МАТЕМАТИЧЕСКИЕ ОСНОВЫ КОМПЬЮТЕРНОЙ БЕЗОПАСНОСТИ

DOI 10.17223/20710410/11/6

УДК 004.94

ПРАВИЛА ПРЕОБРАЗОВАНИЯ СОСТОЯНИЙ БАЗОВОЙ РОЛЕВОЙ ДП-МОДЕЛИ УПРАВЛЕНИЯ ДОСТУПОМ И ИНФОРМАЦИОННЫМИ ПОТОКАМИ В ОПЕРАЦИОННЫХ СИСТЕМАХ[1]

П. Н. Девянин

Институт криптографии, связи и информатики, г. Москва, Россия

E-mail: peter_devyanin@hotmail.com

Рассматривается базовая ролевая ДП-модель управления доступом и информационными потоками *в операционных системах*, в которую по сравнению с базовой ролевой ДП-моделью включены учетные записи пользователей, сущности, параметрически ассоциированные с субъект-сессиями или ролями, мандатный контроль целостности, фактические доступы субъект-сессий. Основное внимание уделено изменениям в условиях и результатах применения правил преобразования состояний. Обосновывается утверждение о возможности использования только монотонных правил преобразования состояний для анализа условий передачи прав доступа ролей, получения доступов или реализации информационных потоков.

Ключевые слова: *компьютерная безопасность, ролевая ДП-модель, операционная система.*

1. Описание модели

Существующие дискреционные, мандатные и ролевые ДП-модели [1, 2] предоставляют достаточно развитые механизмы моделирования управления доступом и информационными потоками в компьютерных системах. В то же время современные операционные системы (ОС) обладают существенными особенностями (например, мандатным контролем целостности в ОС семейства *Microsoft Windows*), обуславливающими необходимость адаптации существующих формальных моделей для обеспечения условий анализа безопасности ОС, а реализуемые в ОС механизмы защиты достаточно сложны, что затрудняет непосредственно разработку формальной модели, детально им соответствующей. В связи с этим в работе строится ролевая ДП-модель, которую целесообразно считать промежуточным шагом на пути формирования ролевой модели управления доступом и информационными потоками в ОС, направленным на совершенствование техники формального описания данного механизма защиты. Она основана на базовой ролевой ДП-модели (БР ДП-модели) [2], ДП-модели с функционально или параметрически ассоциированными с субъектами сущностями (ФПАС ДП-модели) [3], при этом для обеспечения возможности теоретического анализа условий обеспечения целостности программной среды ОС в ней использованы элементы модели мандатной политики целостности информации Биба и модели мандатного ролевого управления доступом [2]. Построенную ДП-модель будем называть базовой ро-

[1]Работа выполнена при поддержке гранта МД-2.2010.10.

левой ДП-моделью управления доступом и информационными потоками в ОС (или, сокращенно, БРОС ДП-моделью).

С учетом специфики реализации управления доступом и информационными потоками в ОС сделаем предположения.

Предположение 1. В рамках БРОС ДП-модели пользователям соответствуют их учетные записи, для каждой из которых задается множество авторизованных ролей. Каждая учетная запись пользователя вне зависимости от имеющихся у нее авторизованных ролей является учетной записью либо доверенного, либо недоверенного пользователя. Каждая субъект-сессия является либо доверенной, либо недоверенной и функционирует от имени учетной записи доверенного или недоверенного пользователя соответственно. Доверенные субъект-сессии не инициируют создание субъект-сессий. Каждая недоверенная субъект-сессия может создать недоверенную субъект-сессию.

Предположение 2. Для каждой учетной записи пользователя задается множество сущностей, параметрически с ней ассоциированных, реализация от каждой из которых информационного потока по памяти к субъект-сессии позволяет ей создать субъект-сессию от имени данной учетной записи пользователя. Множество сущностей, параметрически ассоциированных с учетной записью пользователя, не содержит субъект-сессий и не изменяется в процессе функционирования системы.

Предположение 3. Функционально ассоциированными с субъект-сессией являются сущности, от которых зависит вид преобразования данных, реализуемого субъект-сессией. Только информационный поток по памяти к сущности, функционально ассоциированной с субъект-сессией, приводит к изменению вида преобразования данных, реализуемого этой субъект-сессией. Множество сущностей, функционально ассоциированных с субъект-сессией, задается при её создании и не изменяется в процессе функционирования системы. При создании новой субъект-сессии множество функционально ассоциированных с ней сущностей задается только в зависимости от сущности, из которой создается данная субъект-сессия, и учетной записи пользователя, от имени которой другая субъект-сессия создает данную субъект-сессию.

Предположение 4. Параметрически ассоциированными с субъект-сессией являются сущности, которые содержат данные, позволяющие идентифицировать вид преобразования данных, реализуемого субъект-сессией. Множество сущностей, параметрически ассоциированных с субъект-сессией, не содержит субъект-сессий и не изменяется в процессе функционирования системы.

Предположение 5. Для каждой роли или административной роли задается (возможно, пустое) множество сущностей, параметрически с ней ассоциированных и не являющихся субъект-сессиями. При этом для получения или удаления роли из множества текущих ролей субъект-сессии кроме наличия данной роли во множестве авторизованных ролей учетной записи пользователя, от имени которой функционирует субъект-сессия, необходимо реализовать к себе информационные потоки по памяти от всех сущностей, параметрически ассоциированных с данной ролью.

Заметим, что в ДП-модели файловой системы (ФС ДП-модели) [4] для анализа условий создания новых субъектов, выполняющих функции защиты файловой системы (например, криптографической защиты или кодирования данных), рассматривались потенциальные доверенные субъекты. При этом реализация информационных потоков по памяти от сущностей, параметрически ассоциированных с ними, позволяла недоверенным субъектам создать доверенные субъекты. В рамках БРОС ДП-модели вместо потенциальных доверенных субъектов анализируются учетные записи пользователей, параметрически ассоциированные с ними сущности и сущности, параметриче-

ски ассоциированные с ролями. Кроме того, сущности, параметрически ассоциированные с учетной записью пользователя и использованные для создания субъект-сессии, не обязательно становятся параметрически ассоциированными с субъект-сессией. Например, для создания субъект-сессии на удаленном сетевом ресурсе могут потребоваться сущность «имя пользователя» и сущность «пароль». При этом параметрически ассоциированной с созданной субъект-сессией сущностью может стать только сущность «идентификатор удаленной сессии», «знание» которой (реализация от нее информационного потока по памяти) может позволить получить контроль над созданной субъект-сессией.

Введём следующие определения и обозначения:

$E = O \cup C$ — множество сущностей, где O — множество объектов, C — множество контейнеров и $O \cap C = \varnothing$;

U — множество учетных записей пользователей;

$]u[\subset E \setminus S$ — множество сущностей, параметрически ассоциированных с учетной записью пользователя $u \in U$;

$UE = \{e \in]u[\colon u \in U\}$ — множество сущностей, каждая из которых параметрически ассоциирована хотя бы с одной учетной записью пользователя;

L_U — множество учетных записей доверенных пользователей;

N_U — множество учетных записей недоверенных пользователей, при этом по определению справедливы равенства $L_U \cup N_U = U$, $L_U \cap N_U = \varnothing$;

$S \subseteq E$ — множество субъект-сессий учетных записей пользователей;

L_S — множество доверенных субъект-сессий;

N_S — множество недоверенных субъект-сессий, при этом по определению справедливо равенство $L_S \cap N_S = \varnothing$;

R — множество ролей;

AR — множество административных ролей ($AR \cap R = \varnothing$);

$]r[\subset E \setminus S$ — множество сущностей, параметрически ассоциированных с ролью или административной ролью $r \in R \cup AR$;

$RE = \{e \in]r[\colon r \in R \cup AR\}$ — множество сущностей, параметрически ассоциированных со всеми ролями;

$R_r = \{read_r, write_r, append_r, execute_r, own_r\}$ — множество видов прав доступа;

$R_a = \{read_a, write_a, append_a, own_a\}$ — множество видов доступа;

$R_f = \{write_m, write_t\}$ — множество видов информационных потоков;

$P \subseteq E \times R_r$ — множеств прав доступа к сущностям;

$A \subseteq S \times E \times R_a$ — множество доступов субъект-сессий к сущностям;

$F \subseteq E \times E \times R_f$ — множество информационных потоков;

$UA : U \to 2^R$ — функция авторизованных ролей учетных записей пользователей;

$AUA : U \to 2^{AR}$ — функция авторизованных административных ролей учетных записей пользователей;

$PA : R \to 2^P$ — функция прав доступа ролей;

