МАТЕМАТИЧЕСКИЕ ОСНОВЫ КОМПЬЮТЕРНОЙ БЕЗОПАСНОСТИ

DOI 10.17223/20710410/16/4

УДК 004.056.52

СЕМАНТИЧЕСКАЯ РОЛЕВАЯ МОДЕЛЬ УПРАВЛЕНИЯ ДОСТУПОМ

Н. А. Семенова

Московский государственный институт электроники и математики (ТУ), г. Москва, Россия

E-mail: natasha_sem@inbox.ru

Приводится описание расширения ролевой модели $RBAC$, используемой для управления доступом в автоматизированную информационную систему (АИС) предприятия, за счёт введения семантического контекста, позволяющего отразить взаимосвязь между должностными обязанностями сотрудников и ролями, назначенными соответствующим им учётным записям пользователей в системе. Основное отличие модели — возможность автоматизации операций назначения и отзыва ролей. Приведено доказательство безопасности системы ролевого управления доступом, построенной на основе предложенной семантической модели.

Ключевые слова: *ролевое управление доступом, автоматизация управления ролями.*

Введение

Модель ролевого управления доступом $RBAC$ [1] применяется для построения систем управления доступом в АИС. Одной из особенностей классической модели $RBAC$ является то, что все операции по назначению и отзыву ролей выполняются вручную администраторами системы. При большом количестве ролей в АИС нагрузка на администраторов возрастает, что может привести к увеличению времени, необходимому для обеспечения пользователя АИС, соответствующего некоторому сотруднику предприятия, всеми нужными ему для выполнения должностных обязанностей правами доступа [2]. В данной работе предлагается ролевая модель управления доступом, основанная на классической модели и позволяющая автоматизировать назначение и отзыв ролей учётным записям пользователей при наступлении в системе ряда определённых событий.

1. Основные понятия и определения классической модели $RBAC$

Рассмотрим некоторые основные понятия классической модели $RBAC$, которые потребуются для дальнейшего описания [3]:

P — множество возможных прав доступа;

U — множество учётных записей пользователей системы;

$R \subseteq 2^P$ — множество ролей системы;

$UA : U \to 2^R$ — функция, задающая для каждой учетной записи пользователя множество ролей, на которые она может быть авторизована;

$PA : R \to 2^P$ — функция, задающая для каждой роли множество прав доступа. При этом для каждого права доступа $p \in P$ существует роль $r \in R$, такая, что $p \in PA(r)$;

$user : S \to U$ — функция, задающая для каждой сессии учётную запись, от имени которой она авторизована.

Определение 1. Иерархией ролей в классической модели ролевого управления доступом называется заданное на множестве ролей R отношение частичного порядка $\geqslant$. При этом по определению выполняется условие: для учётной записи пользователя $u \in U$ если $r_1, r_2 \in R$, $r_1 \in UA(u)$ и $r_1 \geqslant r_2$, то $r_2 \in UA(u)$.

Зададим $RH \subseteq R \times R$ — иерархическое отношение на множестве ролей — следующим образом: $(r_1 \geqslant r_2) \Leftrightarrow ((r_1, r_2) \in RH)$. Если $r_1 \geqslant r_2$, то r_1 считается старшей по отношению к r_2 (предком r_2), а r_2 — младшей (потомком r_1).

Определение 2. Два права доступа $p_1, p_2 \in P$ называются статически взаимоисключающими, если они не могут входить в состав одной и той же роли одновременно: для любой $r \in R$ выполняется неравенство $\{p_1, p_2\} \cap PA(r) \leqslant 1$.

Определение 3. Две роли $r_1, r_2 \in R$ называются статически взаимоисключающими, если они не могут быть назначены одной учётной записи пользователя одновременно: для любой $u \in U$ выполняется неравенство $\{r_1, r_2\} \cap UA(u) \leqslant 1$.

Определение 4. Множество предварительных условий CR — это множество, включающее в себя ограничения на роли, которыми должна обладать учётная запись пользователя, чтобы быть назначенной на новую роль, а также ограничения взаимного исключения ролей. Пусть $z_r : U \to \{\texttt{false}, \texttt{true}\}$ — функция, такая, что $z_r(u) = \texttt{true}$ тогда и только тогда, когда для $u \in U$ и $r \in R$ выполняется условие $r \in UA(u)$. Пусть $c_r : U \to \{\texttt{false}, \texttt{true}\}$ — функция, такая, что $c_r(u) = c_r(z_{r_1}(u), \ldots, z_{r_k}(u))$, где $u \in U$; $r, r_1, \ldots, r_k \in R$; $r \neq r_i$ для $i = 1, \ldots, k$ и $c_r(y_1, \ldots, y_k)$ — булева функция от k переменных. Тогда $CR = \{c_{r_1}, \ldots, c_{r_t}\}$ — множество функций, определяющих, при каких условиях той или иной учётной записи пользователя может быть назначена некоторая роль $r_j \in R$.

Для обеспечения возможности администрирования множества назначенных учётным записям пользователей ролей используем следующие функции, определённые в стандарте ARBAC'97 [4]:

- AP — множество административных прав доступа;
- $AR \subseteq 2^{AP}$ — множество административных ролей;
- $APA : AR \to 2^{AP}$ — функция, задающая для каждой административной роли множество административных прав доступа; при этом для каждого права доступа $p \in AP$ существует роль $r \in AR$, такая, что $p \in APA(r)$;
- $AUA : U \to 2^{AR}$ — функция, задающая для каждой учётной записи пользователя множество административных ролей, на которые она может быть авторизована;
- $can\_assign : AR \to CR \times 2^R$ — функция, определяющая для каждой административной роли множество ролей, которые могут быть назначены учётной записи пользователя с использованием данной административной роли при выполнении заданных предварительных условий CR;
- $can\_revoke : AR \to 2^R$ — функция, определяющая для каждой административной роли множество ролей, которые могут быть отозваны у учётной записи пользователя с использованием данной административной роли;
- $roles : S \to 2^R \cup AR$ — функция, задающая для учётной записи пользователя множество ролей, на которые она авторизована в данной сессии. При этом в каждый момент времени для каждой сессии $s \in S$ выполняется условие $roles(s) \in \in UA(user(s)) \cup AUA(user(s))$.

Определение 5. Под классической системой ролевого управления доступом будем понимать конечный автомат, заданный кортежем $K = \langle \Gamma, Q, a, \Psi \rangle$, где Γ — множество состояний; Q — множество запросов на доступ к объектам АИС; Ψ — множество действий, переводящих систему из состояния в состояние; $a : \Gamma \times Q \to \{\texttt{true}, \texttt{false}\}$ — функция, определяющая, является ли запрос истинным в заданном состоянии. Состояние системы задаётся кортежем $\gamma = (S, U, UA, user, roles) \in \Gamma$, а правила перехода между состояниями определяются предварительными условиями CR.

Таким образом, классическая модель $RBAC$ не содержит в себе информацию о связи между должностными обязанностями пользователей и ролями, назначенными их учётным записям (о семантическом смысле назначения ролей). Ещё одним важным ограничением классической модели $RBAC$ является то, что роли всегда назначаются и отзываются сессией, инициированной некоторой учётной записью пользователя системы, обладающей административными правами, и не могут быть назначены (отозваны) при наступлении заданного события, поскольку модель не содержит в себе средств описания событий и логики реакции на них. Далее описаны предлагаемые расширения классической модели за счёт введения семантического контекста.

