МАТЕМАТИЧЕСКИЕ ОСНОВЫ КОМПЬЮТЕРНОЙ БЕЗОПАСНОСТИ

УДК 004.94

ПРАВИЛА ПРЕОБРАЗОВАНИЯ СОСТОЯНИЙ СИСТЕМЫ В РАМКАХ ДП-МОДЕЛИ УПРАВЛЕНИЯ ДОСТУПОМ В КОМПЬЮТЕРНЫХ СЕТЯХ, ПОСТРОЕННЫХ НА ОСНОВЕ ОС СЕМЕЙСТВА LINUX

В. Ю. Тележников

Учебно-методическое объединение по образованию в области информационной безопасности, г. Москва, Россия

Рассматривается мандатная сущностно-ролевая ДП-модель управления доступом в компьютерных сетях ОС семейства *Linux*, построенная на основе МРОСЛ ДП-модели. В рамках данной модели уточняются существующие и задаются новые де-юре правила преобразования состояний системы, позволяющие учитывать особенности функционирования рассматриваемых компьютерных сетей, обосновывается корректность их задания относительно требований, предъявляемых к мандатному и ролевому управлению доступом. Наличие новых правил преобразования состояний позволяет более детально описывать спецификации механизмов управления доступом, что нацелено на построение теоретически обоснованных локальных и сетевых подсистем безопасности компьютерных сетей.

Ключевые слова: *компьютерные сети, модели безопасности, операционная система Linux, ДП-модели.*

DOI 10.17223/20710410/31/7

SYSTEM STATE TRANSFORMATION RULES IN DP-MODEL OF ACCESS CONTROL IN COMPUTER NETWORKS BASED ON OPERATING SYSTEMS OF LINUX

V. Y. Telezhnikov

EMA IS, Moscow, Russia

E-mail: telezhnikov.v@yandex.ru

When modern computer systems (CS) are created, a big attention is paid to theoretical explanation of their access control security mechanisms. For this aim, some formal models are built and mandatory MROSL DP-model is the most developed of them. However, it is important to consider peculiarities of logical access control organization in computer networks and the existence of different security policies of network stations. MROSL DP-model and other models known to the author do not take this into account. Besides, it is necessary to provide flexible specification of access control to network resources in the context of theoretical models of logical access control in computer systems including hundreds and thousands users. The simple mechanism of access control administration is also needed. The author is building new mandatory

object-role access control DP-model for the computer systems based on OS of Linux family (MROCS DP-model) relying on MROSL DP-model in order to consider mentioned peculiarities. Existing de-jure rules of system state transformation are refined and new ones are specified in the context of this model for the purpose of taking into account peculiarities of functioning CS under consideration. These changes allow to describe in details specifications of access control mechanisms. Besides, the correctness of these rules with respect to mandatory and role-based access control requirements is shown, so it makes possible to construct theory-based network security subsystem of CS. De-jure rules of state transformation in MROCS DP-model connected with the organization of logical access control in the context of CS are directed to realization in special operating system Astra Linux Special Edition.

Keywords: *computer networks, operating systems of Linux, DP-model.*

Введение

При создании современных автоматизированных систем большое внимание уделяется теоретическому обоснованию безопасности их механизмов управления доступом. Для этого строятся формальные модели логического управления доступом, наиболее развитой из которых является мандатная сущностно-ролевая ДП-модель управления доступом и информационными потоками в защищённых ОС семейства *Linux* (МРОСЛ ДП-модель) [1, 2]. Одним из примеров её состоятельности является тот факт, что в настоящее время широкое распространение получила ОС специального назначения (ОССН) Astra Linux Special Edition [3], реализующая мандатное управление доступом, развитие которой ведётся в направлении адаптации её подсистемы безопасности к требованиям МРОСЛ ДП-модели с учётом поддержки мандатной политики целостности и ролевого управления доступом. Вместе с тем при объединении ОС в сеть целесообразно принимать во внимание особенности организации логического управления доступом к сетевым объектам, наличие различных политик безопасности рабочих станций сети, которые не учитываются как в МРОСЛ ДП-модели, так и в других известных автору формальных моделях. При этом в рамках теоретических моделей логического управления доступом в сетях, в которых одновременно могут работать сотни и тысячи пользователей, необходимо предусмотреть возможность гибкого задания прав доступа к ресурсам сети; они должны также обладать простым механизмом администрирования. С целью учёта этих особенностей на основе МРОСЛ ДП-модели автором ведётся построение новой мандатной сущностно-ролевой ДП-модели управления доступом в компьютерных сетях, построенных на основе ОС семейства *Linux* (МРОКС ДП-модели), которую можно будет использовать для создания теоретически обоснованного механизма защиты ОС, способных функционировать в рамках сети.