$user : S \to U$ — функция принадлежности субъект-сессии учетной записи пользователя;

$roles : S \to 2^R \cup 2^{AR}$ — функция текущих ролей субъект-сессий;

$can_manage_rights : AR \to 2^R$ — функция администрирования прав доступа ролей;

$[s] \subset E \cup U$ — множество сущностей, функционально ассоциированных с субъект-сессией s (при этом по определению выполняется условие $s \in [s]$), и учетных записей

пользователей, от имени каждой из которых может быть создана субъект-сессия, являющаяся функционально ассоциированной с субъект-сессией s сущностью;

$fa : U \times E \to 2^E \cup 2^U$ — функция, задающая множества сущностей, функционально ассоциированных с субъект-сессией при ее создании от имени учетной записи пользователя из сущности;

$]s[\subset E \setminus S$ — множество сущностей, параметрически ассоциированных с субъект-сессией $s \in S$;

$fp : U \times E \to 2^E$ — функция, задающая множества сущностей, параметрически ассоциированных с субъект-сессией при ее создании из сущности от имени учетной записи пользователя. По определению выполняется условие: для каждой субъект-сессии $s \in S$ существует единственная сущность $e_s \in E$, такая, что справедливы равенства $fp(user(s), e_s) =]s[$ и $fa(user(s), e_s) = [s]$;

$H_E : E \to 2^E$ — функция иерархии сущностей;

$H_R : R \to 2^R$ — функция иерархии ролей;

$H_{AR} : AR \to 2^{AR}$ — функция иерархии административных ролей.

Определение 1. Доверенную субъект-сессию y назовем функционально корректной, когда во множество функционально ассоциированных с ней сущностей $[y]$ не входят недоверенные субъект-сессии.

Определение 2. Доверенную субъект-сессию y назовем функционально корректной относительно доверенной субъект-сессии y' и сущности e, когда субъект-сессия y не реализует информационный поток по памяти от сущности e к некоторой сущности e', функционально ассоциированной с доверенной субъект-сессией y'. При этом используем обозначение $y_f(E) \subset L_S \times E$ — множество пар вида (доверенная субъект-сессия, сущность), относительно которых функционально корректна доверенная субъект-сессия y.

Определение 3. Доверенную субъект-сессию y назовем параметрически корректной относительно доверенной субъект-сессии y' и сущности e, когда субъект-сессия y не реализует информационный поток по памяти к сущности e от некоторой сущности e', параметрически ассоциированной с доверенной субъект-сессией y'. При этом используем обозначение $y_p(E) \subset L_S \times E$ — множество пар вида (доверенная субъект-сессия, сущность), относительно которых параметрически корректна доверенная субъект-сессия y.

Определение 4. Доверенную субъект-сессию назовем корректной относительно информационных потоков по времени, когда она не участвует в их реализации. При этом используем обозначения: $LF_S \subset L_S$ — множество доверенных субъект-сессий, корректных относительно информационных потоков по времени; $NF_S \subset L_S$ — множество доверенных субъект-сессий, некорректных относительно информационных потоков по времени, при этом по определению справедливы равенства $LF_S \cap NF_S = \varnothing$, $LF_S \cup NF_S = L_S$.

В ОС семейств *Microsoft Windows Vista*/7/2008 реализуется механизм мандатного контроля целостности (*MIC — Mandatory Integrity Control*), предназначенный для обеспечения контроля целостности программной среды ОС. Данный механизм является перспективным и эффективным средством повышения безопасности управления доступом и информационными потоками в ОС. Таким образом, целесообразно, помимо ролевого управления доступом, при построении БРОС ДП-модели включить в нее элементы, позволяющие анализировать механизм мандатного контроля целостности. При этом без ограничения общности можно рассматривать только два уровня целостности,

предоставляющие возможность обеспечить целостность доверенных субъект-сессий системы. Также имеет смысл учесть наличие в ОС специального механизма защиты от несанкционированного повышения уровней целостности процессов (например, механизм *User Account Control — UAC* в ОС семейств *Microsoft Windows Vista*/7/2008). Данный механизм предусматривает выполнение специальных процедур для активизации процессов или получения доступа к сущностям с высоким уровнем целостности.

Используем следующие обозначения:

$(LI, \leqslant)$ — линейная шкала двух уровней целостности данных, где $LI = \{i_low, i_high\}$, $i_low < i_high$;

$(i_u, i_e, i_r, i_s) \in I$ — четверка функций уровней целостности, при этом:

$i_u : U \to LI$ — функция, задающая для каждой учетной записи пользователя максимальный разрешенный уровень целостности субъект-сессий, функционирующих от ее имени;

$i_e : E \setminus S \to LI$ — функция, задающая уровень целостности для каждой сущности, не являющейся субъект-сессией;

$i_r : R \cup AR \to LI$ — функция, задающая для каждой роли или административной роли ее уровень целостности;

$i_s : S \to LI$ — функция, задающая для каждой субъект-сессии ее текущий уровень целостности;

I — множества всех четверок функций заданного вида.

По аналогии с понятием параметрической корректности доверенных субъект-сессий дадим следующее определение.

Определение 5. Доверенную субъект-сессию y назовем корректной в смысле целостности относительно сущности e, обладающей высоким уровнем целостности i_high, если субъект-сессия y не реализует информационный поток по памяти к сущности e от некоторой сущности e', обладающей низким уровнем целостности i_low. При этом используем обозначение $y_i(E) \subset E$ — множество сущностей, относительно которых корректна в смысле целостности доверенная субъект-сессия y.

В рамках БРОС ДП-модели определим состояние системы.

Определение 6. Пусть определены:

- множества учетных записей пользователей U, сущностей E, субъект-сессий S, прав доступа к сущностям P, учетных записей доверенных пользователей L_U, доверенных субъект-сессий L_S, доступов субъект-сессий к сущностям A, информационных потоков F;
- функции авторизованных ролей учетных записей пользователей UA, авторизованных административных ролей учетных записей пользователей AUA, прав доступа ролей PA, принадлежности субъект-сессий учетным записям пользователей $user$, текущих ролей субъект-сессий $roles$, уровней целостности (i_u, i_e, i_r, i_s), иерархии ролей H_R, иерархии административных ролей H_{AR}, иерархии сущностей H_E.

Определим $G = (UA, AUA, PA, user, roles, (i_u, i_e, i_r, i_s), A, F, H_R, H_{AR}, H_E, L_U, L_S)$ — состояние системы.

Заметим, что, так же как в БР ДП-модели, механизм статических и динамических ограничений рассматриваться не будет.

Используем обозначения:

$\Sigma(G^*, OP)$ — система, при этом G^* — множество всех возможных состояний, OP — множество правил преобразования состояний, заданных в таблице;

$G \vdash_{op} G'$ — переход системы $\Sigma(G^*, OP)$ из состояния G в состояние G' с использованием правила преобразования состояний $op \in OP$;

$\Sigma(G^*, OP, G_0)$ — система $\Sigma(G^*, OP)$ с начальным состоянием G_0.

В существующих компьютерных системах, в том числе в ОС, параметры системы управления доступом (списки прав доступа к файлам, списки прав доступа или привилегий пользователей или ролей), как правило, достаточно надежно защищены. Возможность изменения этих параметров нарушителем часто является следствием преодоления (взлома) им системы защиты. Кроме того, уровень целостности процессов ОС, например в ОС семейств *Microsoft Windows Vista*/7/2008, устанавливается только при их активизации. При необходимости повышения уровня целостности процесса осуществляется его перезапуск. Таким образом, в первую очередь усилия нарушителя могут быть направлены на несанкционированное повышение им своих полномочий (в том числе, уровня целостности) за счет получения контроля над доверенными процессами ОС. С учетом сказанного в рамках БРОС ДП-модели можно не рассматривать действия по администрированию параметров целостности системы, задаваемых функциями (i_u, i_e, i_r, i_s). Таким образом, по аналогии с БР ДП-моделью используем следующие предположения, при этом дополним предположение 2.

Предположение 6. В рамках БРОС ДП-модели на траекториях функционирования системы не изменяются: значения множеств U, L_U, R, функции UA, AUA, H_R, H_{AR}, i_u, i_r и значения множеств сущностей, параметрически ассоциированных с каждой ролью или административной ролью. Новые значения для функций i_e и i_s задаются только при создании соответствующих новых сущностей или субъект-сессий. Таким образом, будем использовать следующее сокращенное обозначение для состояния системы: $G = (PA, user, roles, A, F, H_E)$.