2. Семантическая ролевая модель управления доступом

2.1. Иерархия ролей «по предусловию»

В классической модели $RBAC$ считается, что родительская роль содержит в себе все права дочерних ролей, а также более высокопривилегированные права доступа, чем дочерние (например, роль «Начальник отдела» является родительской для роли «Рядовой сотрудник» и содержит как все права доступа, назначенные рядовым сотрудникам, так и права, которыми обладают только руководители структурной единицы предприятия). В реальных АИС иерархические отношения между ролями обычно имеют следующий смысл [5]: меньшая роль описывает некоторые базовые права доступа к объекту, а родительские роли — более функциональные права доступа. В качестве примеров таких АИС можно привести службу каталогов *ActiveDirectory* [5], где базовым правом доступа является включение учётной записи в домен (роль *DomainUsers*), а также СУБД *Oracle* [6], содержащую базовые права доступа $CONNECT$ и $CREATESESSION$, разрешающие создание соединения с базой данных (роль *DatabaseUser*). Таким образом, родительские роли в общем случае не включают в себя права доступа дочерних ролей. В то же время можно показать, что в классической модели $RBAC$ существует неявно выраженная зависимость между предварительными условиями назначения ролей, находящихся в иерархической связи друг с другом. Пусть в классической модели $RBAC$ роль r_1 является старшей для роли r_2. Тогда, по определению 1, для учётной записи пользователя $u \in U$ если $(r_1, r_2) \in RH$, $r_1 \in UA(u)$ и $r_1 \geqslant r_2$, то $r_2 \in UA(u)$. В этом случае можно утверждать, что учётная запись пользователя u удовлетворяет условиям: если $c_{r_1}(u) = \texttt{true}$, то $c_{r_2}(u) = \texttt{true}$, в противном случае роль r_2 не могла бы быть назначена учётной записи пользователя. Таким образом, существует неявным образом выраженная зависимость между предварительными условиями назначения ролей r_1 и r_2.

Переопределим иерархические связи между ролями, используя описание множества предварительных условий CR.

Определение 6. Иерархией ролей «по предусловию» будем называть заданное на множестве ролей R отношение строгого порядка $>$. При этом для учётной записи пользователя $u \in U$ если $(r_1, r_2) \in RH$, $r_1 > r_2$ и $c_{r_1}(u) = \texttt{true}$, то $c_{r_2}(u) = \texttt{true}$ и

$c_{r_1}(u) = c_{r_1}(z_1(u), \ldots, c_{r_2}(u), \ldots, z_k(u))$. Роль r_1 будем называть предком r_2 «по предусловию».

Таким образом, по определению, для возможности авторизации на некоторую роль учётная запись пользователя должна удовлетворять и всем предварительным условиям для дочерней роли.

Предположение 1. В данной работе предполагается, что в модели действуют ограничения статического взаимного исключения прав доступа для ролей, состоящих в иерархических отношениях: для каждого множества ролей, состоящих в иерархии, каждое право доступа может входить в состав только одной роли: $P = PA(r_1) \cup \cdots \cup \cup PA(r_m)$, где $(r_i, r_j) \in RH$; $PA(r_i) \cap PA(r_j) = \varnothing$ для $1 \leqslant i < j \leqslant m$, $r_i, r_j \in R$.

Предположение 2. Будем считать, что если для некоторой административной роли $r_a \in AR$ и роли $r \in R$ выполняется условие $r \in can\_revoke(r_a)$, то для каждой роли $r_0 \in R$, такой, что $r_0 > r$, выполняется условие $r_0 \in can\_revoke(r_a)$.

Сформулируем ряд утверждений, описывающих свойства ролевой модели управления доступом, в которой выполняются условия предположений 1 и 2, а иерархия ролей строится по определению 6.

1) Отзыв у учётной записи пользователя родительской роли не влечёт за собой автоматического отзыва всех дочерних ролей, поскольку множества прав доступа, входящих в родительскую и дочерние роли, не пересекаются (не выполняется требование, справедливое для классической модели: если $r_1 \geqslant r_2$, то $PA(r_1) \supseteq PA(r_2)$).

2) Отзыв дочерней роли приводит к отзыву всех родительских ролей. По определению 6, при удалении у учётной записи пользователя u некоторой роли r для u перестают выполняться предварительные условия для всех ролей, родительских для r. Это означает, что в результате удаления r у учётной записи пользователя одновременно отзываются и все родительские для r роли. Для возможности каскадного удаления ролей у учётной записи пользователя необходимо выполнение условий предположения 2.

3) Каскадное назначение прав на отзыв административным ролям в направлении «снизу вверх» не влечёт за собой превышения прав доступа административных пользователей, поскольку родительские роли не включают в себя права доступа дочерних ролей.

Определение 7. Пусть $INIT : \Psi \to 2^U$ — функция, задающая для каждого действия $\sigma \in \Psi$ множество возможных инициаторов его выполнения. Говорим, что субъекты (учётные записи) из множества значений функции $INIT(\sigma) \subseteq U$ могут быть инициаторами действия $\sigma \in \Psi$, переводящего систему $\langle \Gamma, Q, a, \Psi \rangle$ из состояния $\gamma \in \Gamma$ в состояние $\gamma_g \in \Gamma$, если сессии, функционирующие от имени данных субъектов, могут выполнить действие $\sigma \in \Psi$, приводящее к переходу модели из состояния $\gamma \in \Gamma$ в состояние $\gamma_g \in \Gamma$, т. е. если существует действие $\sigma \in \Psi$, для которого выполняются следующие условия:

— $init(\sigma) \in INIT(\sigma)$, где $init : \Psi \to U$ — функция, такая, что $init(\sigma) = user(s)$, если сессия s выполнила действие σ;
— $\sigma_1(\gamma) = \gamma_g$.

Таким образом, для каждого возможного действия в системе значение функции $INIT$ — это совокупность учётных записей пользователей, обладающих в рамках заданной сессии s административными правами, позволяющими им выполнить данное действие. Будем использовать обозначение $INIT(\sigma_1, \sigma_2, \ldots, \sigma_i) = INIT(\sigma_1) \times \times INIT(\sigma_2) \times \cdots \times INIT(\sigma_i)$.

Пример 1. Множество учётных записей пользователей, обладающих возможностью назначить роль $r \in R$ учётной записи пользователя $u \in U$, задаётся выражением $\{u_a \in U : u_a \in AUA(can\_assign^{-1}(c_r(u), r))\}$, где $c_r \in CR$ — предварительное условие, определяющее, может ли роль r быть назначена учётной записи пользователя u.

