*Математические структуры и моделирование 2018. № 4(48). С. 128–138*

УДК 004.056
DOI: 10.25513/2222-8772.2018.4.128-138

# О ПРЕДСТАВЛЕНИИ НЕКОТОРЫХ РОЛЕВЫХ МОДЕЛЕЙ РАЗГРАНИЧЕНИЯ ДОСТУПА ОБЪЕКТНО-ОРИЕНТИРОВАННОЙ МОДЕЛЬЮ HRU

**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. Рассмотрены ролевая модель без иерархии, а также ролевая модель с иерархией с наследованием «сверху». 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

Дискреционные политики безопасности известны ещё с 70-х годов прошлого столетия и традиционно базируются на субъектно-объектной парадигме компьютерной системы. Однако в связи с возрастающей актуальностью объектно-ориентированного подхода к построению компьютерных систем возникает необходимость в пересмотре классических политик безопасности. Так, например, в [1] была предложена объектно-ориентированная модель разграничения доступа, базирующаяся на модели HRU (Харриссона–Руззо–Ульмана) [2], однако обладающая более широкими возможностями, в частности, в рамках охвата компьютерных систем, которые можно описать с помощью этой модели.

Ролевые политики разграничения доступа стали широко известны только в 90-х годах после выхода работ Феррайоло и Куна [3]. Они отличаются от списков контроля доступа (ACL), используемых в системах управления доступом, основанных на дискреционных моделях безопасности, тем, что позволяют назначать на сложные операции с составными данными, а не только на атомарные операции с низкоуровневыми объектами данных.

Концепции иерархии ролей и ограничений позволяют создать или смоделировать контроль доступа на основе решетки доступов. RBAC широко используется для управления пользовательскими привилегиями в пределах единой системы или приложения. Список таких систем включает в себя Microsoft

Active Directory, FreeBSD, Solaris, СУБД Oracle, PostgreSQL 8.1 и множество других. В работах Рави Санду (Ravi Sandhu) [4] было показано, что технология управления доступом на основе ролей обладает достаточной гибкостью для моделирования как дискреционных, так и мандатных политик безопасности. Однако в данных работах моделируются лишь частные дискреционные политики, близкие по структуре к Take-Grant, но заметно отличающиеся от HRU.

В данной работе будет построено ролевое описание некоторого частного случая объектно-ориентированной модели HRU, но в первую очередь мы рассмотрим обратное отображение, позволяющее смоделировать ролевую политику разграничения доступа на основе дискреционной модели Харриссона–Руззо–Ульмана.

В работе [5] 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.

## 1. Объектно-ориентированная модель безопасности с дискреционным разграничением доступа (OOHRU)

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.

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 $o'' \in \mathbf{O}$, то 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 [5]:

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 [5].

## 2. Ролевая политика безопасности

Ролевая политика разграничения доступа вводит в субъектно-объектную модель компьютерной системы новый класс активных сущностей — класс ролей.

Компьютерная система представляется совокупностью следующих множеств: множества пользователей $U$, множества ролей $\mathbf{R}$, множества полномочий $\mathbf{P}$ и множества сеансов работы пользователей с системой [3]. Множество объектов системы не задаётся в явном виде, а представляется через множество полномочий $\mathbf{P}$: каждое полномочие заключает в себе отношение на декартовом произведении множества прав доступа и множества объектов системы.

Ролевые отношения устанавливаются отображениями, связывающими множество ролей с множеством полномочий и множеством пользователей.

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 $R_c \subset \mathbf{R}$ и подмножество допустимых полномочий $P_c \subset \mathbf{P}$. Считается, что система функционирует безопасно, если пользователь $u_c$ может осуществлять действия только в рамках полномочий из множества $P_c$ 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. Такой подход, тесно связанный с объектно-ориентированной парадигмой, редко упоминается при рассмотрении ролевых моделей разграничения доступа, но на деле бывает очень полезен, когда иерархия ролей строится путём «от абстрактного к конкретному». Так, например, можно рассмотреть фрагмент иерархии «Сотрудник — Сотрудник финансового отдела — Бухгалтер — Главный бухгалтер». Здесь каждая последующая роль цепочки является уточнением предыдущей. Сохраняя полномочия роли-родителя, она получает новые полномочия.