1. Основные элементы МРОКС ДП-модели

В силу того, что разработка новой модели ведётся на основе МРОСЛ ДП-модели, принятые в ней обозначения используются при изложении МРОКС ДП-модели.

В МРОКС ДП-модели рабочей станции сети соответствует любой отдельный персональный компьютер либо сервер сети, каждый из которых оборудован сетевым адаптером. При этом в описании модели предполагается, что заранее известны рабочие станции, которые могут функционировать в рамках сети:

— ω — количество рабочих станций, которые могут функционировать в сети, $\omega \geqslant 1$;
— υ — количество инициализированных (подключённых) рабочих станций сети в заданный момент времени, $1 \leqslant \upsilon \leqslant \omega$;

— $WU \in U$ — множество специальных учётных записей (учётная запись-компьютер), каждая из которых соответствует отдельной рабочей станции сети.

В реальных компьютерных сетях есть возможность управления доступом на уровне рабочей станции сети, когда доступ предоставляется или запрещается, исходя не только из того, от имени какой учётной записи он осуществляется и к какому ресурсу, но и из того, какие рабочие станции при этом взаимодействуют. Для возможности отражения подобных ситуаций задаются новые множества и функции.

В МРОКС ДП-модели каждой рабочей станции, которая может функционировать в рамках сети, соответствуют отдельные подмножества субъект-сессий и сущностей сети:

— S_i — множество субъект-сессий рабочей станции сети, где $1 \leqslant i \leqslant \omega$;
— $E_i = O_i \cup C_i$ — множество сущностей рабочей станции сети, где O_i — множество объектов, C_i — множество контейнеров и $O_i \cap C_i = \varnothing$, $1 \leqslant i \leqslant \omega$;
— $OS_i \subset O_i$ — множество объектов-сокетов рабочей станции сети, где $1 \leqslant i \leqslant \omega$.

Замечание 1. В множество сущностей рабочих станций сети необходимо включить все сущности реальной ОС, к которым требуется назначать права доступа. Применительно к ОС семейства *Linux* такими объектами являются: регулярные (обычные) файлы, директории, символьные ссылки, файлы блочных устройств, специальные файлы устройств, сокеты, именованные каналы. В качестве сущностей следует рассматривать и отдельные файловые системы в совокупности. Среди всех сущностей только директории относятся к сущностям-контейнерам. Все остальные являются сущностями-объектами. Такие объекты файловых систем, как жёсткие ссылки, относятся также к сущностям-объектам и позволяют помещать их сразу в несколько сущностей-контейнеров.

В качестве субъект-сессий необходимо рассматривать все активные сущности ОС — «демоны», системные и прикладные (пользовательские) процессы, а также возможные их совокупности, например процессы, выполняющие одну задачу в рамках системы. В том случае, когда ОС функционирует в компьютерной сети, целесообразно рассматривать данную ОС в качестве самостоятельной субъект-сессии, которой будут подчинены все остальные субъект-сессии в иерархии, заданной на множестве субъект-сессий в МРОСЛ ДП-модели [1].

В множество объектов-сокетов необходимо включать все виды объектов реальной ОС, взаимодействие с которыми организуется через интерфейс сокетов и которые предназначены для предоставления доступа к ресурсам удалённых рабочих станций сети. Стоит отметить, что применительно к ОС семейства *Linux* интерфейс сокетов используется для работы как с сетевыми, так и с файловыми сокетами. Однако файловые сокеты используются как механизм межпроцессного взаимодействия в рамках одной рабочей станции, поэтому в МРОКС ДП-модели элементами множества OS являются только сетевые сокеты. При этом, с учётом специфики функционирования ОС семейства *Linux* (невозможность создания «жёстких» ссылок на сетевые сокеты, размещение их в корне псевдофайловой системы), все сетевые сокеты отдельной рабочей станции сети содержатся в единственной сущности-контейнере: существует единственная сущность-контейнер $c \in C$, такая, что для любого объекта-сокета $os \in OS$ выполняется $os \in H_E(c)$, то есть область значения функции $H_E^{-1}(OS)$ соответствует множеству, состоящему из одной сущности-контейнера.