2.2. Атрибуты учётной записи пользователя

Дадим определение семантически осмысленной ролевой модели, являющейся расширением классической модели ролевого управления доступом. Введём следующие дополнительные обозначения:

- A — множество атрибутов учётных записей пользователей. Элементами A являются атрибуты, характеризующие положение сотрудника, соответствующего учётной записи пользователя АИС, в организационно-штатной структуре предприятия, его должность и другие признаки, позволяющие определить его служебные обязанности. Например, $A = \{a_1, a_2\} = \{$«Должность», «Отдел»$\}$.
- V — множество допустимых значений атрибутов. Для каждого элемента множества A множество V содержит вектор значений, которые может принимать данный атрибут: $V = \{(v_{ij})\}$, где i — номер атрибута из A; $j = 1, \ldots, N_i$; N_i — число возможных значений атрибута $a_i \in A$.
- $values : A \to 2^V$ — функция, задающая для каждого атрибута множество его допустимых значений. Например, $V = \{values(a_1), values(a_2)\} = \{(v_1), (v_2)\} = \{(v_{11}, v_{12}, v_{13}), (v_{21}, v_{22}, v_{23})\}=\{$(«Специалист», «Старший специалист», «Ведущий специалист»), («Бухгалтерия», «Отдел Кадров», «Разработка»)$\}$.

Для обеспечения возможности администрирования множества атрибутов учётных записей пользователей используем функцию $can\_change\_attr : AR \to 2^U$, определяющую для каждой административной роли множество учётных записей, атрибуты которых могут быть изменены с использованием данной административной роли. Покажем теперь, каким образом определение элементов множества учётных записей пользователей U может быть расширено за счёт данных об атрибутах учётных записей пользователей.

Определение 8. Множество атрибут-пользователей $\hat{U}$ — это множество векторов $(u, ux_1, \ldots, ux_{na})$, где $ux_i \in values(a_i), u \in U$.

Элементы множества $\hat{U}$ описывают учётные записи пользователей АИС в соответствии с должностными обязанностями, выполняемыми связанными с ними сотрудниками. При этом будем считать, что для $\hat{u} = (u, ux_1, \ldots, ux_{na}) \in \hat{U}$ заданы функции, аналогичные функциям для элементов из множества U, следующим образом:

- $c_r(\hat{u}) = c_r(u)$;
- $UA(\hat{u}) = UA(u)$;
- $AUA(\hat{u}) = AUA(u)$.

Пример 2. Элемент множества $\hat{U}$, представляющий учётную запись пользователя АИС, соответствующую сотруднику отдела кадров, занимающему должность старшего специалиста, может быть представлен как $\hat{u} =$ (u, «Старший специалист», «Отдел кадров»), где u — элемент множества U, соответствующий данной учётной записи пользователя в АИС.

Для обеспечения возможности автоматического (без вмешательства администратора) изменения множества авторизованных ролей учетных записей пользователей АИС при изменении атрибутов, описывающих их должностные обязанности, необходимо ввести в модель управления доступом следующие элементы:

— учётную запись пользователя $system$, от имени которой выполняются все автоматические операции по изменению авторизованных ролей учётных записей пользователей АИС;
— учётную запись пользователя $attribute\_source$, от имени которой выполняются все автоматические операции по изменению атрибутов учётных записей пользователей АИС. Учётная запись пользователя $attribute\_source$ обладает административной ролью $r_{ua} \in AR$, дающей право изменения атрибутов учётной записи любого пользователя из $\hat{U}$, но не авторизована ни на одну из других ролей из AR или R: $UA(attribute\_source) = r_{ua}$;
— правила, описывающие взаимосвязь между атрибутами учётных записей пользователей и назначенными им ролями. Будем называть такие правила атрибут-условиями.

Определение 9. Определим множество атрибут-условий CA.

Пусть $t : \hat{U} \to \{\texttt{false}, \texttt{true}\}$ — функция, такая, что $t_i(\hat{u}) = \texttt{true}$ тогда и только тогда, когда для $\hat{u} \in \hat{U}$ выполняется условие $ux_i = t$.

Пусть $ca_r : \hat{U} \to \{\texttt{false}, \texttt{true}\}$ — функция, такая, что $ca_r(\hat{u}) = ca_r(t_1(\hat{u}), \ldots, t_{na}(\hat{u}))$, $\hat{u} \in \hat{U}$, $r \in R$, где $ca_r(y_1, \ldots, y_{na})$ — булева функция от na переменных, $na = 1, \ldots, A$. Тогда $CA = \{ca_{r1}, \ldots, ca_{rt}\}$ — множество функций, определяющих, при каких ограничениях на атрибуты той или иной учётной записи пользователя может быть назначена некоторая роль $r_i \in R$.

Пример 3. Пусть $A = \{a_1, a_2\} = \{$«Должность», «Отдел»$\}$, $V = \{values(a_1), values(a_2)\} = \{$(«Специалист», «Старший специалист», «Ведущий специалист»), («Бухгалтерия», «Отдел Кадров», «Разработка»)$\}$. Тогда условие для роли r, назначаемой учётной записи пользователя $\hat{u}$, соответствующего сотруднику бухгалтерии с должностью «Ведущий специалист», имеет вид $ca_r(\hat{u}) = (t_1(\hat{u}), t_2(\hat{u})) =$(«Ведущий специалист»$(\hat{u})$, «Бухгалтерия»$(\hat{u})$).

Определение 10. Если роль r имеет связанное атрибут-условие $ca_r \in CA$ и назначение её учётной записи пользователя $\hat{u}$ по этому условию инициировано сессией от имени учётной записи пользователя $system$, то такое назначение называется автоматическим, а роль — атрибут-ролью.

Атрибут-роль $r \in R_a$ может быть автоматически назначена учётной записи пользователя $(u, ux_1 \ldots ux_{na}) = \hat{u} \in \hat{U}$ тогда и только тогда, когда $ca_r(\hat{u}) = \texttt{true}$ и выполняются предварительные условия из CR (т. е. $c_r(\hat{u}) = \texttt{true}$).

Определение 11. Роли, для которых не существует ни одного атрибут-условия $ca_r \in CA$, будем называть классическими, а способ их назначения учётным записям пользователей — классическим.

По определению функции can_assign классическая роль $r \in R_k$ может быть назначена учётной записи пользователя $(u, ux_1, \ldots, ux_{na}) = \hat{u} \in \hat{U}$ сессией, функционирующей от имени администратора $\hat{u_2} \in \hat{U}$, тогда и только тогда, когда $r \in \in can\_assign\,(CR, AUA(u_2))$.

Предположение 3. Будем считать, что если для некоторой роли $r \in R$ существует атрибут-условие $ca_r \in CA$, то она всегда назначается автоматически и не может быть назначена классическим способом. В этом случае множество всех ролей R может быть представлено в виде $R = R_k \cup R_a$, $R_k \cap R_a = \varnothing$, где R_k — классические роли и R_a — множество атрибут-ролей.

Предположение 4. Будем считать, что учётная запись *system* авторизована на все административные роли $AR_a \in AR$, такие, что $can\_revoke^{-1}(AR_a) = R_a$ и $can\_assign^{-1}(AR_a) = R_a$.

Утверждение 1. Пусть атрибут-роли $r_1, r_2 \in R_a$ таковы, что $r_1 > r_2$ и атрибут-условие для роли r_2 имеет вид $ca_{r_2}(\hat{u}) = ca_{r_2}(t_1(\hat{u}), \ldots, t_k(\hat{u}))$. Тогда атрибут-условие для роли r_1 имеет вид $ca_{r_1}(\hat{u}) = ca_{r_1}(t_{k+1}(\hat{u}), \ldots, ca_{r_2}(\hat{u}), \ldots, t_{k+p}(\hat{u}))$, $t_i \in values(a_i)$, т. е. для возможности авторизации на некоторую роль учётная запись пользователя должна удовлетворять и всем атрибут-условиям для дочерних ролей.

