозванного полномочия применяем команды вида

$CommandAddPermission\_\alpha\_r\_f(k_\alpha, o : k_o)$

$Enter(r, k_\alpha, o.f)$,

на этот раз начиная с класса полномочия y и классов листовых ролей, при этом двигаясь в обратном направлении, «снизу вверх» по таксономическому графу иерархии ролей.

4) Случай нестрогого таксономического подхода. В этом случае мы не можем воспользоваться конструкцией из предыдущего случая, поскольку одно и то же полномочие может принадлежать сразу нескольким листовым ролям, а удалить его нам может понадобиться только из одной листовой роли. Будем использовать команды того же вида, что и в строгом случае (разве что не станем

создавать классы полномочий), но несколько изменим порядок их применения.

Алгоритм отзыва полномочия y у роли z (все описанные ниже операции осуществляются инструментами модели OOHRU, описанными в случае 3 и ранее).

Шаг 1. Двигаясь (здесь и далее по графу дерева ролей при обходе «в ширину») сверху вниз, удаляем полномочие y у всех ролей.

Шаг 2. Добавляем полномочие y во все листовые роли, кроме роли z из числа тех, которым оно принадлежало ранее.

Шаг 3. Двигаясь снизу вверх, пытаемся добавить полномочие y во все обойдённые вершины. Благодаря условиям целостности мы сможем добавить его лишь тем ролям, которые «наследовали» бы его согласно устройству таксономической листовой иерархии ролей.

Алгоритм добавления полномочия y роли z гораздо проще: добавляем y листовой роли z средствами OOHRU и применяем Шаг 3 предыдущего алгоритма. Полномочие будет добавлено, куда положено, благодаря условиям целостности.

Тем самым доказана

Теорема 1. *Для любой субъектно-объектной иерархической ролевой модели с таксономическим листовым подходом к наследованию существует реализующая ее иерархическая модель OOHRU.*

Литература

1. Усов С.В. О представлении некоторых ролевых моделей разграничения доступа объектно-ориентированной моделью HRU // Математические структуры и моделирование. Омск : Ом. гос. ун-т, 2018. № 48. С. 127–137.
2. Harrison M.A., Ruzzo W.L., Ulman J.D. Protection in Operating Systems // Communications of the ACM. 1975. P. 14–25.
3. Ferraiolo D.F., Kuhn D.R. Role-Based Access Control // 15th National Computer Security Conference. 1992. P. 554–563.
4. Усов С.В. Об отношении между дискреционными моделями объектно-ориентированных и субъектно-объектных компьютерных систем // Проблемы информационной безопасности. Компьютерные системы. 2013. Т. 3. С. 18–26.

ON THE REPRESENTATION OF ROLE-BASED ACCESS CONTROL MODELS BY OBJECT-ORIENTED HRU MODEL: TAXONOMIC LEAF-BASED HIERARCHY CASE

S.V. Usov
Ph.D.(Eng.), Associate Professor, e-mail: raintower@mail.ru

Dostoevsky Omsk State University, Omsk, Russia

Abstract. In this paper the possibility of representing of some types of role-based access control models by object-oriented discretionary access control model is considered. The role-based security models with taxonomic hierarchy are considered. The permissions of the role-based access control model are represented as a set of pairs of object and access right. A hierarchy of classes of the object-oriented HRU model, based on the role-based access control policy, is constructed. Commands of the object-oriented HRU model, corresponding to the reassignment of roles in the original role-based model, are described.

Keywords: role-based access control model, object-oriented discretionary access control model, role hierarchy.

References

1. Usov S.V. O predstavlenii nekotorykh rolevykh modelei razgranicheniya dostupa ob"ektno-orientirovannoi model'yu HRU. Matematicheskie struktury i modelirovanie, Omsk, Om. gos. un-t, 2018, no. 48, pp. 127–137. (in Russian)
2. Harrison M.A., Ruzzo W.L., and Ulman J.D. Protection in Operating Systems. Communications of the ACM, 1975, pp. 14–25.
3. Ferraiolo D.F. and Kuhn D.R. Role-Based Access Control. 15th National Computer Security Conference, 1992, pp. 554–563.
4. Usov S.V. Ob otnoshenii mezhdu diskretsionnymi modelyami ob"ektno-orientirovannykh i sub"ektno-ob"ektnykh komp'yuternykh sistem. Problemy informatsionnoi bezopasnosti. Komp'yuternye sistemy, 2013, vol. 3, pp. 18–26. (in Russian)

Дата поступления в редакцию: 30.10.20