Предположение 7. Уровень целостности роли не превосходит уровней целостности ролей, которым она подчинена в иерархии ролей. Уровень целостности сущности, входящей в состав сущности-контейнера и не являющейся субъект-сессией, не превосходит уровня целостности сущности-контейнера. Уровни целостности сущностей, параметрически ассоциированных с учетной записью пользователя, совпадают с ее уровнем целостности. Текущий уровень целостности субъект-сессии не превосходит уровня целостности учетной записи пользователя, от имени которой она функционирует, и текущего уровня субъект-сессии, которой она подчинена в иерархии. Уровень целостности роли не может быть выше уровня целостности учетной записи пользователя, которая на нее может быть авторизована, и текущего уровня целостности субъект-сессии, во множество текущих ролей которой она входит. Права доступа владения own_r, на запись $write_r$ и запись в конец $append_r$ к сущности, не являющейся субъект-сессией, могут принадлежать только ролям, имеющим уровень целостности не ниже, чем уровень целостности сущности. Право доступа владения own_r к субъект-сессии может принадлежать только ролям, имеющим уровень целостности не ниже, чем уровень целостности субъект-сессии. Таким образом, выполняются условия:

- для ролей $r, r' \in R \cup AR$: если $r \leqslant r'$, то $i_r(r) \leqslant i_r(r')$;
- для сущностей $e, e' \in E \setminus S$: если $e \leqslant e'$, то $i_e(e) \leqslant i_e(e')$;
- для субъект-сессий $s, s' \in S$: если $s \leqslant s'$, то $i_s(s) \leqslant i_s(s')$;
- для каждой сущности $e \in]u[$, где $u \in U$, справедливо равенство $i_e(e) = i_u(u)$;
- для субъект-сессии $s \in S$ верно неравенство $i_s(s) \leqslant i_u(user(s))$;
- для учетной записи пользователя $u \in U$ и роли $r \in R$: если $r \in UA(u)$, то $i_r(r) \leqslant i_u(u)$;
- для субъект-сессии $s \in S$ и роли $r \in R$: если $r \in roles(s)$, то $i_r(r) \leqslant i_s(s)$;

— для права доступа к сущности $(e, \alpha) \in P$, где $\alpha \in \{own_r, write_r, append_r\}$, и роли $r \in R$: если $(e, \alpha) \in PA(r)$, то или $e \in E \setminus S$ и $i_e(e) \leqslant i_r(r)$, или $e \in S$, $\alpha = own_r$ и $i_s(e) \leqslant i_r(r)$.

Предположение 8. Учетные записи доверенных субъект-сессий имеют высокий уровень целостности i_high, а недоверенных субъект-сессий — низкий i_low. Во множестве сущностей имеется сущность $i_entity \in E$, обладающая высоким уровнем целостности, реализация информационного потока по памяти к которой от субъект-сессии является необходимым (в дополнение, в том числе, к требованиям предположения 2) в следующих случаях:

— когда субъект-сессия создает новую субъект-сессию с высоким уровнем целостности;
— когда субъект-сессия берет во множество текущих ролей роль с высоким уровнем целостности;
— когда субъект-сессия создает сущность с высоким уровнем целостности;
— когда субъект-сессия меняет у роли права доступа к сущности с высоким уровнем целостности.

Таким образом, выполняются условия:

— для учетной записи доверенного пользователя $u \in L_U$ верно равенство $i_u(u) = i_high$;
— для учетной записи недоверенного пользователя $u \in N_U$ верно равенство $i_u(u) = i_low$;
— верно равенство $i_e(i_entity) = i_high$.

Предположение 9. Субъект-сессии могут иметь друг к другу только доступ владения own_a. Роли могут обладать к субъект-сессиям только правом доступа владения own_r. Таким образом, для каждой роли $r \in R$ во множестве ее прав доступа $PA(r)$ отсутствуют права доступа вида (s, α), где $s \in S$ и $\alpha \neq own_r$.

Предположение 10. Если субъект-сессия s реализовала информационный поток по памяти от себя к сущности, функционально ассоциированной с другой субъект-сессией s', или субъект-сессия s реализовала информационный поток по памяти к себе от всех сущностей, параметрически ассоциированных с другой субъект-сессией s', то субъект-сессия s получает доступ владения own_a к субъект-сессии s'.

Предположение 11. Если субъект-сессия s имеет доступ владения own_a к субъект-сессии s', то субъект-сессия s получает следующие возможности:

— использовать роли из множества текущих ролей субъект-сессии s';
— изменять множество текущих ролей субъект-сессии s';
— использовать текущий уровень целостности субъект-сессии s';
— использовать доступы субъект-сессии s';
— получать доступ владения own_a к субъект-сессиям, доступом владения к которым обладает субъект-сессия s';
— использовать административные роли субъект-сессии s' для осуществления действий над ролями и сущностями, которые позволяют ей изменять права доступа ролей субъект-сессии s';
— использовать информационные потоки, в реализации которых участвует субъект-сессия s'.

Используем обозначения:

$de_facto_roles : S \to 2^{R \cup AR}$ — функция фактических текущих ролей субъект-сессий, при этом по определению в каждом состоянии системы $G = (PA, user, roles, A,$

F, H_E) для каждой субъект-сессии $s \in S$ верно равенство
$de_facto_roles(s_1) = roles(s) \cup \{r \in R \cup AR$: существует $s' \in S$, такая, что $(s, s', own_a) \in A$ и $r \in roles(s')\}$;

$de_facto_rights : S \to 2^P$ — функция фактических текущих прав доступа субъект-сессий, при этом по определению в каждом состоянии системы $G = (PA, user, roles, A, F, H_E)$ для каждой субъект-сессии $s \in S$ верно равенство
$de_facto_rights(s) = \{p \in P$: существует $r \in de_facto_roles(s)$, такая, что $p \in PA(r)\}$;

$de_facto_accesses : S \to 2^A$ — функция фактических доступов субъект-сессий, при этом по определению в каждом состоянии системы $G = (PA, user, roles, A, F, H_E)$ для каждой субъект-сессии $s \in S$ верно равенство
$de_facto_accesses(s) = \{(s, e, \alpha_a) : (s, e, \alpha_a) \in A\} \cup \{(s', e, \alpha_a) : (s, s', own_a), (s', e, \alpha_a) \in A\}$;

$de_facto_actions : S \to S \times U \times 2^P \times 2^R$ — функция фактических возможных действий субъект-сессий, при этом по определению в каждом состоянии системы $G = (PA, user, roles, A, F, H_E)$ для каждой субъект-сессии $s \in S$ верно равенство
$de_facto_actions(s) = (\{s\} \times \{user(s)\} \times PA(roles(s)) \times can_manage_rights(roles(s) \cap \cap AR)) \cup \{(s', u', PA(roles(s')), can_manage_rights(roles(s') \cap AR)) : (s, s', own_a) \in A$, либо $u' = \varnothing$, либо $u' = user(s')$ и для каждой сущности $e \in]u'[$ выполняется условие $(e, s, write_m) \in F\}$.

Определение 7. Будем говорить, что субъект-сессия s фактически обладает ролью r, когда она принадлежит множеству $de_facto_roles(s)$. При этом роль r будем называть фактической ролью субъект-сессии s. Права доступа из множества $de_facto_rights(s)$ будем называть фактическими текущими правами доступа субъект-сессии s. Доступы из множества $de_facto_accesses(s)$ будем называть фактическими доступами субъект-сессии s.

Заметим, что фактические доступы субъект-сессий в рамках БР ДП-модели не рассматривались.

Определение 8. Будем говорить, что субъект-сессия s обладает фактической возможностью осуществить от имени субъект-сессии s' (с учетной записью пользователя u') действие над сущностью y и ролью r с применением права доступа α_r, когда выполняется условие $(s', u', (y, \alpha_r), r) \in de_facto_actions(s)$. При этом в случае, когда выполняется условие $(s', \varnothing, \varnothing, \varnothing) \in de_facto_actions(s)$, будем говорить, что субъект-сессия s обладает фактическим текущим уровнем целостности $i_s(s')$.

В случае, когда для осуществления субъект-сессией s от имени субъект-сессии s' действия над сущностью y и ролью r с применением права доступа α_r не требуется наличия информационных потоков по памяти от всех сущностей $e \in]user(s')[$, может быть достаточным выполнение условия $(s', \varnothing, (y, \alpha_r), r) \in de_facto_actions(s)$.

2. Правила преобразования состояний

В рамках БРОС ДП-модели используются правила преобразования состояний из множества OP, приведенные в следующей таблице.