Доказательство. Пусть $r_1 \in UA(\hat{u})$, $\hat{u} \in \hat{U}$, $r_1 \in R_a$. Отсюда следует, что $ca_{r_1}(\hat{u}) = \texttt{true}$. По определению иерархии, $r_2 \in UA(\hat{u})$ и $ca_{r_2}(\hat{u}) = \texttt{true}$. Таким образом, из $ca_{r_1}(\hat{u}) = \texttt{true}$ всегда следует $ca_{r_2}(\hat{u}) = \texttt{true}$. Следовательно, $ca_{r_2}(\hat{u}) = \texttt{true}$ является необходимым условием для $ca_{r_1}(\hat{u}) = \texttt{true}$, и $ca_{r_1}(\hat{u}) = ca_{r_1}(t_1(\hat{u}), \ldots, ca_{r_2}(\hat{u}), \ldots, t_k(\hat{u}))$. ■

Определение 12. Функция ограничения действий над учётной записью пользователя — это функция $H : \hat{U} \times \Gamma \times \Psi \to 2^{\Psi}$, значение которой для каждого действия $\sigma_i \in \Psi$, выполненного над учётной записью пользователя $\hat{u} \in \hat{U}$ и приведшего систему в текущее состояние $\gamma \in \Gamma$, равно множеству действий, которые могут быть выполнены над заданной учётной записью пользователя для перехода в последующее состояние. Множество Ψ всех возможных действий над учётной записью пользователя описано в таблице.

Пример 4. Пусть $H(\hat{u}, \gamma_t, \sigma_1) = \{\sigma_2, \sigma_3, \ldots, \sigma_q\}$. Это означает, что после выполнения над учётной записью пользователя $\hat{u}$ действия σ_1 над данной учётной записью в системе могут быть выполнены только действия из подмножества $\{\sigma_2, \sigma_3, \ldots, \sigma_q\} \subseteq \Psi$.

Определение 13. Траектория переходов $Tr : \hat{U} \times \Gamma \times \Gamma \times H \to 2^{\Psi}$ — это функция, задающая конечную последовательность действий $\gamma_i \in \Psi$, которая должна быть выполнена над учётной записью пользователя $\hat{u} \in \hat{U}$ в системе, находящейся в состоянии $\gamma_0 \in \Gamma$, для перехода системы в некоторое новое состояние $\gamma_1 \in \Gamma$ при условии, что множество возможных действия ограничено значением функции $H(\hat{u}, \gamma_0, \Psi)$.

Траектория используется пользователем *system* для определения последовательности выполнения автоматических действий по отзыву и назначению ролей.

Предположение 5. Администраторы системы могут выполнять действия над учётными записями пользователей в произвольном порядке при условии, что одновременно над этими учётными записями не выполняет никаких действий сессия от имени *system* (т. е. в текущий момент времени траектория действий для данной учетной записи пользователя пуста).

Пусть $Tr(\hat{u}, \gamma_0, \gamma, H(\hat{u}, \gamma_0, \sigma_0)) = \{\sigma_1, \ldots, \sigma_n\}$. Введём обозначения: $tr[0] = \sigma_1$ — первое действие, выполненное в системе над учётной записью $\hat{u}$ из состояния γ_0; $tr[1] = \sigma_2$ — второе действие и т. д. То есть если $\gamma = \sigma_n(\ldots\sigma_2(\sigma_1(\sigma_0(\gamma_0))))$, то $Tr(\hat{u}, \gamma_0, \gamma, H(\hat{u}, \gamma_0, \sigma_0)) = (tr[0], \ldots, tr[n]) = (\sigma_1, \ldots, \sigma_n)$, где $\sigma_i \in H(\hat{u}, \gamma_i, tr[i-1])$. Ограничения на разрешённые для каждого состояния системы действия и порядок их выполнения необходимы, так как операторы не являются коммутативными. Результирующее состояние системы зависит от порядка, в котором были выполнены действия.