Для любого $1 \leqslant i \leqslant \omega$ из множества субъект-сессий выделяется множество сетевых субъект-сессий рабочей станции сети $SN_i \subseteq S_i$.

Для ОС, функционирующей на рабочей станции сети, задаётся отдельный элемент множества сетевых субъект-сессий: $wsn_i \in SN_i$ — сетевая субъект-сессия, соответствующая отдельной рабочей станции в рамках сети; $CSN = \{wsn_1, wsn_2, \ldots, wsn_\omega\}$ — множество сетевых субъект-сессий, каждая из которых соответствует отдельной рабочей станции, которая может функционировать в сети.

Замечание 2. В множество сетевых субъект-сессий SN входят те процессы системы, которые являются участниками сетевого соединения с удалёнными рабочими станциями сети. При этом только в процессе функционирования сетевых субъект-сессий представляется возможным получить какой-либо удалённый доступ к ресурсам сети. В рамках разрабатываемой МРОКС ДП-модели отдельные рабочие станции сети являются сетевыми субъект-сессиями системы. Таким образом, сетевые субъект-сессии, входящие во множество CSN, представляют собой отдельные рабочие станции сети, участвующие в сетевом взаимодействии. В качестве такой сетевой субъект-сессий возможно рассматривать процесс *init*, запущенный на каждой отдельной рабочей станции сети.

Замечание 3. В МРОСЛ ДП-модели [1] для любой учётной записи пользователя задаётся соответствующая только ей индивидуальная роль. Исходя из этого, предлагается на практике обеспечить возможность функционирования каждой субъект-сессии wsn_i от имени учётной записи пользователя $u \in WU$, идентификатор которой будет уникальным в рамках всей рассматриваемой сети. Благодаря этому и в силу того, что их индивидуальные роли также уникальны, субъект-сессии wsn_i единственным образом ставится в соответствие конкретная рабочая станция сети.

Используем следующие обозначения:

— $csn\_initialized : CSN \to \{\texttt{true}, \texttt{false}\}$ — функция принадлежности рабочей станции к множеству инициализированных рабочих станций сети, такая, что для любого i, где $1 \leqslant i \leqslant \omega$, рабочая станция считается инициализированной, если для соответствующей ей сетевой субъект-сессии $wsn_i \in SN_i$ выполняется $csn\_initialized(wsn_i) = \texttt{true}$, в противном случае wsn_i выполняется на изолированной рабочей станции сети;

— $csn\_accessory : (E_1 \cup \ldots \cup E_\omega) \cup (S_1 \cup \ldots \cup S_\omega) \to CSN$ — функция принадлежности субъект-сессий и сущностей к рабочей станции сети, такая, что для сущности или субъект-сессии $e\_s \in (E_1 \cup \ldots \cup E_\omega) \cup (S_1 \cup \ldots \cup S_\omega)$ существует единственное i, $1 \leqslant i \leqslant \omega$, такое, что $csn\_accessory(e\_s) = wsn_i$. По определению $csn\_accessory(wsn_i) = wsn_i$, $1 \leqslant i \leqslant \omega$.

Замечание 4. В тот момент, когда происходит подключение (инициализация) компьютера к сети, в множество субъект-сессий и сущностей сети добавляются множества субъект-сессий и сущностей, принадлежащих инициализируемой рабочей станции. При этом подразумевается, что заданные параметры функционирования инициализируемой рабочей станции удовлетворяют всем требованиям к реализации мандатного и ролевого управления доступом, предъявляемым к уже инициализированным рабочим станциям сети.

Таким образом, задаются непересекающиеся подмножества сущностей и субъект-сессий, принадлежащих отдельным рабочим станциям, которые могут функционировать в рамках сети:

— для любой субъект-сессии $s \in S_i$ выполняется $csn\_accessory(s) = wsn_i$, $1 \leqslant i \leqslant \omega$;
— для любой сущности $e \in E_i$ выполняется $csn\_accessory(e) = wsn_i$, $1 \leqslant i \leqslant \omega$;
— для $1 \leqslant i < j \leqslant \omega$ выполняется $S_i \cap S_j = \varnothing$, $E_i \cap E_j = \varnothing$.