Правила преобразования состояний БРОС ДП-модели

Правило	Исходное состояние $G = (PA, user, roles, A, F, H_E)$	Результирующее состояние $G' = (PA', user', roles', A', F', H'_E)$
1	2	3
$take_role(x, w, r)$	$x, w \in S$, $(w, user(w), \varnothing, \varnothing) \in$ $\in de_facto_actions(x)$, $r \in UA(user(w)) \cup AUA(user(w))$, для $e \in]r[$ выполняется условие $(e, x, write_m) \in F$, $i_r(r) \leqslant i_s(w)$, и если $i_r(r) = i_high$, то $(x, i_entity, write_m) \in F$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $A' = A$, $F' = F$, $H'_E = H_E$, $roles'(w) = roles(w) \cup \{r\}$ и для $s \in S \setminus \{w\}$ выполняется равенство $roles'(s) = roles(s)$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, s, write_t)$: $s \in (N_S \cup NF_S) \cap S$, $x \neq s$ и или $s = w$, или $(w, own_a) \in de_facto_accesses(s)\}$, если $x \in LF_S \cap S$, то $F' = F$
$remove_role(x, w, r)$	$x, w \in S$, $r \in roles(w)$, $(w, user(w), \varnothing, \varnothing) \in de_facto_actions(x)$, для $e \in]r[$ выполняется условие $(e, x, write_m) \in F$, и если $i_r(r) = i_high$, то $(x, i_entity, write_m) \in F$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $A' = A$, $F' = F$, $H'_E = H_E$, $roles'(w) = roles(w) \setminus \{r\}$ и для $s \in S \setminus \{w\}$ выполняется равенство $roles'(s) = roles(s)$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, s, write_t)$: $s \in (N_S \cup NF_S) \cap S$, $x \neq s$ и или $s = w$, или $(w, own_a) \in de_facto_accesses(s)\}$, если $x \in LF_S \cap S$, то $F' = F$
$grant_right(x, r, (y, \alpha_r))$	$x \in S$, $y \in E$, $(y, \alpha_r) \in P$, $(w, \varnothing, (y, own_r), r) \in de_facto_actions(x)$, $i_r(r) \leqslant i_s(w)$, если $y \in S$, то $\alpha_r = own_r$ и $i_s(y) \leqslant i_r(r)$; если $y \in E \setminus S$ и $\alpha_r \in \{own_r, write_r, append_r\}$, то $i_e(y) \leqslant i_r(r)$; если $i_e(y) = i_high$, то $(x, i_entity, write_m) \in F$	$S' = S$, $E' = E$, $user' = user$, $roles' = roles$, $A' = A$, $H'_E = H_E$, $PA'(r) = PA(r) \cup \{(y, \alpha_r)\}$, и для $r' \in R \setminus \{r\}$ выполняется равенство $PA'(r') = PA(r')$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, s, write_t) : s \in (N_S \cup NF_S) \cap S$, $x \neq s$ и $r \in de_facto_roles(s)\}$, если $x \in LF_S \cap S$, то $F' = F$
$remove_right(x, r, (y, \alpha_r))$	$x \in S$, $y \in E$, $(y, \alpha_r) \in PA(r)$ и $(w, \varnothing, (y, own_r), r) \in de_facto_actions(x)$, $i_r(r) \leqslant i_s(w)$, и если $i_e(y) = i_high$, то $(x, i_entity, write_m) \in F$	$S' = S$, $E' = E$, $user' = user$, $roles' = roles$, $A' = A$, $H'_E = H_E$, $PA'(r) = PA(r) \setminus \{(y, \alpha_r)\}$, и для $r' \in R \setminus \{r\}$ выполняется равенство $PA'(r') = PA(r')$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, s, write_t)$: $s \in (N_S \cup NF_S) \cap S$, $x \neq s$ и $r \in de_facto_roles(s)\}$, если $x \in LF_S \cap S$, то $F' = F$
$create_entity(x, r, y, yi, z)$	$x \in S$, $y \notin E$, $z \in E \setminus S$, $(w, \varnothing, (z, \alpha_r), r) \in de_facto_actions(x)$, где $\alpha_r \in \{write_r, append_r\}$, $i_r(r) \leqslant i_s(w)$, $i_e(z) \leqslant i_s(w)$, $yi \leqslant i_e(z)$, $yi \leqslant i_r(r)$, если $i_e(z) = i_high$, то $(x, i_entity, write_m) \in F$	$S' = S$, $E' = E \cup \{y\}$, при этом $y \notin UE \cup RE$, $user' = user$, $roles' = roles$, $A' = A$, $i'_e(y) = yi$, $H'_E(z) = H_E(z) \cup \{y\}$, $H'_E(y) = \varnothing$, для $e \in E \setminus \{z\}$ выполняется равенство $H'_E(e) = H_E(e)$, $PA'(r) = PA(r) \cup \{(y, own_r)\}$, и для $r' \in R \setminus \{r\}$ выполняется равенство $PA'(r') = PA(r')$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, e, write_t) : e \in E$ и $y \leqslant e\} \cup$ $\cup \{(x, s, write_t) : s \in (N_S \cup NF_S) \cap S$, $x \neq s$ и $r \in de_facto_roles(s)\}$, если $x \in LF_S \cap S$, то $F' = F$
$rename_entity(x, y, z)$	$x \in S$, $y, z \in E \setminus S$, $y \in H_E(z)$, $(w, \varnothing, (z, \alpha_r), r) \in de_facto_actions(x)$, где $\alpha_r \in \{write_r, append_r\}$, $i_r(r) \leqslant i_s(w)$, $i_e(z) \leqslant$ $\leqslant i_s(w)$, если $i_e(z) = i_high$, то $(x, i_entity, write_m) \in F$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $A' = A$, $H'_E = H_E$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup$ $\cup \{(x, e, write_t) : e \in E, x \neq e, e \leqslant z\} \cup$ $\cup \{(x, s, write_t) : s \in (N_S \cup NF_S) \cap S, x \neq s$, $(e, \alpha_a) \in de_facto_accesses(s)$, где $e \in E$, $e \leqslant y$, $\alpha_a \in R_a\}$, если $x \in LF_S \cap S$, то $F' = F$

Продолжение таблицы

1	2	3
$delete_entity(x, y, z)$	$x \in S$, $y, z \in E \setminus S$, $y \in H_E(z)$, $\{e \in E : e \leqslant y\} \cap (UE \cup RE) = \varnothing$, $(w, \varnothing, (z, write_r), r) \in de_facto_actions(x)$ и $i_r(r) \leqslant i_s(w)$, $i_e(z) \leqslant i_s(w)$, если $i_e(z) = i_high$, то $(x, i_entity, write_m) \in F$	$S' = S$, $E' = E \setminus \{e \in E: e \leqslant y\}$, $user' = user$, $roles' = roles$, $H'_E(z) = H_E(z) \setminus \{y\}$, для всех $e \in E' \setminus \{z\}$ выполняется равенство $H'_E(e) = H_E(e)$, для $r \in R$ выполняется равенство $PA'(r) = PA(r) \setminus \{(e, \alpha_r): e \leqslant y, \alpha_r \in R_r\}$, $A' = A \setminus \{(s, e, \alpha_a): s \in S, e \in E, e \leqslant y, \alpha_a \in R_a\}$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = (F \cup \{(x, e, write_t) : e \in E, x \neq ez \leqslant e\} \cup \cup \{(x, s, write_t) : s \in (N_S \cup NF_S) \cap S', x \neq s$ и $(e, \alpha_a) \in de_facto_accesses(s)$, где $e \in E$, $e \leqslant y$ и $\alpha_a \in R_a\}) \setminus (\{(e, e', \alpha_f): e, e' \in E, e' \leqslant y, \alpha_f \in R_f\} \cup \{(e', e, \alpha_f): e, e' \in E, e' \leqslant y, \alpha_f \in R_f\})$, если $x \in LF_S \cap S$, то $F' = F$
$create_first_session(x, u, r, y, z, zi)$	$x \in S$, $u \in U$, $y \in E$, $z \notin E$, $(y, execute_r) \in PA(UA(u))$ и $r \in can_manage_rights(AUA(u))$, $zi \leqslant i_u(u)$, $zi \leqslant i_r(r)$, $\{(e, x, write_m) : e \in]u[\} \subset F$, если $zi = i_high$, то $(x, i_entity, write_m) \in F$	$S' = S \cup \{z\}$, $E' = E \cup \{z\}$, $A' = A$, $i'_s(z) = zi$, $user'(z) = u$, $roles'(z) = \varnothing$, для $s \in S$ выполняются равенства $user'(s) = user(s)$, $roles'(s) = roles(s)$, $[z] = fa(u, y)$, $]z[= fp(u, y)$, $H'_E(z) = \varnothing$, для $e \in E$ выполняется равенство $H'_E(e) = H_E(e)$, $PA'(r) = PA(r) \cup \{(z, own_r)\}$, и для $r' \in R \setminus \{r\}$ выполняется равенство $PA'(r') = PA(r')$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \cup \{(z, x, write_t), (x, z, write_t)\} \cup \{(x, e, write_t) : e \in E$ и $y \leqslant e\} \cup \{(x, s, write_t) : s \in (N_S \cup \cup NF_S) \cap S$, $x \neq s$ и $r \in de_facto_roles(s)\}$, если $x \in LF_S \cap S$, то $F' = F$
$create_session(x, w, r, y, z, zi)$	$x, w \in S$, $y \in E$, $z \notin E$, $(w, \varnothing, (y, execute_r), r) \in de_facto_actions(x)$, $zi \leqslant i_r(r) \leqslant i_s(w)$	$S' = S \cup \{z\}$, $E' = E \cup \{z\}$, $A' = A$, $i'_s(z) = zi$, $user'(z) = user(w)$, $roles'(z) = \varnothing$, для $s \in S$ выполняются равенства $user'(s) = user(s)$, $roles'(s) = roles(s)$, $[z] = fa(user(w), y)$, $]z[= fp(user(w), y)$, $H'_E(w) = H_E(w) \cup \{z\}$, $H'_E(z) = \varnothing$, для $e \in E \setminus \{w\}$ выполняется равенство $H'_E(e) = = H_E(e)$, $PA'(r) = PA(r) \cup \{(z, own_r)\}$, и для $r' \in R \setminus \{r\}$ выполняется равенство $PA'(r') = PA(r')$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(z, x, write_t), (x, z, write_t)\} \cup \cup \{(x, e, write_t) : e \in E$ и $y \leqslant e\} \cup \{(x, s, write_t) : s \in (N_S \cup NF_S) \cap S, x \neq s$ и $r \in de_facto_roles(s)\}$, если $x \in LF_S \cap S$, то $F' = F$