Правила преобразования состояний семантической ролевой модели

Правило	Исходное состояние $\gamma_1 = (S, \hat{U}_1, UA_1, user_1, roles_1, H, TR_1)$	Результирующее состояние $\gamma_2 = \left(S, \hat{U}_2, UA_2, user_2, roles_2, H, TR_2\right)$
σ_1=$take\_role(x, r)$ — активация сессией одной из авторизированных ролей	$\sigma_1 \in H(user(x), \gamma_1, \sigma)$, $x_1 \in S$, $r \in R$, $r \in UA(user(x))$	$H(user(x), \gamma_2, \sigma_1) = H(user(x), \gamma_1, \sigma)$, $TR_2 = TR_1$, $S_2 = S_1$, $PA_2 = PA_1$, $user_2 = user_1$, $UA_2 = UA_1$, $roles_2(x) = roles_1(x) \cup \{r\}$, для $x_2 \in S \setminus \{x_1\}$ выполняется $roles_2(x_2) = roles_1(x_2)$
σ_2=$remove\_role(x, r)$ — удаление из сессии одной из ранее активированных ролей	$s_2 \in H(user(x), \gamma_1, \sigma)$, $x_1 \in S$, $r \in R$, $r \in roles(x)$	$TR_2 = TR_1$, $S_2 = S_1$, $PA_2 = PA_1$, $H(user(x), \gamma_2, \sigma_2) = H(user(x), \gamma_1, \sigma)$, $user_2 = user_1$, $UA_2 = UA_1$, $roles_2(x_1) = roles_1(x_1) \setminus \{r\}$, для $x_2 \in S \setminus \{x_1\}$ выполняется $roles_2(x_2) = roles_1(x_2)$
$\sigma_3 = assign\_role(u_1, u_2, r)$ — назначение роли классическим способом	$\sigma_3 \in H(u_1, \gamma_1, \sigma)$, $u_1, u_2 \in \hat{U}$, $r \in R_k$, $r \in can\_assign(CR, AUA(u_2))$	$H(u_1, \gamma_2, \sigma_3) = H(u_1, \gamma_1, \sigma)$, $S_2 = S_1$, $PA_2 = PA_1$, $user_2 = user_1$, $roles_2(x_1) = roles_1(x_1) \setminus \{r\}$, для $u \in \hat{U} \setminus \{u_1\}$ выполняется $UA_2(u) = UA_1(u)$, $UA_2(u_1) = UA_1(u_1) \cup \{r\}$, $roles_2 = roles_1$
σ_4=$revoke\_role(u_1, u_2, r)$ — отзыв роли классическим способом	$\sigma_4 \in H(u_1, \gamma_1, \sigma)$, $x \in S$, $u_1, u_2 \in \hat{U}$, $r \in R_k$, $r \in can\_revoke(AUA(u_2))$; для всех x, таких, что $user(x) = u_1$, выполняется $r \neq roles(x)$	$TR_2 = TR_1$; если для r существуют родительские роли, то $H(u_1, \gamma_2, \sigma_4) = \{\sigma_4\}$ (нельзя выполнять никаких действий, пока родительские роли не будут отозваны); иначе $H(u_1, \gamma_2, \sigma_4) = H(u_1, \gamma_1, \sigma)$; $S_2 = S_1$, $PA_2 = PA_1$, $user_2 = user_1$, для $\hat{u} \in \hat{U} \setminus \{u_1\}$ выполняется $UA_2(u) = UA_1(u)$, $UA_2(u_1) = UA_1(u_1) \setminus \{r\}$, $roles_2(x) = roles_1(x) \setminus \{r\}$
$\sigma_5 = change\_user\_attr(u, i, v)$ — изменение значения i-го атрибута учётной записи пользователя на новое значение v	$tr_1[0] = \sigma_5$, $\sigma_5 \in H(u, \gamma_1, \sigma)$, $x \in S$, $u = (u, ux_1, \ldots, ux_i, \ldots, x_n) \in \hat{U}$, $x_i \neq v$	$tr_2[0] = \sigma_8$, $H(u, \gamma_2, \sigma_5) = \{\sigma_8\}$ (после изменения атрибутов нельзя выполнять никаких действий, кроме перерасчёта ролей), $u = (u, ux_1, \ldots, v, \ldots, ux_n)$, для $\hat{u} \in \hat{U} \setminus \{u\}$ выполняется $\hat{U}_2 = \hat{U}_1$, $S_2 = S_1$, $PA_2 = PA_1$, $user_2 = user_1$, $UA_2 = UA_1$, $roles_2 = roles_1$
$\sigma_6 = auto\_assign\_role(u_1, r)$ — автоматическое назначение роли	$tr_1[0] = \sigma_6$, $x \in S$, $u_1 \in \hat{U}$, $r \in R_a$, $ca_r(u_1) =$ `true`	$H(u_1, \gamma_2, \sigma_6) = \{\sigma_6, \sigma_7\}$, $\hat{U}_2 = \hat{U}_1$, $S_2 = S_1$, $PA_2 = PA_1$, $user_2 = user_1$, для $\hat{u} \in \hat{U} \setminus \{u_1\}$ выполняется $UA_2(\hat{u}) = UA_1(\hat{u})$, $UA_2(u_1) = UA_1(u_1) \cup \{r\}$, $roles_2 = roles_1$
$\sigma_7 = auto\_revoke\_role(u_1, r)$ — автоматический отзыв роли	$tr_1[0] = \sigma_7$, $x \in S$, $u_1 \in \hat{U}$, $r \in UA(u_1)$, $r \in R_a$, $ca_r(u_1) =$ `false`	$H(u_1, \gamma_2, \sigma_7) = \{\sigma_6, \sigma_7\}$, $\hat{U}_2 = \hat{U}_1$, $S_2 = S_1$, $PA_2 = PA_1$, $user_2 = user_1$, для $u_2 \in \hat{U} \setminus \{u_1\}$ выполняется $UA_2(u_2) = UA_1(u_2)$, $UA_2(u_1) = UA_1(u_1) \setminus \{r\}$, $roles_2 = roles_1$
$\sigma_8 = auto\_recalculate(u)$ — построение траектории автоматического отзыва и назначения ролей	$tr_1[0] = \sigma_8$, $u \in \hat{U}$	$H(u, \gamma_2, \sigma_8) = \{\sigma_6, \sigma_7\}$, $TR_2 = TR_1 \cap \{\sigma_{71}, \sigma_{72}, \ldots, \sigma_{7k}, \sigma_{61}, \ldots, \sigma_{6m}\}$, где σ_{7i} — действие по отзыву роли $r_i \in R_a$, σ_{6j} — действие по назначению роли $r_j \in R_a$, $S_2 = S_1$, $PA_2 = PA_1$, $user_2 = user_1$, $UA_2 = UA_1$

Пример 5. Последовательный отзыв всех родительских ролей при отзыве дочерней роли у учётной записи пользователя $\hat{u} \in \hat{U}$ может быть выражен следующей траекторией: $Tr(\hat{u}, \gamma_t, \gamma_{t+1}, H(\hat{u}, \gamma_t, \sigma_{r_1})) = \left(\sigma_{r_2}, \ldots, \sigma_{r_q}\right)$, где σ_{r_1} — действие по отзыву роли r_1 и $r_2, \ldots, r_q$ — родительские роли для r_1. Таким образом, ни одно другое действие над данной учётной записью пользователя в системе не может быть выполнено, пока у неё не будут отозваны все роли, согласно иерархии.

Определение 14. Семантически осмысленная модель ролевого управления доступом задается множествами $\hat{U}$, R, P, $\hat{RH}$, CA, CR, V, A, AR, AP и функциями PA,

UA, AUA, H, Tr, $roles$ и $user$. При этом множества CA, A, V, $\hat{U}$ составляют семантический контекст ролевой модели.

Семантический контекст ролевой модели отражает связь между множествами учётных записей, прав доступа и ролей, организационно-штатной структурой и спецификой деятельности предприятия.

Предположение 6. В рамках данной работы считается, что административные роли AR не имеют семантической составляющей и назначаются всегда классическим способом.

Расширив определение 5 за счет семантического контекста и учёта траектории переходов, введём определение семантической системы ролевого управления доступом.

Определение 15. Семантическая система ролевого управления доступом — это конечный автомат $\Sigma = (\Gamma, Q, a, \Psi)$, состояния которого задаются кортежем $\gamma = \left(S, \hat{U}, UA, user, roles, H, Tr\right) \in \Gamma$, а правила перехода между состояниями определяются предварительными условиями CR, атрибут-условиями CA, функцией ограничения действий H и траекторией Tr.

Предположение 7. Предполагается, что множества $P, R, \hat{RH}, A, V, AR$, а также условия CR и CA не изменяются в процессе функционирования системы.

3. Анализ безопасности семантической системы ролевого управления доступом

3.1. Изменение атрибутов учётной записи и расчёт траектории перехода системы между состояниями

Сформулируем и обоснуем правила, на основании которых определяется порядок следования элементов в траектории Tr.

Утверждение 2. Пусть в результате успешного выполнения действия σ_5 система $\Sigma = \langle \Gamma, Q, a, \Psi \rangle$ перешла в состояние γ_1, где атрибуты учетной записи пользователя $\hat{u} \in \hat{U}$ изменились таким образом, что необходимо отозвать у него атрибут-роли $r_{11}, r_{12}, \ldots, r_{1k} \in R_a$ и назначить ему атрибут-роли $r_{21}, r_{22}, \ldots, r_{2m} \in R_a$, переведя систему в состояние γ_2. Пусть $Tr(\hat{u}, \gamma_1, \gamma_2, H)$ — траектория перехода из состояния γ_1 в состояние γ_2. Переход из состояния γ_1 в состояние γ_2 по траектории Tr возможен только в том случае, если построенная траектория удовлетворяет следующим свойствам:

1) действия σ_7 по отзыву ролей должны следовать в траектории перед действиями по назначению ролей σ_6;

2) если r_i и r_j — такие роли, что $r_i > r_j$, то $\sigma_7(r_j)$ должно следовать в траектории раньше, чем $\sigma_7(r_i)$, и $\sigma_6(r_i)$ должно выполняться раньше, чем $\sigma_6(r_j)$.

Доказательство. Покажем достаточность выполнения условий 1 и 2 для возможности перехода по заданной траектории TR. Невозможность выполнения одного или более действий в траектории перехода может произойти в одном из двух случаев.