Множества субъект-сессий и сущностей различных инициализированных рабочих станций объединяются в множества субъект-сессий и сущностей сети соответственно:

— $S = S_{i_1} \cup S_{i_2} \cup \ldots \cup S_{i_v}$ и для $wsn_{i_j} \in SN_{i_j}$ выполняется $csn\_initialized(wsn_{i_j}) = \texttt{true}$, где $1 \leqslant j \leqslant v$ и $1 \leqslant i_j \leqslant \omega$;
— $E = E_{i_1} \cup E_{i_2} \cup \ldots \cup E_{i_v}$ и для $wsn_{i_j} \in SN_{i_j}$ выполняется $csn\_initialized(wsn_{i_j}) = \texttt{true}$, где $1 \leqslant j \leqslant v$ и $1 \leqslant i_j \leqslant \omega$.

Замечание 5. В МРОСЛ ДП-модели [1] учитывается, что ОС семейства *Linux* предоставляют возможность создания «жёстких» ссылок на объекты файловой системы. Это приводит к тому, что в рамках модели одна и та же сущность-объект (файл) может содержаться в нескольких сущностях-контейнерах (директориях). Однако это верно только для сущностей в рамках любой отдельной рабочей станции сети, так как «жёсткая» ссылка не может выходить за пределы файловой системы, в которой находится объект, на который она ссылается. Поэтому невозможно создать «жёсткую» ссылку на удалённой рабочей станции сети и, как следствие, определение функции $csn\_accessory$ корректно относительно создания «жёстких» ссылок.

Пусть $IS \subseteq OS \times OS$ — множество установленных соединений между объектами-сокетами, используемых субъект-сессиями для передачи данных между собой. По определению если $(os_i, os_j) \in IS$, то $(os_j, os_i) \in IS$, где $os_i, os_j \in OS$, $1 \leqslant i,\ j \leqslant \omega$ и $csn\_initialized(wsn_i) = csn\_initialized(wsn_j) = \texttt{true}$.

Замечание 6. Под установленным соединением будем понимать пару связанных между собой сокетов, используемых для передачи данных между субъект-сессиями посредством интерфейса сокетов с применением сетевых протоколов, ориентированных на установление соединения (например, TCP).

Теоретические модели компьютерных сетей, в которых одновременно могут работать сотни и тысячи пользователей, должны предоставлять возможность гибкого задания прав доступа к ресурсам сети и обладать простым механизмом администрирования ролевого управления доступа. Для этих целей в рамках разрабатываемой автором МРОКС ДП-модели наряду с множествами ролей и административных ролей задаётся новое множество запрещающих ролей, отличительная особенность которых в ограничении прав доступа пользователей, обладающих ими.

Используем следующие обозначения:

— R — множество ролей;
— AR — множество административных ролей, при этом по определению $AR \cap R = \varnothing$;
— UR — множество запрещающих ролей, ограничивающих права доступа к сущностям, при этом по определению $R \cap UR = \varnothing$ и $AR \cap UR = \varnothing$.

Отличительная особенность запрещающих ролей в том, что через них задается множество тех прав доступа к сущностям, которые не должна получить субъект-сессия, функционирующая от имени учётной записи пользователя, права доступа которой необходимо ограничить.

Замечание 7. Отсутствие множества запрещающих административных ролей объясняется тем, что число ролей и запрещающих ролей, используемых в реальных ОС, значительно меньше числа сущностей системы, что позволяет гибко задавать административные права доступа к ролям, запрещающим ролям при наличии только административных ролей.

На множестве запрещающих ролей задаётся иерархия аналогично тому, как задаётся иерархия на множестве ролей в МРОСЛ ДП-модели [1].

Определение 1. Иерархией запрещающих ролей называется заданное на множестве запрещающих ролей UR отношение частичного порядка $\leqslant$. При этом по определению справедливо: для административной роли $ar \in AR$, если запрещающие роли $ur', ur \in UR$ таковы, что $(ur, read_r) \in APA(ar)$ и $ur' \leqslant ur$, то выполняется условие $(ur', read_r) \in APA(ar)$. При этом по определению роли, запрещающие роли и административные роли несравнимы между собой, т. е. их иерархии считаются независимыми.

Множества R_r видов прав доступа, R_a видов доступа, R_f видов информационных потоков (по памяти и по времени) и P прав доступа к сущностям и субъект-сессиям задаются аналогично МРОСЛ ДП-модели [1]:

— $R_r = \{read_r, write_r, execute_r, own_r\}$ — множество видов прав доступа;
— $R_a = \{read_a, write_a\}$ — множество видов доступа;
— $R_f = \{write_m, write_t\}$ — множество видов информационных потоков (по памяти и по времени);
— $P \subseteq (E \times R_r) \cup (S \times \{own_r\})$ — множество прав доступа к сущностям и субъект-сессиям.