Продолжение таблицы

1	2	3
$delete_session(x, w, z)$	$x, w, z \in S$, $(w, \varnothing, (z, own_r), \varnothing) \in de_facto_actions(x)$	$S' = S \setminus \{z\}$, $E' = E \setminus \{z\}$, для $s \in S'$ выполняются равенства $user'(s) = user(s)$, $roles'(s) = roles(s)$, для $z' \in S$, такого, что $z \in H_E(z')$, справедливо равенство $H'_E(z') = (H_E(z') \setminus \{z\}) \cup H_E(z)$, при этом выполняется условие: для $e \in E' \setminus \{z'\}$ выполняется равенство $H'_E(e) = H_E(e)$, $PA'(r) = PA(r) \setminus \{(z, own_r)\}$, и для $r' \in R \setminus \{r\}$ выполняется равенство $PA'(r') = PA(r')$, $A' = A \setminus (\{(z, e, \alpha_a)\colon e \in E, \alpha_a \in R_a\} \cup \{(s, z, own_a)\colon s \in S\})$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = (F \cup \cup\{(x, s, write_t) : e \in E, x \neq e \text{ и } z < e\} \cup \cup\{(x, s, write_t) : s \in (N_S \cup NF_S) \cap S, x \neq s \text{ и } (z, own_a) \in de_facto_accesses(s)\}) \setminus \setminus(\{(z, e, \alpha_f) : e \in E, \alpha_f \in R_f\} \cup \{(e, z, \alpha_f)\colon e \in E, \alpha_f \in R_f\})$, если $x \in LF_S \cap S$, то $F' = F$
$control(x, y, z)$	$x, y \in S$, $x \neq y$, $z \in [y]$ и или $x = z$, или $(x, z, write_m) \in F$, или $z \in S$ и $(x, z, own_a) \in A$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $H'_E = H_E$, $A' = A \cup \{(x, y, own_a)\}$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, e, write_t)\colon e \in E, x \neq e \text{ и } y \leqslant e\}$, если $x \in LF_S \cap S$, то $F' = F$
$know(x, y)$	$x, y \in S$, $x \neq y$, и для каждой $e \in]y[$ существует $(e, x, write_m) \in F$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $H'_E = H_E$, $A' = A \cup \{(x, y, own_a)\}$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, e, write_t)\colon e \in E, x \neq e \text{ и } y \leqslant e\}$, если $x \in LF_S \cap S$, то $F' = F$
$take_access_own(x, y, z)$	$x, y, z \in S$, $x \neq z$, $\{(x, y, own_a), (y, z, own_a)\} \subset A$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $H'_E = H_E$, $A' = A \cup \{(x, z, own_a)\}$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, e, write_t)\colon e \in E, x \neq e \text{ и } z \leqslant e\}$, если $x \in LF_S \cap S$, то $F' = F$
$access_own(x, w, y)$	$x, w, y \in S$, $w \neq y$, $(w, \varnothing, (y, own_r), \varnothing) \in de_facto_actions(x)$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $H'_E = H_E$, $A' = A \cup \{(w, y, own_a)\}$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \cup\{(x, e, write_t) : e \in E, x \neq e \text{ и } y \leqslant e\} \cup \cup\{(x, s, write_t) : s \in (N_S \cup NF_S) \cap S, x \neq s \text{ и } (y, own_a) \in de_facto_accesses(s)\}$, если $x \in LF_S \cap S$, то $F' = F$
$access_read(x, w, y)$	$x, w \in S$, $y \in E \setminus S$, $(w, \varnothing, (y, read_r), \varnothing) \in de_facto_actions(x)$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $H'_E = H_E$, $A' = A \cup \{(w, y, read_a)\}$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(y, w, write_m)\} \cup \{(x, e, write_t)\colon e \in E, x \neq e \text{ и } y \leqslant e\}$, если $x \in LF_S \cap S$, то $F' = F \cup \{(y, w, write_m)\}$

Окончание таблицы

1	2	3
$access_write(x, w, y)$	$x, w \in S$, $y \in E \setminus S$, $(w, \varnothing, (y, write_r), \varnothing) \in de_facto_actions(x)$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $H'_E = H_E$, $A' = A \cup \{(w, y, write_a)\}$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(w, y, write_m)\} \cup \{(x, e, write_t)$: $e \in E$, $x \neq e$ и $y \leqslant e\}$, если $x \in LF_S \cap S$, то $F' = F \cup \{(w, y, write_m)\}$
$access_append(x, w, y)$	$x, w \in S$, $y \in E \setminus S$, $(w, \varnothing, (y, append_r), \varnothing) \in de_facto_actions(x)$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $H'_E = H_E$, $A' = A \cup \{(w, y, append_a)\}$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(w, y, write_m)\} \cup \{(x, e, write_t)$: $e \in E$, $x \neq e$ и $y \leqslant e\}$, если $x \in LF_S \cap S$, то $F' = F \cup \{(w, y, write_m)\}$
$flow(x, y, y', z)$	$x, z \in S$, $y, y' \in E$, $x \neq z$, $y \leqslant y'$, [или $x = y$, или $(y, \alpha_a) \in de_facto_accesses(x)$], [или $z = s'$, или $(y', \beta_a) \in \in de_facto_accesses(z)$], где α_a, $\beta_a \in R_a$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $A' = A$, $H'_E = H_E$, если x, $z \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, z, write_t), (z, x, write_t)\}$, если $\{x,z\} \cap LF_S \cap S) \neq \varnothing$, то $F' = F$
$find(x, y, z)$	$x, y \in S$, $z \in E$, $x \neq z$, $(x, y, \alpha) \in F$, где $\alpha \in \{write_m, write_t\}$, и [или $(z, \beta) \in de_facto_accesses(y)$, где $\beta \in \{write_a, append_a\}$], [или $(y, z, \beta) \in F$, где $\beta \in \{write_m, write_t\}$]	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $A' = A$, $H'_E = H_E$, если $write_t \notin \{\alpha, \beta\}$, то $F' = F \cup \{(x, z, write_m)\}$, если $write_t \in \{\alpha, \beta\}$ и $x, y \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, z, write_t)\}$, если $write_t \in \{\alpha, \beta\}$ и $\{x, y\} \cap (LF_S \cap S) \neq \varnothing$, то $F' = F$
$post(x, y, z)$	$x, z \in S$, $y \in E$, $x \neq z$, $(y, read_a) \in de_facto_accesses(z)$ и [или $(y, \alpha) \in de_facto_accesses(x)$, где $\alpha \in \{write_a, append_a\}$], [или $(x, y, \alpha) \in F$, где $\alpha \in \{write_m, write_t\}$]	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $A' = A$, $H'_E = H_E$, если $\alpha \neq write_t$, то $F' = F \cup \{(x, z, write_m)\}$, если $\alpha = write_t$ и x, $z \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, z, write_t)\}$, если $\alpha = write_t$ и $\{x, z\} \cap (LF_S \cap S) \neq \varnothing$, то $F' = F$
$pass(x, y, z)$	$y \in S$, $x, z \in E$, $x \neq z$, $(x, read_a) \in de_facto_accesses(y)$ и [или $(z, \alpha) \in de_facto_accesses(y)$, где $\alpha \in \{write_a, append_a\}$], [или $(y, z, \alpha) \in F$, где $\alpha \in \{write_m, write_t\}$]	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $A' = A$, $H'_E = H_E$, если $\alpha \neq write_t$, то $F' = F \cup \{(x, z, write_m)\}$, если $\alpha = write_t$ и $y \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, z, write_t)\}$, если $\alpha = write_t$ и $y \in LF_S \cap S$ то $F' = F$
$take_flow(x, y)$	$x, y \in S, x \neq y, (x, y, own_a) \in A$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $A' = A$, $H'_E = H_E$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, e, \alpha)$: $(y, e, \alpha) \in F$, $e \in E$, $\alpha \in \{write_m, write_t\}\}$, если $x \in LF_S \cap S$, то $F' = F \cup \{(x, e, write_m)$: $(y, e, write_m) \in F$, $e \in E\}$