Случай 1. Пусть последовательность действий в траектории TR такова, что некоторой учётной записи $\hat{u}$ необходимо автоматически назначить роль r_a, но данное назначение противоречит предварительным условиям CR, так как учётная запись пользователя $\hat{u}$ уже авторизована на роли из множества $\{r_{11}, r_{12}, \ldots, r_{1k}\} \subseteq R_a$, исключающие назначение роли r_a. Данная ситуация противоречит условию 1 утверждения, так как действия по отзыву ролей $r_{11}, r_{12}, \ldots, r_{1k}$ всегда выполняются до выполнения действий по назначению новых ролей.

С л у ч а й 2. Пусть последовательность действий в траектории TR такова, что некоторой учётной записи $\hat{u}$ необходимо автоматически назначить роль r_a, но данное назначение противоречит предварительным условиям CR, так как учётная запись пользователя $\hat{u}$ не авторизована на роли из множества $\{r_{21}, r_{22}, r_{2y-1}, \ldots, r_{2y+1}, \ldots, r_{1m}\} \subseteq R_a$, являющиеся предусловием для назначения роли r_a. По определению иерархии, если роль r_2 является предусловием для роли r_1, то $r_1 > r_2$. Следовательно, данная ситуация противоречит условию 2 утверждения, так как дочерние роли назначаются учётной записи пользователя раньше, чем родительские роли. ■

Замечание. Условия утверждения 2 являются необходимыми, но не достаточными для перехода в состояние γ_2. Невозможность выполнения действий, заданных траекторией, может произойти также в результате некорректного определения предварительных условий, если две или более ролей из множества $\{r_{21}, r_{22}, \ldots, r_{2m}\} \subseteq R_a$ являются взаимоисключающими. Если предварительные условия CR не противоречат атрибут-условиям CA и ни одна из ролей $r_{21}, r_{22}, \ldots, r_{2m} \in R_a$ не исключает других ролей из того же множества, то условия утверждения являются необходимыми и достаточными для перехода в состояние γ_2 по вычисленной траектории Tr.

Утверждение 3. Пусть семантически осмысленная система ролевого управления доступами $\Sigma = \langle \Gamma, Q, a, \Psi \rangle$ содержит только атрибут-роли (т. е. $R = R_a$, $AR_a = AR$) и учётную запись *system*, от имени которой инициируется сессия, выполняющая автоматический перерасчёт назначенных учётной записи пользователя ролей при изменении её атрибутов согласно траектории, удовлетворяющей условиям утверждения 2. Пусть атрибуты учётной записи пользователя полностью описывают его должностные обязанности и все изменения атрибутов передаются в каталог системы ролевого управления доступом. Тогда каждая учётная запись пользователя $\hat{u} \in \hat{U}$ всегда имеет набор ролей, соответствующих его должностным обязанностям.

Доказательство. Пусть должностные обязанности учётной записи пользователя, соответствующей некоторому сотруднику предприятия, характеризуются набором атрибутов $A = a_1, a_2, \ldots, a_N$. Пусть $r_1, r_2 \in R_a$ — атрибут-роли, с которыми связаны различные атрибут-условия ca_{r1} и ca_{r2} соответственно. Пусть первоначально должностные обязанности учётной записи пользователя $\hat{u}$ были таковы, что $ca_{r1}(\hat{u}) = \texttt{true}$ и $ca_{r2}(\hat{u}) = \texttt{false}$. В таком случае учётной записи пользователя назначена только роль r_1 и соответствующие ей права доступа. Пусть после смены должности атрибуты учётной записи пользователя изменяются таким образом, что $ca_{r1}(\hat{u}) = \texttt{false}$ и $ca_{r2}(\hat{u}) = \texttt{true}$. В этом случае, согласно правилам, приведённым в таблице, и определению атрибут-роли, в системе инициируется автоматический отзыв роли r_1 и автоматическое назначение роли r_2. Поскольку, по определению модели $RBAC$, права доступа назначаются учётной записи пользователя только посредством ролей, она будет обладать только правами доступа, соответствующими новой должности связанного с ней сотрудника. ■

3.2. А н а л и з д е й с т в и й п о п е р е х о д у м е ж д у с о с т о я н и я м и

Проведем анализ действия *change_user_attr*, выполняемого сессией от имени учётной записи *attribute_source*. Выполнение действия $change\_user\_attr(u, i, v)$ для заданного состояния системы является успешным только в том случае, если выполняются следующие условия:

— значение некоторого атрибута учётной записи пользователя, соответствующего сотруднику в системе учёта кадров предприятия, не совпадает со значением соот-

ветствующего атрибута учётной записи связанного пользователя $\hat{u} \in \hat{U}$ в каталоге учётных записей системы ролевого управления доступом;

— в данный момент времени над учётной записью пользователя не выполняется никаких действий ($Tr(\hat{u}, \gamma_1, \gamma_2, H) = \varnothing$). В случае успешного выполнения действия над заданным состоянием системы $\gamma_1 = (\hat{U}_1, UA_1, user_1, roles_1)$ новое состояние $\gamma_1 = \left(\hat{U}_2, UA_2, user_2, roles_2\right)$ отличается от исходного множеством значений $\hat{U}$.

Результатом успешного выполнения действия по изменению атрибутов учётной записи $\hat{u_1} = (u, ux_1, \ldots, ux_i, \ldots, ux_n) \in \hat{U}$ является состояние, в котором $\hat{U_2} = \left(\hat{U_1} \setminus \{\hat{u_1}\}\right) \cup$ $\cup\{\hat{u_2}\}$, где $\hat{u_2} = (u, ux_1, \ldots, v, \ldots, ux_n)$, $v \neq ux_i$. В случае неуспешного выполнения действия состояние системы не изменится.

Проведем анализ действий $auto\_assign\_role$ и $auto\_revoke\_role$, выполняемых сессией-инициатором $system$. Выполнение действия $auto\_assign\_role\,(\hat{u}, r)$ для заданного состояния системы является успешным только в том случае, если выполняются следующие условия:

— выполняются атрибут-условия назначения роли $ca_r(\hat{u}) = \texttt{true}$;

— выполняются предварительные условия CR. Поскольку субъект $system$ авторизован на все возможные административные роли для управления назначениями атрибут-ролей, то значение функции $can\_assign\,(CR, AUA\,(system))$ зависит только от аргумента CR;

— σ_6 является текущим значением траектории ($tr[0] = \sigma_6$).

В случае успешного выполнения действия над заданным состоянием системы $\gamma_1 = (\hat{U_1}$, $UA_1, user_1, roles_1)$ новое состояние $\gamma_2 = \left(\hat{U_2}, UA_2, user_2, roles_2\right)$ отличается от исходного значением функции UA. Результатом успешного выполнения действия является состояние, для которого $UA_2(\hat{u_1}) = UA_1(\hat{u_1}) \cup \{r\}$. Выполненное действие удаляется из траектории. В случае неуспешного выполнения действия состояние системы не изменится.

Рассмотрим действие автоматического отзыва роли $auto\_revoke\_role\,(\hat{u}, r)$, где сессия от имени учётной записи пользователя $system$ отзывает у учётной записи пользователя $\hat{u}$ роль $r \in R_a$. Выполнение данного действия для заданного состояния системы является успешным только в том случае, если:

— учётная запись пользователя $\hat{u}$ авторизована на роль r, т. е. $r \in UA\,(u)$;

— значения атрибутов учётной записи пользователя не соответствуют атрибут-условиям назначения для роли r, т. е. $ca_r\,(\hat{u}) = \texttt{false}$;

— σ_7 является текущим значением траектории ($tr[0] = \sigma_7$).