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.

## 3. Связь между субъектно-объектной ролевой моделью без иерархии на множестве ролей и OOHRU

Основной результат данной работы заключается в том, что субъектно-объектная ролевая модель может быть реализована объектно-ориентированной моделью OOHRU.

В первую очередь договоримся представлять полномочия ролевой модели в виде совокупности так называемых «элементарных» полномочий, каждое из которых задаётся парой $(x, r)$ и заключает в себе право доступа $r \in R$ к объекту или группе одинаковых объектов $x$ субъектно-объектной ролевой модели. В качестве $r$ может также выступать право вызова, если $x$ представляет собой активную сущность системы.

Дабы избежать путаницы в обозначениях, переобозначим множества ролевой модели следующим образом:

$Z = \{z_1, z_2, \dots z_n\}$ — множество ролей системы,

$Y = \{y_1, y_2, \dots y_m\}$ — множество полномочий системы,

$X$ — множество всех объектов системы (в рамках субъектно-объектной парадигмы),

$z.y$ — элементарное полномочие $y$, относящееся к роли $z$.

$z.Y$ — полный набор полномочий роли $z$, $z.Y \in Y$.

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 $\Sigma'$, вычленив права доступа $r$ из элементарных полномочий. Во-вторых, представим каждое полномочие $y \in Y$ матрицей, строки которой подписаны элементами из множества $R$, столбцы — элементами из множества $X$, а на пересечении строки $r$ и столбца $a \in X$ стоит 1, если полномочие $y$ подразумевает право доступа $r$ к объекту $a$, в противном случае стоит 0.

Рассмотрим всевозможные подмножества ролей из $Z$, всего их $2^n$. Каждому такому подмножеству $\alpha \in 2^Z$ сопоставим класс $k_\alpha \in K$. Иерархия на классах вводится естественным образом: класс $k_\alpha$ является непосредственным наследником класса $k_\beta$, если и только если существует такая роль $z$, что $\alpha = \beta \cup z$.

(На самом деле достаточно ввести только классы, соответствующие наборам ролей, которыми могут наделяться пользователи системы, чтобы избежать избыточности классов. В этом случае иерархия строится аналогичным образом: класс $k_\alpha$ является непосредственным наследником класса $k_\beta$ если и только если существует такое подмножество ролей $\gamma \subset Z$, что $\alpha = \beta \cup \gamma$, но ни для какого подмножества $\gamma' \subset \gamma$ класса $k_{\beta \cup \gamma'}$ не существует.)

В дальнейшем мы будем ссылаться на эту иерархию классов как на естественную иерархию подмножеств ролей.

Теперь если пользователь $u \in U$ при авторизации на сеанс $c \in C$ получает набор ролей $\alpha$, то в модели $\Sigma'$ в классе $k_\alpha$ создаётся новый объект $o_u \in O$. 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.

Также нам необходимо ввести классы для объектов доступа ролевой модели. Так как в ролевой модели отсутствует формальное описание объектов системы, извлекать объекты будем из полномочий. Пусть $X$ — множество всех объектов системы (в рамках субъектно-объектной парадигмы). Произведём разбиение на этом множестве по следующему правилу: два объекта $a$ и $b$ из $X$ отнесём к разным блокам разбиения, если выполнено хотя бы одно из следующих условий:

1. Объекты $a$ и $b$ обладают принципиально различной природой (как,

например, текстовый документ и исполняемое приложение), то есть требуют различного набора методов взаимодействий.

2. Существует полномочие $y \in Y$, в матрице которого столбцы, соответствующие объектам $a$ и $b$, не совпадают.

Таким образом, объекты попадают в один блок разбиения, если над ними можно осуществлять одни и те же операции одними и теми же методами, и в матрице любого полномочия столбцы этих объектов совпадают.