Рассмотрим условия и результаты применения правил преобразования состояний БРОС ДП-модели (их зависимость показана на рис. 1) в первую очередь с целью анализа их основных отличий от аналогичных правил БР ДП-модели.

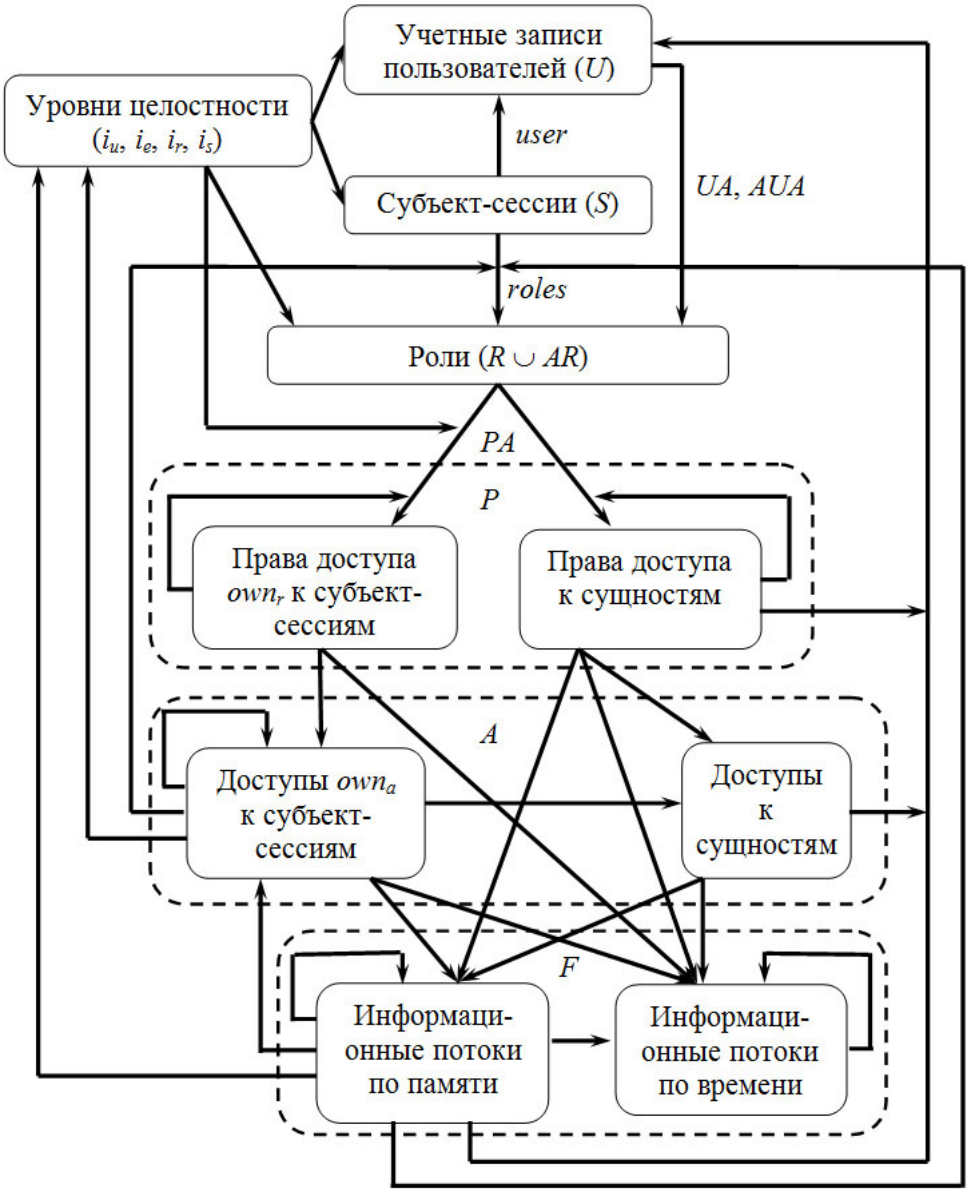

Рис. 1. Зависимость условий и результатов применения правил преобразования состояний БРОС ДП-модели

Правила $take_role(x, w, r)$ и $remove_role(x, w, r)$ позволяют субъект-сессии x добавить (удалить) роль r, на которую субъект-сессия может быть авторизована, либо в множество своих текущих ролей, либо, при наличии доступа владения к другой субъект-сессии w, в множество её текущих ролей. При этом в соответствии с предположением 7 при добавлении роли необходимо, чтобы ее уровень целостности не превосходил текущего уровня целостности субъект-сессии, которая ее получает. Таким образом, если в БР ДП-модели предполагалось, что изменять множество текущих ролей субъект-сессии может только сама субъект-сессия, то в рамках БРОС ДП-модели в соответствии с предположением 11 субъект-сессия может изменять множество своих фактических ролей. Для этого требуется, чтобы субъект-сессия x, получившая контроль над субъект-сессией w, реализовала к себе информационные потоки по памяти от всех сущностей учетной записи пользователя субъект-сессии w; в случае, когда

уровень целостности роли r равняется i_high, в соответствии с предположением 8 требуется реализация субъект-сессией x информационного потока по памяти от себя к сущности i_entity. Также при добавлении или удалении роли дополнительно (по сравнению с БР ДП-моделью) требуется, чтобы субъект-сессия x реализовала к себе информационные потоки от всех сущностей, параметрически ассоциированных с ролью r. Кроме того, так как добавление или удаление роли из множества текущих ролей субъект-сессии может быть использовано для передачи данных, то при реализации правил возможно возникновение информационных потоков по времени от субъект-сессии x к субъект-сессии w и ко всем субъект-сессиям, обладающим к ней фактическим доступом владения.

Правила $grant_right(x, r, (y, \alpha_r))$ и $remove_right(x, r, (y, \alpha_r))$ позволяют субъект-сессии x добавить или удалить соответственно право доступа α_r к сущности y из множества прав доступа роли r. В соответствии с предположением 11 для применения правил необходимо наличие у субъект-сессии x фактической возможности осуществить действие над сущностью y и ролью r с применением права доступа владения own_r. Кроме того, в соответствии с предположением 7 для применения правил требуется, чтобы уровень целостности роли не превосходил текущего уровня целостности субъект-сессии, от имени которой добавляется или удаляется право доступа. Если осуществляется добавление права доступа own_r, $write_r$ или $append_r$ к сущности, то требуется, чтобы уровень целостности сущности y (если y является субъект-сессией, то текущий уровень целостности y) не превосходил уровня целостности роли r. При этом в случае, когда уровень целостности сущности y равняется i_high, в соответствии с предположением 8 требуется реализация субъект-сессией x информационного потока по памяти от себя к сущности i_entity. Так как добавление или удаление права доступа из множества прав доступа роли r может быть использовано для передачи данных, то в результате применения правил могут возникнуть информационные потоки по времени от субъект-сессии x к субъект-сессиям, фактически обладающим ролью r.