В случае успешного выполнения действия над заданным состоянием системы $\gamma_1 = (\hat{U_1}$, $UA_1, user_1, roles_1)$ новое состояние $\gamma_2 = \left(\hat{U_2}, UA_2, user_2, roles_2\right)$ отличается от исходного значением функции UA. Результатом успешного выполнения действия является состояние, для которого $UA_2(\hat{u_1}) = UA_1(\hat{u_1}) \setminus (\{r\} \cup up\,(r))$, где $up\,(r)$ — множество ролей, больших r. Выполненное действие удаляется из траектории. В случае неуспешного выполнения действия состояние системы не изменится.

Рассмотрим действие $auto\_recalculate\,(\hat{u})$. Условием выполнения данного действия является переход системы в состояние, где σ_8 является единственным возможным действием: $H(\hat{u}, \gamma, \sigma) = \{\sigma_8\}$, $TR[0] = \sigma_8$. Согласно таблице, такая ситуация возможна в результате успешного выполнения действия σ_5. Результатом выполнения действия

является траектория Tr, состоящая из действий σ_6 и σ_7 в соответствии с количеством ролей, которые необходимо назначить или отозвать.

Проведем анализ действий $\sigma_3 = assign\_role$ и $\sigma_4 = revoke\_role$, выполняемых сессией от имени учетной записи администратора системы $admin \in INIT\{\sigma_3, \sigma_4\}$. Выполнение действия $assign\_role(admin, \hat{u}, r)$ для заданного состояния системы γ_1 является успешным только в том случае, если выполняются следующие условия:

— $r \in R_k$ — классическая роль;
— выполняются предварительные условия $CR : c_r(\hat{u}) = \texttt{true}$;
— $r \in can\_assign(CR, AUA(admin))$;
— $\sigma_3 \in H(\hat{u}, \gamma, \sigma)$ и над учётной записью пользователя в данный момент не выполняется никаких других действий — $Tr(\hat{u}, \gamma_1, \gamma_2, H) = \varnothing$.

В случае успешного выполнения действия над заданным состоянием системы $\gamma_1 = (\hat{U_1}, UA_1, user_1, roles_1)$ новое состояние $\gamma_2 = \left(\hat{U_2}, UA_2, user_2, roles_2\right)$ отличается от исходного значением функции UA. Результатом успешного выполнения действия является состояние, для которого $UA_2(\hat{u}) = UA_1(\hat{u}) \cup \{r\}$. В случае неуспешного выполнения действия состояние системы не изменится.

Аналогичные условия выполняются для действия классического отзыва роли $revoke\_role(admin, \hat{u}, r)$, где сессия от имени учётной записи администратора $admin$ отзывает у учётной записи пользователя $\hat{u}$ роль $r \in R_k$. Выполнение данного действия для заданного состояния системы является успешным только в том случае, если:

— $r \in R_k$ — классическая роль;
— $\{r\} \cup up(r) \subseteq can\_revoke(AUA(admin))$, где $up(r)$ — множество ролей, старших r;
— $\sigma_4 \in H(\hat{u}, \gamma, \sigma)$ и над учётной записью пользователя в данный момент не выполняется никаких других действий — $Tr(\hat{u}, \gamma_1, \gamma_2, H) = \varnothing$.

В случае успешного выполнения действия над заданным состоянием системы $\gamma_1 = (\hat{U_1}, UA_1, user_1, roles_1)$ новое состояние $\gamma_2 = \left(\hat{U_2}, UA_2, user_2, roles_2\right)$ отличается от исходного значением функции UA. Результатом успешного выполнения действия является состояние, для которого $UA_2(\hat{u}) = UA_1(\hat{u}) \setminus (\{r\} \cup up(r))$. Выполненное действие удаляется из траектории. В случае неуспешного выполнения действия состояние системы не изменится.

3.3. Критерий безопасности функционирования системы

Сформулируем критерий безопасного функционирования системы семантического ролевого управления доступом и покажем, что в случае, когда управление назначениями атрибут-ролей осуществляется автоматически от лица единственной доверенной учётной записи $system \in INIT(\sigma)$, а классические роли управляются исключительно сессиями, функционирующими от имени учётных записей пользователей из подмножества администраторов $U_a \subseteq INIT(\sigma)$, $U_a \cap \{system\} = \varnothing$, то система является безопасной с точки зрения данного критерия.

Определение 16. Состояние $\gamma \in \Gamma$ системы ролевого управления доступом $\Sigma = \langle \Gamma, Q, a, \Psi \rangle$ называется безопасным, если каждой учётной записи пользователя $\hat{u} \in \hat{U}$ назначены только роли, не противоречащие условиями CR и CA.

Определение 17. Система $\Sigma = \langle \Gamma, Q, a, \Psi \rangle$ называется безопасной, если любое состояние γ_g, достижимое из начального безопасного состояния γ посредством действий, инициированных сессией от имени любой из учётных записей пользователей $\hat{u} \in INIT(\Psi)$, является безопасным.

Теорема 1. Пусть семантически осмысленная система ролевого управления доступом $\Sigma = \langle \Gamma, Q, a, \Psi \rangle$ обладает следующими свойствами:

1) все действия по назначению и отзыву атрибут-ролей $R_a \in R$ инициируются автоматически в порядке, определённом траекторией Tr;
2) множество значений функции $INIT(\sigma_5)$ включает в себя ровно одну учётную запись пользователя $attribute\_source$, т. е. $INIT(\sigma_5) = \{attribute\_source\}$;
3) множество значений функции $INIT(\sigma_6, \sigma_7)$ включает в себя ровно одну учётную запись пользователя $system$, т. е. $INIT(\sigma_6, \sigma_7) = \{system\}$;
4) $INIT(\sigma_3, \sigma_4) = U_a$, при этом $INIT(\sigma_3, \sigma_4) \cap INIT(\sigma_6, \sigma_7) = \varnothing$.

Тогда, если начальное состояние ролевой модели $\gamma_0 = \left(\hat{U}_0, UA_0, user_0, roles_0\right)$ является безопасным, то система Σ является безопасной.

Доказательство. Пусть начальное состояние ролевой модели γ_0 является безопасным. Поскольку при выполнении действий σ_3 и σ_6 по назначению роли учётной записи пользователя осуществляется проверка того, что новое назначение не противоречит предварительным условиям CR, нарушение свойства безопасности системы при переходе из γ_0 в новое состояние γ_g может произойти только в следующих случаях.

С л у ч а й 1. Невозможность автоматического назначения роли в результате назначения взаимоисключающей роли администратором системы: $\sigma_6(\sigma_3(\sigma_5(\gamma_0))) = \gamma_g$. Пусть атрибуты учётной записи пользователя в системе были изменены в результате действия σ_5 и сессией администратора $u_a \in U_a$ инициируется выполнение действия σ_3 по назначению классической роли r_k до того, как сессия $system$ инициирует действие σ_6 по назначению атрибут-роли r_a. После инициализации σ_6 может возникнуть ситуация, когда атрибут-роль r_a должна быть назначена учётной записи пользователя по условиям CA, но не может быть назначена по условиям CR, так как роль r_k является взаимоисключающей с r_a. Данная ситуация противоречит условию 1 теоремы, так как любые другие действия над учётной записью пользователя в системе запрещены до завершения цепочки действий σ_7 и σ_6, выполняемых согласно траектории, рассчитанной при выполнении σ_8. Действие σ_8 является единственным разрешённым действием после успешного выполнения σ_5, и до завершения σ_8 любые другие действия над заданной учётной записью запрещены.