Поскольку система конечна, конечным окажется и число блоков разбиения. Каждому такому блоку $X' \subset X$ сопоставим класс $k_{X'}$, а каждому объекту этого блока — объект класса $k_{X'}$. Структура объектов класса будет повторять структуру объектов исходной системы.

Переход от субъектно-объектной модели множества $X$ к объектно-ориентированной HRU-модели произведём согласно алгоритму, изложенному в работе [5]. В частности, пассивной сущности $x \in X$ субъектно-объектной модели будет соответствовать объект $o$, содержащий поле $f$, содержащее информацию объекта $x$, и приватное поле $M$ — матрицу доступов. Активной сущности $x'$ субъектно-объектной модели будет соответствовать объект $o'$, содержащий метод $s$, выполняющий активные функции субъекта $x'$, и приватное поле $M$ — матрицу доступов.

Заполнение матриц доступов объектов происходит следующим образом: если множество ролей $\alpha \in 2^Z$ содержит роль с 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 $k_\alpha$ заносится право доступа $r$: $o.M[k_\alpha, f] := o.M[k_\alpha, f] \cup r$.

Если множество ролей $\alpha$ содержит роль с 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 $k_\alpha$ заносится 1: $o.M[k_\alpha, f] := 1$.

Отсутствие перераспределения доступов в рамках одного сеанса влечёт отсутствие таких элементарных операторов, как $Enter$, $Delete$, $Grant$, $Deprive$ в командах модели $\Sigma'$. Старт пользователем $u$ сеанса $c$, в котором ему назначается множество ролей $\alpha$, будет выражен в $\Sigma'$ следующей командой:

$CommandStartSession\_\alpha(o_u : k_\alpha)$

$\quad Create(o_u, k_\alpha)$.

Команда завершения сеанса этим же пользователем:

$CommandEndSession\_\alpha(o_u : k_\alpha)$

$\quad Destroy(o_u)$.

Тот факт, что перед стартом сеанса выполняется проверка возможности пользователю $u$ назначить множество ролей $\alpha$, 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 $u$ ролей из множества $\alpha$. В этом случае при создании объекта $o_c$ происходит модификация матриц доступа объекта $o*_u$ и класса $k_\alpha$ следующей командой:

$CommandCreateSession\_\alpha(o_c : k_C; o*_u : k_U; k_\alpha)$

$Create(o_c, k_C)$,

$Grant(o_c, o *_u .user)$,

$Grant(o_c, k_\alpha.user)$.

При выполнении этой команды матрицы доступов модифицируются следующим образом: $o *_u .M[o_c, user] = 1$, $k_\alpha.M[o_c, user] = 1$ (объект $o_c$ получает право запускать метод $user$ объекта $o*_u$, относящегося к пользователю $u$, и аналогичный метод класса $k_\alpha$). 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 множество ролей $\alpha$:

$CommandStartSession\_\alpha(o_c : k_C; o*_u : k_U; o_u : k_\alpha)$

If

$o *_u .M[o_c, user] = 1$ and $k_\alpha.M[o_c, user] = 1$

then

$Create(o_u, k_\alpha)$.

Для смены набора назначенных ролей $\alpha$ на набор $\beta$ пользователю необходимо завершить текущий сеанс и начать новый, в котором ему будет назначен требуемый набор ролей (конечно, если такой сеанс существует). В OOHRU это осуществимо последовательностью команд:

$CommandEndSession\_\alpha$,

$CommandStartSession\_\beta$.

Также средства OOHRU позволяют модифицировать множество привилегий, назначенных роли $z$, однако это потребует введения в модель дополнительных сущностей.

Заметим, что построенная нами модель OOHRU будет однородной в зоне классов естественной иерархии подмножеств ролей. Сформулируем полученный результат в виде теоремы.