Правила $create_entity(x, r, y, yi, z)$, $rename_entity(x, y, z)$ и $delete_entity(x, y, z)$ позволяют субъект-сессии x создать, переименовать или удалить соответственно сущность y, не являющуюся субъект-сессией и входящую в состав контейнера z. При этом нельзя удалять или создавать сущности, входящие в состав учетных записей пользователей, или сущности, параметрически ассоциированные с ролями, и в отличие от БР ДП-модели для применения правил в соответствии с предположением 7 требуется, чтобы уровень целостности роли, используемой для создания, переименования или удаления сущности, и уровень целостности сущности-контейнера z, в котором содержится сущность y, не превосходили текущего уровня целостности субъект-сессии, от имени которой данные действия осуществляются. Если выполняется создание, переименование или удаление сущности y с уровнем целостности i_high, то в соответствии с предположением 8 требуется реализация субъект-сессией x информационного потока по памяти от себя к сущности i_entity. Кроме того, в соответствии с предположением 7 при создании сущности требуется, чтобы уровень целостности yi сущности y не превосходил уровня целостности сущности-контейнера z и уровня целостности роли r, которая получает право доступа владения к сущности y. Также следует заметить, что при применении правил $rename_entity(x, y, z)$ или $delete_entity(x, y, z)$ возникают информационные потоки по времени от субъект-сессии x к субъект-сессиям, имеющим текущие фактические доступы (а не фактические текущие права доступа, как в БР ДП-модели) к сущностям, подчиненным в иерархии сущности y.

Правило $create_first_session(x, u, r, y, z, zi)$ позволяет субъект-сессии x с использованием сущности y и учетной записи пользователя u создать от имени u новую субъект-сессию z с текущим уровнем целостности zi (в БР ДП-модели создание субъект-сессии осуществлял пользователь u). При этом в соответствии с предположением 2 для применения правила необходима реализация субъект-сессией x к себе информационных потоков по памяти от всех сущностей, параметрически ассоциированных с учетной записью пользователя u. Кроме того, в соответствии с предположением 7 требуется, чтобы текущий уровень целостности zi создаваемой субъект-сессии z не превосходил уровня целостности учетной записи пользователя u и уровня целостности роли r, которая получает право доступа владения к субъект-сессии z. Однако уровень целостности субъект-сессии z может превосходить уровень целостности субъект-сессии x, что соответствует подходу, реализованному, например, в ОС семейств *Microsoft Windows Vista*/7/2008. В случае, когда субъект-сессия z создается с уровнем целостности i_high, в соответствии с предположением 8 требуется реализация субъект-сессией x информационного потока по памяти от себя к сущности i_entity.

Правила $create_session(x, w, r, y, z, zi)$ и $delete_session(x, w, z)$ позволяют субъект-сессии x непосредственно либо от имени субъект-сессии w, к которой субъект-сессия x имеет доступ владения, создать или удалить соответственно субъект-сессию z. При этом в соответствии с предположением 7 требуется, чтобы текущий уровень целостности zi субъект-сессии z не превосходил уровня целостности роли r, получающей право доступа владения к субъект-сессии z, который, в свою очередь, не должен превосходить текущего уровня целостности субъект-сессии w, осуществляющей создание субъект-сессии z. Правило $delete_session(x, w, z)$ не рассматривалось в БР ДП-модели (кроме того, в дискреционных или мандатных ДП-моделях также отсутствовало правило удаления субъектов), для удаления субъект-сессий в этой модели использовалось правило удаления сущностей $delete_entity(x, y, z)$. Условия применения правила $delete_session(x, w, z)$ позволяют учесть, что удаление субъект-сессии z может быть осуществлено субъект-сессией x от имени субъект-сессии w, обладающей правом доступа владения к субъект-сессии z.

Правила $control(x, y, z)$ и $know(x, y)$ в рамках предположений 3 и 4 позволяют субъект-сессии x получить доступ владения к субъект-сессии y с использованием соответственно либо информационного потока по памяти от себя к сущности z, функционально ассоциированной с субъект-сессией y, либо информационных потоков по памяти к себе от всех сущностей, параметрически ассоциированных с субъект-сессией y. Параметрически ассоциированные сущности и правило $know(x, y)$ не рассматривались в БР ДП-модели (они анализировались в дискреционных ФПАС и ФС ДП-моделях [3, 4]).

Условия и результаты применения правила $take_access_own(x, y, z)$ не изменились по сравнению с БР ДП-моделью. Оно позволяет субъект-сессии x получить доступ владения к субъект-сессии z (при этом требуется, чтобы субъект-сессия x обладала доступом владения к субъект-сессии y, имеющей, в свою очередь, доступ владения к субъект-сессии z).

Правила $access_own(x, w, y)$, $access_read(x, w, y)$, $access_write(x, w, y)$ и $access_append(x, w, y)$ позволяют субъект-сессии x, обладающей фактической возможностью осуществить от имени субъект-сессии w с соответствующим правом доступа действие над сущностью y, либо получить самой (когда $x = w$), либо инициировать получение субъект-сессией w соответствующего доступа к сущности y. Данные правила отличаются от аналогичных правил БР ДП-модели тем, что субъект-сессия x не непосред-

ственно сама получает доступ к сущности y, а это делает субъект-сессия w, доступ владения к которой имеет субъект-сессия x.

Правила $flow(x, y, y', z)$, $find(x, y, z)$, $post(x, y, z)$, $pass(x, y, z)$ и $take_flow(x, y)$ позволяют субъект-сессиям создавать информационные потоки по памяти или по времени. Отличие условий применения данных правил от их аналогов в БР ДП-модели заключается в том, что вместо обладания фактическими текущими правами доступа для реализации информационных потоков от субъект-сессий требуется наличие соответствующих фактических доступов. Кроме того, при применении правила $flow(x, y, y', z)$ для создания информационных потоков по времени допускаются равенства $x = y$ или $y' = z$.

3. Монотонные и немонотонные правила преобразования состояний

По аналогии с дискреционными и мандатными ДП-моделями, а также с БР ДП-моделью дадим определение.

Определение 9. Монотонное правило преобразования состояний — это правило преобразования состояний из множества OP, применение которого не приводит к удалению из состояний

— ролей из множества текущих ролей субъект-сессий;
— прав доступа ролей к сущностям;
— субъект-сессий или сущностей;
— доступов субъект-сессий к сущностям;
— информационных потоков.

По определению 9 и в соответствии с условиями и результатами применения правил преобразования состояний, заданных в таблице, монотонными являются следующие правила: $take_role(x, w, r)$, $grant_right(x, r, (y, \alpha_r))$, $create_entity(x, r, y, yi, z)$, $rename_entity(x, y, z)$, $create_first_session(x, u, r, y, z, zi)$, $create_session(x, w, r, y, z, zi)$, $control(x, y, z)$, $know(x, y)$, $take_access_own(x, y, z)$, $access_own(x, w, y)$, $access_read(x, w, y)$, $access_write(x, w, y)$, $access_append(x, w, y)$, $flow(x, y, y', z)$, $find(x, y, z)$, $post(x, y, z)$, $pass(x, y, z)$ и $take_flow(x, y)$. Немонотонными правилами преобразования состояний являются $remove_role(x, w, r)$, $remove_right(x, r, (y, \alpha_r))$, $delete_entity(x, y, z)$, $delete_session(x, w, z)$.

Поскольку в рамках БРОС ДП-модели условия и результаты применения правил преобразования состояний по сравнению с БР ДП-моделью претерпели существенные изменения, целесообразно обосновать следующее утверждение.

Утверждение 1. Пусть $G_0 = (PA_0, user_0, roles_0, A_0, F_0, H_{E_0})$ — начальное состояние системы $\Sigma(G^*, OP, G_0)$. Пусть также существуют состояния системы $G_1, \ldots, G_N = (PA_N, user_N, roles_N, A_N, F_N, H_{E_N})$ и правила преобразования состояний $op_1, \ldots, op_N$, такие, что $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$, где $N \geqslant 0$. Тогда существуют состояния $G'_1, \ldots, G'_M = (PA'_M, user'_M, roles'_M, A'_M, F'_M, H'_{E_M})$, где $M \geqslant 0$, и монотонные правила преобразования состояний $op'_1, \ldots, op'_M$, такие, что $G_0 \vdash_{op'_1} G'_1 \vdash_{op'_2} \ldots \vdash_{op'_M} G'_M$, и выполняются следующие условия.

1. Верно включение $S_N \subset S'_M$ и для каждой субъект-сессии $s \in S_N$ выполняются условия: $user_N(s) = user'_M(s)$, $roles_N(s) \subset roles'_M(s)$.

2. Верно включение $E_N \subset E'_M$, для каждой сущности $e \in E_N \setminus S_N$, не являющейся субъектом, выполняется условие $H_{E_N}(e) \subset H'_{E_M}(e)$, и для любых сущностей $e, e' \in E_N$ если в состоянии G_N выполняется условие $e < e'$, то данное условие выполняется в состоянии G'_M.