При этом следующие множества переопределяются с учётом введения запрещающих ролей:

— $AP \subseteq (R \cup UR \cup AR) \times R_r$ — множество административных прав доступа к ролям, запрещающим ролям или административным ролям;
— $AA \subseteq S \times (R \cup UR \cup AR) \times R_a$ — множество доступов субъект-сессий к ролям, запрещающим ролям или административным ролям;
— $F \subseteq (E \cup S \cup R \cup UR \cup AR) \times (E \cup S \cup R \cup UR \cup AR) \times R_f$ — множество информационных потоков;
— $PA : R \cup UR \cup AR \to 2^P$ — функция прав доступа к сущностям ролей, запрещающих ролей и административных ролей (для запрещающих ролей право доступа владения к сущностям не задаётся);
— $APA : AR \to 2^{AP}$ — функция административных прав доступа к ролям, запрещающим ролям и административным ролям административных ролей.

Вместе с тем такие множества, как P и AA представляют собой объединение соответствующих множеств, заданных для каждой инициализированной рабочей станции сети, при этом $1 \leqslant i \leqslant \omega$ и $csn\_initialized(wsn_i) = \texttt{true}$:

— $P_i \subseteq (E_i \times R_r) \cup (S_i \times \{own_r\})$ — множество прав доступа к сущностям и субъект-сессиям, содержащимся на рабочей станции сети wsn_i (для $y \in E_i \cup S_i$ выполняется $csn\_accessory(y) = wsn_i$);
— $AA_i \subseteq S_i \times (R \cup UR \cup AR) \times R_a$ — множество административных доступов субъект-сессий рабочей станции wsn_i к ролям, запрещающим ролям или административным ролям (для $s \in S_i$ выполняется $csn\_accessory(s) = wsn_i$).

С учётом этого, для $1 \leqslant i \leqslant \omega$ задаются отдельные функции прав доступа:

— $PA_i : R \cup UR \cup AR \to 2^{P_i}$ — функция прав доступа к сущностям рабочей станции сети wsn_i ролей, запрещающих ролей и административных ролей.

Множества A и F, помимо этого, включают в себя соответственно множества доступов и информационных потоков, возникающих между субъект-сессиями и сущностями различных инициализированных рабочих станций сети, где $1 \leqslant i \leqslant \omega$, $1 \leqslant j \leqslant \omega$ и $csn\_initialized(wsn_i) = csn\_initialized(wsn_j) = \texttt{true}$:

— $A_{i_j} \subseteq S_i \times E_j \times R_a$ — множество доступов субъект-сессий рабочей станции сети wsn_i к сущностям рабочей станции wsn_j (для $s \in S_i$ верно $csn\_accessory(s) = wsn_i$, для $e \in E_j$ выполняется $csn\_accessory(e) = wsn_j$);
— $F_{i_j} \subseteq (E_i \cup S_i \cup R \cup UR \cup AR) \times (E_j \cup S_j \cup R \cup UR \cup AR) \times R_f$ — множество информационных потоков между рабочими станциями wsn_i и wsn_j.

Множество функций $Constraint_{APA}$, задающих ограничение на значения множеств административных прав доступа административных ролей, является одинаковым для всех рабочих станций сети. Множества функций $Constraint_{PA}$ и $Constraint_{AA}$ задаются отдельно для каждой рабочей станции сети.

Права доступа административных ролей AR к запрещающим ролям UR интерпретируются следующим образом:

— $execute_r$ — право обращаться к ролям, подчинённым данной роли в иерархии запрещающих ролей;
— наличие права доступа $read_r$ к запрещающей роли указывает на необходимость включения роли во множество текущих ролей субъект-сессии: если $x \in S$ и существует $ar \in AR$, такая, что $(x, ar, read_a) \in AA$, то для любой запрещающей роли $ur \in UR$, для которой $(ur, read_r) \in PA(ar)$, выполняется $(x, ur, read_a) \in AA$;
— $write_r$ — право изменять множество прав доступа запрещающей роли.

Замечание 8. Для возможности назначения «принудительного» (при получении доступа на чтение к административной роли) доступа $read_a$ к запрещающим ролям в рамках МРОКС ДП-модели предполагается, что не существует параметрически ассоциированных с ними сущностей: для запрещающей роли $ur \in UR$ верно равенство $]ur[= \varnothing$.