**Теорема 1.** *Для любой субъектно-объектной базовой ролевой модели, свободной от иерархии, существует реализующая её иерархическая модель OOHRU.*

## 4. Связь между субъектно-объектной ролевой моделью с иерархией на множестве ролей и OOHRU. Случай наследования «сверху»

Перейдём к рассмотрению случая ролевой модели с иерархией, основанной на наследовании «сверху». Классы объектов, а также вспомогательные класс пользователей и класс сеансов в реализующей её модели OOHRU останутся неизменными, необходимо построить лишь иерархию классов ролей.

Иерархия классов ролей в OOHRU будет повторять иерархию ролей в исходной ролевой модели. Так, каждой роли $z$ будет сопоставлен класс $k_z$, и если роль $z'$ является непосредственным потомком роли $z$ в исходной 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 индуцированной иерархией ролей.

Как и ранее, если пользователь $u \in U$ при авторизации на сеанс $c \in C$ получает роль $z$, то в модели $\Sigma'$ 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.

Переход от субъектно-объектной модели множества $X$ к объектно-ориентированной HRU-модели произведём согласно алгоритму, изложенному в работе [5].

Заполнение матриц доступов объектов происходит следующим образом: если роль $z$ 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 роль $z$ обладает элементарным полномочием $y = (x', r) = (o'.p, r)$, 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: $o.M[k_z, f] = 1$.

Дополним OOHRU командой старта сеанса $c$ пользователем $u$ с авторизацией его на роль $z$ и командой завершения сеанса.

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

Команда завершения сеанса этим же пользователем:

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

Первая команда может быть дополнена проверкой возможности сеанса $c$:

$CommandStartSession\_z(o_c : k_C; o*_u : k_U; o_u : k_z)$

If
$o_u.M[o_c, user] = 1$ and $k_z.M[o_c, user] = 1$
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 подход иерархии «вниз» на множестве ролей, назначение одной роли субъекту кажется достаточным, поскольку роль определяется непосредственно сферой обязанностей пользователя, и при необходимости совмещения нескольких обязанностей достаточно создать роль, являющуюся в иерархии наследницей ролей, упомянутых выше обязанностей. Так, если в системе существуют роли $z$ и $z'$, которые необходимо назначить одному пользователю, совмещающему соответствующие обязанности, то достаточно создать новую роль $z''$, набор полномочий которой будет являться объединением наборов полномочий ролей $z$ и $z'$, а сама роль $z''$ — наследником как $z$, так и $z'$.

Однако 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 $\alpha$.

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

Модификация множества привилегий, назначенных роли $z$, осуществима следующим образом. Допустим, роли $z_1 \dots z_t$ являются непосредственными потомками роли $z$, которой необходимо добавить новое элементарное полномочие $y = (o.f, r)$, то есть право доступа $r$ к полю $f$ объекта $o$ некоторого класса $k_o$. Для этого используем команду

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

Данная команда не нарушает наследование прав доступа в иерархии, поскольку в OOHRU оператор $Enter$ может быть выполнен, только если соблюдены так называемые условия целостности [1]:

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

При необходимости добавить более широкий набор полномочий, расширяем список элементарных операторов $Enter$ в команде, добавляющих необходимые 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:

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

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

Модификация строк матриц доступа объекта $o$, соответствующих объектам класса $k_z$, не требуется за отсутствием таковых, поскольку изменение набора полномочий, соответствующего роли, не может осуществляться в ролевой модели безопасности во время сеанса, завязанного на эту роль.

Тем самым доказана

**Теорема 2.** *Для любой субъектно-объектной иерархической ролевой модели с наследованием «сверху» существует реализующая её иерархическая модель OOHRU.*

## Литература

1. Усов С.В. Объектно-ориентированный подход в построении политики безопасности. Системы с естественной иерархией. // Математические структуры и моделирование. 2010. N. 21. С. 152-162.
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. Sandhu R., Munawer Q. How to do discretionary access control using roles // 3rd ACM Workshop on Role-Based Access Control. 1998. P. 47–54.
5. 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

**S.V. Usov**
Ph.D. (Eng.), Associste 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 model without hierarchy and the role-based security model with hierarchy with inheritance "from above" 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.

*Дата поступления в редакцию: 15.11.2018*