3. Для каждой роли $r \in R$ выполняется условие $PA_N(r) \subset PA'_M(r)$.
4. Верно включение $A_N \subset A'_M$.
5. Верно включение $F_N \subset F'_M$.

Доказательство. Пусть существуют состояния G_0, G_1, …, G_N системы $\Sigma(G^*, OP, G_0)$ и правила преобразования состояний op_1, …, op_N, такие, что $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$, где $N \geqslant 0$. Докажем утверждение индукцией по длине N последовательности состояний.

Пусть $N = 0$. Тогда положим $M = 0$, $G'_0 = G_0$ и для состояний G_0 и G'_0 условия 1–5 утверждения выполнены.

Пусть $N > 0$ и условия утверждения выполнены для всех последовательностей состояний длины $0 \leqslant L < N$. Докажем, что условия 1–5 утверждения выполнены для последовательностей состояний длины N. По предположению индукции, для последовательности состояний $G_0, G_1, \ldots, G_{N-1}$ существует последовательность $G_0, G'_1, \ldots, G'_K = (PA'_K, user'_K, roles'_K, A'_K, F'_K, H'_{E_K})$, где $K \geqslant 0$, и для состояний G_{N-1} и G'_K выполнены условия 1–5 утверждения. Рассмотрим правило преобразования состояний op_N. Если условия его применения не выполняются в состоянии G_{N-1}, то по определению справедливо равенство $G_{N-1} = G_N$. Положим $M = K$; для состояний G_N и G'_M выполнены условия 1–5 утверждения. Пусть условия применения правила op_N выполняются в состоянии G_{N-1}. Тогда возможны два случая.

П е р в ы й с л у ч а й: правило преобразования состояний op_N является монотонным. Так как для состояний G_{N-1} и G'_K выполнены условия 1–5 утверждения и, по предположению 6, на траекториях системы не изменяются значения функций, задающих уровни целостности сущностей или субъект-сессий, существовавших в предшествующих состояниях системы, то в состоянии G'_K выполняются условия применения правила op_N. Тогда положим $M = K + 1$, $op'_M = op_N$, и пусть состояние G'_M получено из состояния G'_K применением к нему правила op_N: $G'_K \vdash_{op_N} G'_M$. В соответствии с заданными в таблице результатами применения правил преобразования для состояний G_N и G'_M выполнены условия 1–5 утверждения.

В т о р о й с л у ч а й: правило преобразования состояний op_N является немонотонным.

Пусть $op_N = remove_role(x, w, r)$. Так как для состояний G_{N-1} и G'_K выполнены условия 1–5 утверждения, то выполняются условия $(w, user'_K(w), \varnothing, \varnothing) \in de_facto_actions'_K(x)$, для $e \in]r[$ выполняется условие $(e, x, write_m) \in F'_K$, и если $i_r(r) = i_high$, то $(x, i_entity, write_m) \in F'_K$. Кроме того, так как в состоянии G_{N-1} выполняется условие $r \in roles_{N-1}(w)$, то по предположению 7 в состоянии G'_K справедливо $r \in UA'_K(user'_K(w)) \cup AUA'_K(user'_K(w))$ и $i_r(r) \leqslant i_s(w)$. Следовательно, в состоянии G'_K выполнены условия применения правила $take_role(x, w, r)$. Тогда положим $M = K+1$, $op'_M = take_role(x, w, r)$, и пусть состояние G'_M получено из состояния G'_K применением к нему правила op'_M: $G'_K \vdash_{op'_M} G'_M$. В соответствии с заданными в таблице результатами применения правил преобразования состояний $remove_role(x, w, r)$ и $take_role(x, w, r)$ для состояний G_N и G'_M выполнены условия 1–5 утверждения.

Пусть $op_N = remove_right(x, r, (y, \alpha_r))$. Так как для состояний G_{N-1} и G'_K выполнены условия 1–5 утверждения, то выполняются условия $(w, \varnothing, (y, own_r), r) \in de_facto_actions'_K(x)$, $i_r(r) \leqslant i_s(w)$, и если $i_e(y) = i_high$, то $(x, i_entity, write_m) \in F'_K$. При этом если $\alpha_r \in \{own_r, write_r, append_r\}$, то по предположению 7 в состоянии G'_K выполняется условие $i_r(r) \leqslant i_s(w)$. Следовательно, в состоянии G'_K выполнены условия применения правила $grant_right(x, r, (y, \alpha_r))$. Тогда положим $M = K + 1$, $op'_M = grant_right(x, r, (y, \alpha_r))$, и пусть состояние G'_M получено из состояния G'_K при-

менением к нему правила $op'_M : G'_K \vdash_{op'_M} G'_M$. В соответствии с результатами применения правил преобразования $remove_right(x, r, (y, \alpha_r))$ и $grant_right(x, r, (y, \alpha_r))$ для состояний G_N и G'_M выполнены условия 1–5 утверждения.

Пусть $op_N = delete_entity(x, y, z)$. Так как для состояний G_{N-1} и G'_K выполнены условия 1–5 утверждения, то выполняются условия $(w, \varnothing, (z, write_r), r) \in de_facto_actions'_K(x)$, $i_r(r) \leqslant i_s(w)$, $i_e(z) \leqslant i_s(w)$, и если $i_e(z) = i_high$, то $(x, i_entity, write_m) \in F'_K$. Следовательно, в состоянии G'_K выполнены условия применения правил $rename_entity(x, y, z)$ и $access_write(x, w, z)$. Тогда положим $M = K+2$, $op'_{M-1} = rename_entity(x, y, z)$, $op'_M = access_write(x, w, z)$, и пусть состояние G'_M получено из состояния G'_K применением к нему правил op'_{M-1}, op'_M: $G'_K \vdash_{op'_{M-1}} G'_{M-1} \vdash_{op_M} G'_M$. В соответствии с заданными в таблице результатами применения правил преобразования состояний $delete_entity(x, y, z)$, $rename_entity(x, y, z)$ и $access_write(x, w, z)$ для состояний G_N и G'_M выполнены условия 1–5 утверждения.

Пусть $op_N = delete_session(x, w, z)$. Так как для состояний G_{N-1} и G'_K выполнены условия 1–5 утверждения, то выполняется условие $(w, \varnothing, (z, own_r), \varnothing) \in de_facto_actions'_K(x)$. Следовательно, в состоянии G'_K выполнены условия применения правила $access_own(x, w, z)$. Тогда положим $M = K + 1$, $op'_M = access_own(x, w, z)$, и пусть состояние G'_M получено из состояния G'_K применением к нему правила op'_M: $G'_K \vdash_{op'_M} G'_M$. В соответствии с заданными в таблице результатами применения правил преобразования состояний $delete_session(x, w, z)$ и $access_own(x, w, z)$ для состояний G_N и G'_M выполнены условия 1–5 утверждения.

Следовательно, условия 1–5 утверждения выполнены для последовательностей состояний длины N, и шаг индукции доказан. Утверждение доказано. ■

Таким образом, несмотря на то, что в БРОС ДП-модель добавлены по сравнению с БР ДП-моделью несколько существенно новых элементов (учетные записи пользователей, сущности, параметрически ассоциированные с субъект-сессиями или ролями, мандатный контроль целостности, фактические доступы субъект-сессий), в ее рамках при анализе условий передачи прав доступа, реализации информационных потоков по памяти или по времени также возможно использование только монотонных правил преобразования состояний. В дальнейшем предполагается построение на основе БРОС ДП-модели новых ролевых ДП-моделей, позволяющих учесть особенности функционирования современных ОС, особенно ОС семейства *Linux*.

ЛИТЕРАТУРА

1. *Девянин П. Н.* Анализ в рамках базовой ролевой ДП-модели безопасности систем с простыми траекториями функционирования // Прикладная дискретная математика. 2010. № 1(7). С. 16–36.
2. *Девянин П. Н.* Модели безопасности компьютерных систем. Управление доступом и информационными потоками. Учеб. пособие для вузов. М.: Горячая линия-Телеком, 2011. 320 с.
3. *Колегов Д. Н.* ДП-модель компьютерной системы с функционально и параметрически ассоциированными с субъектами сущностями // Вестник Сибирского государственного аэрокосмического университета им. акад. М. Ф. Решетнева. 2009. Вып. 1(22). Ч. 1. С. 49–54.
4. *Буренин П. В.* Подходы к построению ДП-модели файловых систем // Прикладная дискретная математика. 2009. № 1(3). С. 93–112.