Используем следующее обозначение:

— $network\_role : R \cup UR \cup AR \to \{\texttt{true}, \texttt{false}\}$ — функция сетевых ролей, такая, что роль, запрещающая роль или административная роль $r \in R \cup UR \cup AR$ является сетевой, когда $network\_role(r) = \texttt{true}$, в противном случае $network\_role(r) = \texttt{false}$ (локальная роль).

Замечание 9. Функция $network\_role$ позволяет определить множество тех ролей, обладание которыми даёт возможность субъект-сессиям участвовать в процессе обмена данными между рабочими станциями по сети. Например, можно выделить роли, доступные учётным записям пользователей независимо от рабочей станции в масштабах всей сети; все остальные роли предназначены для локального использования на конкретной рабочей станции. Для реализации данной функции на практике необходимо либо по умолчанию при получении доступа к удалённым ресурсам учитывать наличие только сетевых ролей, либо предлагать пользователю получить в качестве текущих ролей те необходимые роли $r \in R \cup UR \cup AR$, для которых $network\_role(r) = \texttt{true}$. При реализации первого подхода в рамках отдельной рабочей станции сети пользователю доступны все соответствующие ему авторизованные роли, независимо от значения функции $network\_role$, а в процессе управления доступом к данным, передаваемым по сети, учитываются права доступа только сетевых ролей, имеющихся в множествах текущих ролей соответствующих субъект-сессий.

Отметим, что рабочим станциям сети в рамках модели соответствуют субъект-сессии, к которым в МРОСЛ ДП-модели [1] может задаваться только право доступа own_r. Для обеспечения гибкого управления доступом (задания прав доступа $read_r$, $write_r$ и т. д., что позволит при получении доступа к сущностям учитывать, на ка-

ких рабочих станциях сети они находятся) к удалённым рабочим станциям сети (т. е. соответствующей субъект-сессии) введём следующие обозначения:

— $NR \subset R$ — множество ролей удалённого доступа, при этом по определению для любой роли $nr \in NR$ выполняется $network\_role(nr) = \texttt{true}$ (обладание данными ролями даёт возможность обращения к ресурсам удалённых рабочих станций, в том числе серверов сети);
— $NUR \subset UR$ — множество запрещающих ролей удалённого доступа, при этом по определению для любой роли $nur \in NUR$ выполняется $network\_role(nur) = \texttt{true}$ (обладание данными ролями может запретить обращение к ресурсам удалённых рабочих станций, в том числе серверов сети).

Замечание 10. Для каждой рабочей станции сети существует единственная сущность-«макроконтейнер», в которой содержатся все сущности данной рабочей станции. К ней могут быть заданы права доступа для ролей удалённого доступа: для любого $1 \leqslant i \leqslant \omega$ существует единственная сущность-«макроконтейнер» $em \in C_i$, такая, что $csn\_accessory(em) = wsn_i$ и для любой сущности $e \in E_i$, $e \neq em$, выполняется $e < em$. Все сущности-«макроконтейнеры» объединяются в единое множество $EM \subset C$, где $EM = \{em_1, \ldots, em_\omega\}$ и $csn\_accessory(em_i) = wsn_i$ для любого $1 \leqslant i \leqslant \omega$.

Замечание 11. Сущностью-«макроконтейнером» в реальных ОС семейства *Linux* может являться корневой каталог, который обозначается «/». Именно к нему задаются права доступа для ролей удалённого доступа, на основе которых принимается решение о возможности обращения (установления сетевого соединения) к ресурсам удалённой рабочей станции сети. Права доступа всех остальных ролей к сущностям-«макроконтейнерам» интерпретируются традиционным образом. Таким образом, возможность получения запрашиваемого доступа к сущностям удалённой рабочей станции сети зависит не только от множества текущих ролей и мандатных атрибутов доступа субъект-сессии, но и от рабочей станции сети, на которой она функционирует.

Зададим административные роли:

— $admin\_net \in ADMIN\_ROLES$, получение доступа $read_a$ к которой необходимо для администрирования ролей сетевого доступа из NR (изменение множества прав доступа к сущностям-«макроконтейнерам»);
— $admin\_csn \in ADMIN\_ROLES$, получение доступа $read_a$ к которой необходимо для администрирования удалённых рабочих станций сети.

При этом по определению для заданных административных ролей выполняется $network\_role(admin\_csn) = \texttt{true}$, $network\_role(admin\_net) = \texttt{true}$, $i_r(admin\_net) = i_r(admin\_csn) = i\_high$.