С л у ч а й 2. Невозможность автоматического назначения роли в результате отзыва администратором системы роли, входящей в предусловие: $\sigma_7(\sigma_4(\sigma_5(\gamma_0))) = \gamma_g$. Пусть атрибуты учётной записи пользователя в системе были изменены в результате действия σ_5 и сессия администратора $u_a \in U_a$ инициирует выполнение действия σ_4 по отзыву классической роли r_k до того, как сессия $system$ инициирует действие σ_6 по назначению атрибут-роли r_a. После инициализации σ_4 может возникнуть ситуация, когда роль r_k должна быть назначена учётной записи пользователя по условиям CA, но не может быть назначена ей по условиям CR, так как авторизация на роль r_k является обязательным предусловием назначения r_a. Аналогично случаю 1, данная ситуация противоречит условию 1 теоремы.

С л у ч а й 3. После завершения перерасчёта ролей сессия администратора отзывает роль, дочернюю для автоматически назначенной роли $\sigma_4(\sigma_6(\sigma_5(\gamma_0))) = \gamma_g$. Пусть атрибуты учётной записи пользователя в системе изменены в результате действия σ_5 и атрибут-роль r_a назначена автоматически действием σ_6, но на следующем шаге сессия администратора $u_a \in U_a$ инициирует выполнение действия σ_4 по отзыву классической роли r_k, которая является дочерней (необходимым предусловием) для роли r_a. После инициализации σ_4 может возникнуть ситуация, когда атрибут-роль ra должна быть на-

значена учётной записи пользователя по условиям CA, но не может быть назначена по условиям CR, так как роль r_k является предварительным условием для r_a. Данная ситуация противоречит условиям 3 и 4 теоремы. Если классическая роль r_k обязательно назначается всем учётным записям пользователей, обладающим ролью r_a, то это означает, что $r_k > r_a$. Следовательно, $r_k \in can\_assign\,(CR, AUA\,(u_a))$ и, по определению функции $INIT(\sigma)$, $u_a \in INIT\,(\sigma_6, \sigma_7)$. Но тогда либо $INIT\,(\sigma_3, \sigma_4) \cap INIT\,(\sigma_6, \sigma_7) \neq \varnothing$, либо роль r_k имеет атрибут-условие назначения и не является классической. Противоречие.

С л у ч а й 4. Наличие двух и более администраторов атрибут-ролей.

Пусть $\sigma_7\,(\sigma_6\,(\sigma_5\,(\gamma_0))) = \gamma_g$ и права по администрированию атрибут-ролей назначены одновременно двум и более учётным записям пользователей $system_1$ и $system_2$. Атрибуты учётной записи пользователя в системе были изменены в результате действия σ_5, и сессия $system_1 \in INIT\,(\sigma_6, \sigma_7)$ инициирует выполнение действия σ_6 по назначению атрибут-роли ra_1 до того, как сессия $system_2 \in INIT\,(\sigma_6, \sigma_7)$ инициирует действие σ_7 по отзыву атрибут-роли ra_2, являющейся взаимоисключающей для ra_1. После инициализации σ_6 возможна ситуация, когда роль ra_1 может быть назначена учётной записи пользователя по условиям CA, но её назначение блокируется условиями CR, так как роль ra_2 является взаимоисключающей с ra_1. Данный случай противоречит требованию 3 теоремы о единственности субъекта $system$. ■

Утверждение 4. Необходимым условием для выполнения свойства 4 в теореме 1 является то, что если некоторая роль $r_a \in R$ является атрибут-ролью, то все её старшие роли также являются атрибут-ролями.

Доказательство. От противного. Если атрибут-роль r_a обязательно назначается всем учётным записям пользователей, авторизованным на классическую роль $r_k \in R$, то это означает, что $r_k > r_a$. По предположению 2 в этом случае для некоторой административной роли $r \in AR$, такой, что $r_a \in can\_revoke\,(r)$, выполняется условие $r_k \in can\_revoke\,(r)$. Следовательно, либо $INIT\,(\sigma_3, \sigma_4) \cap INIT\,(\sigma_6, \sigma_7) \neq \varnothing$, либо роль r_k имеет атрибут-условие назначения и не является классической. Получили противоречие. ■

Замечание Свойство распространения атрибут-условий вверх по иерархии доказано в утверждении 1. Поскольку по предположению 7 на протяжении функционирования системы ролевая иерархия $\hat{RH}$ остаётся неизменной, то не может возникнуть ситуации, когда классическая роль включена в иерархию атрибут-ролей администратором системы. По определению множества $INIT(\sigma)$, свойство 3 в теореме эквивалентно требованию $AR = AR_a \cup AR_k$, $AR_a \cap AR_k = \varnothing$, где AR_a — административные роли для атрибут-ролей, AR_k — административные роли для классических ролей. Следовательно, достаточным условием выполнения свойства 3 в теореме 1 является введение ограничения статического взаимного исключения ролей из подмножеств AR_a и AR_k.

Заключение

Дополнение классической модели $RBAC$ за счет семантического контекста позволяет адаптировать её к условиям функционирования реальных АИС предприятий, где каждой учётной записи пользователя АИС соответствует сотрудник предприятия с определённым набором должностных обязанностей, а также автоматизировать операции назначения и отзыва ролей. Ряд элементов классической модели и правила преобразования состояний переопределены для возможности разделения операций по назначению ролей на автоматические и выполняющиеся сессиями, функционирующи-

ми от имени администраторов системы. Кроме того, в семантической модели вводится определение иерархии ролей, отличное от классического, не требующее обязательного включения прав доступа дочерних ролей в родительские, так как это свойство не всегда выполняется в реальных системах. Безопасность системы ролевого управления доступом, построенной на основе предложенной модели, подтверждается проведённым анализом переходов между состояниями.

ЛИТЕРАТУРА

1. `http://csrc.nist.gov/rbac/rbacSTD-ACM.pdf` — National Institute of Standards and Technology, Proposed Standard for Role-Based Access Control.
2. *Ferraiolo D., Kuhn D., and Chandramoul R.* Role Based Access Control. NewYork: ARTECH HOUSE, INC, 2003. 337 с.
3. *Девянин П. Н.* Модели безопасности компьютерных систем. Управление доступом и информационными потоками: учеб. пособие для вузов. М: Горячая линия — Телеком, 2011. 320 с.
4. *Sandhu R. S., Bhamidipati V., and Munawer Q.* The ARBAC97 model for role-based aministration of roles // ACM Trans. Information and Systems Security. No. 2(1). NewYork: ACM Publishing, 1999. P. 105–135.
5. *Нокс Д.* Создание эффективной системы безопасности для Oracle Database 10g. М: Лори, 2007. 576 с.
6. *Джоханссон Дж.* Обеспечение безопасности. Ресурсы Windows Server 2008. СПб.: Русская редакция, 2009. 544 с.