2. Правила задания мандатного и ролевого управления доступом

С целью задания порядка предоставления прав доступа к сущностям для ролей и запрещающих ролей удалённого доступа и доступов субъект-сессий к ним сделаем предположение.

Предположение 1. В МРОКС ДП-модели выполняются следующие условия.

У с л о в и е 1. Административное право доступа $read_r$ к ролям и запрещающим ролям удалённого доступа задаётся только для индивидуальных административных ролей учётных записей-компьютеров $WU \subset U$: для ролей удалённого доступа $nr \in NR$, если административная роль $ar \in AR$ и $(nr, read_r) \in APA(ar)$, то $ar \in \{u\_admin\_i\_high : u \in WU\}$.

У с л о в и е 2. Для каждой учётной записи-компьютера и для каждого уровня конфиденциальности, не превышающего уровня конфиденциальности этой учётной записи, задана по крайней мере одна (количество может зависеть, например, от числа сетевых интерфейсов) индивидуальная роль удалённого доступа, на которую авторизована данная учётная запись: для каждой $wu \in WU$ и каждого уровня конфиденциальности $l \leqslant f_u(wu)$ существует роль удалённого доступа $wu\_nr\_l\_li \in NR$, такая, что $(wu\_nr\_l\_li, read_r) \in APA(wu\_admin\_li)$, $f_r(wu\_nr\_l\_li) = l$ и $i_r(wu\_nr\_l\_li) = i_u(wu)$, $li \in LI$, где $LI = \{i\_low, i\_high\}$ — множество уровней целостности данных, $i\_low < i\_high$.

У с л о в и е 3. Доступ на чтение к ролям и запрещающим ролям удалённого доступа могут получить только субъект-сессии, соответствующие рабочим станциям сети: для субъект-сессии $s \in S \setminus CSN$ и роли $r \in NR \cup NUR$ выполняется $(s, r, read_a) \notin AA$.

У с л о в и е 4. Для создания, удаления, переименования, изменения иерархии ролей удалённого доступа, назначения к ним прав доступа административных ролей, а также для изменения их прав доступа к сущностям-«макроконтейнерам» субъект-сессия должна обладать доступом на чтение к административной роли $admin\_net$.

Замечание 12. При попытке сетевой субъект-сессии s_i получить доступ к сущности $e_j \in E_j$, находящейся на удалённой рабочей станции сети wsn_j, первоначально на рабочей станции сети wsn_i проверяется значение функции $csn\_initialized(wsn_j)$ (доступность или недоступность удалённой рабочей станции сети). Если оно равно `true` (т. е. удалённая рабочая станция инициализирована в сети), то проверяется отсутствие в множестве текущих ролей субъект-сессии wsn_i, функционирующей от имени учётной записи-компьютера wu_i ($user(wsn_i) = wu_i$), запрещающей роли, обладающей правом доступа $execute_r$ к сущности-«макроконтейнеру» em_j, и наличие по крайней мере одной сетевой роли удалённого доступа, обладающей этим же правом доступа к em_j, где $e_j < em_j$. При выполнении этих условий проверяется возможность получения запрашиваемого доступа непосредственно к самой сущности e_j.

С целью задания мандатного управления доступом к новым по сравнению с МРОСЛ ДП-моделью элементам системы сделаем следующее предположение.

Предположение 2. В рамках МРОКС ДП-модели для элементов системы выполняются следующие условия.

У с л о в и е 1 (мандатные уровни конфиденциальности и целостности сущностей-«макроконтейнеров», учётных записей-компьютеров). Вся траектория функционирования системы, соответствующей компьютерной сети, включая её начальное состояние, удовлетворяет следующим требованиям:

— каждая сущность-«макроконтейнер» $em_i \in E_i$ обладает высоким уровнем целостности, при этом выполняется условие $CCRI(em_i) = \texttt{false}$, где $1 \leqslant i \leqslant \omega$;
— сущность-«макроконтейнер» обладает максимальным уровнем конфиденциальности среди всех сущностей конкретной рабочей станции сети: для сущности $em_i \in EM$ и любой сущности $e \in E_i$ справедливо неравенство $f_e(e) \leqslant f_e(em_i) \leqslant f_{em}(em_i)$ и $CCR(em_i) = \texttt{false}$, где $1 \leqslant i \leqslant \omega$;
— каждая учётная запись-компьютер обладает высоким уровнем целостности: для учётной записи $wu \in WU$ выполняется $i_u(wu) = i\_high$.

У с л о в и е 2 (мандатные уровни конфиденциальности и целостности рабочих станций сети). Вся траектория функционирования системы, соответствующей компьютерной сети, включая её начальное состояние, удовлетворяет следующим требованиям:

— уровень доступа субъект-сессии wsn_i не должен превышать уровня конфиденциальности сущности-«макроконтейнера» em_i: для $em_i \in E_i$ и субъект-сессии $wsn_i \in SN_i$ выполняется $f_s(wsn_i) \leqslant f_e(em_i)$, где $1 \leqslant i \leqslant \omega$;
— каждая субъект-сессия, соответствующая отдельной рабочей станции сети, обладает высоким уровнем целостности: для субъект-сессии wsn_i выполняется $i_s(wsn_i) = i\_high$, где $1 \leqslant i \leqslant \omega$.

У с л о в и е 3. Все роли удалённого доступа обладают высоким уровнем целостности: для любой роли удалённого доступа $nr \in NR$ справедливо равенство $i_r(nr) = i\_high$ (это верно и для запрещающих ролей удалённого доступа, так как они все обладают высоким уровнем целостности по условию 3 предположения 1).

У с л о в и е 4. Субъект-сессия wsn_i получает доступ на чтение хотя бы к одной индивидуальной роли удалённого доступа $nr \in NR$, уровень конфиденциальности которой равен уровню доступа субъект-сессии: для любой субъект-сессии $wsn_i \in CSN$ существует роль удалённого доступа $user(wsn_i)\_nr\_f_s(wsn_i)\_i\_high \in NR$, такая, что $(wsn_i, nr, read_a) \in AA_i$, $1 \leqslant i \leqslant \omega$.

Замечание 13. Субъект-сессии может быть предоставлен доступ на запись к сущности только в случае, когда её уровень целостности не выше текущего уровня целостности субъект-сессии. Доступ субъект-сессии wsn_i на запись к сущности-«макроконтейнеру», предоставляемый через владение ролями удалённого доступа, интерпретируется как возможность выполнения административных функций (например, настройка сетевых параметров рабочей станции сети). Выполнять подобные функции могут субъект-сессии, функционирующие от имени учётных записей доверенных пользователей, то есть субъект-сессии с высоким уровнем целостности, что соответствует мандатной политике целостности.

С целью задания порядка предоставления прав доступа к сущностям для запрещающих ролей и доступов субъект-сессий к ним, а также для описания правил мандатного управления доступа и целостности, связанных с использованием запрещающих ролей, сделаем предположение.

Предположение 3. В рамках МРОКС ДП-модели выполняются следующие условия, которые дополняют и уточняют (в части, касающейся множества запрещающих ролей) условия, заданные в МРОСЛ ДП-модели [1].

У с л о в и е 1. Права доступа запрещающих ролей к сущностям являются более приоритетными при проверке выполнения условий де-юре правил преобразования состояний сети и отдельной рабочей станции, в частности, по сравнению с правами доступа, которыми обладают роли к тем же сущностям.

У с л о в и е 2. При создании каждая субъект-сессия получает доступ на чтение к запрещающим ролям с уровнем конфиденциальности, не превосходящим её уровня доступа. При этом к запрещающим ролям обладает соответствующим правом доступа индивидуальная административная роль учётной записи пользователя, от имени которой функционирует данная субъект-сессия: для субъект-сессии $s \in S$ выполняется условие: если $ur \in UR$, $f_r(ur) \leqslant f_s(s)$ и $(ur, read_r) \in APA(user(s)\_admin\_i\_i_s(s))$, то $(s, ur, read_a) \in AA$.

У с л о в и е 3. Запрещающие роли обладают высоким уровнем целостности: для любой запрещающей роли $ur \in UR$ справедливо равенство $i_r(ur) = i\_high$.

У с л о в и е 4. Для получения доступа на запись к запрещающей роли субъект-сессии (или кооперирующей с ней другой субъект-сессии) с уровнем доступа, равным c, необходимо получение доступа на запись к сущности c_i_entity.

У с л о в и е 5. Административная роль может содержать административные права доступа на владение или запись к запрещающей роли только в том случае, когда административная роль обладает высоким уровнем целостности: для административной роли $ar \in AR$ и запрещающей роли $ur \in UR$, если $(ur, \alpha_r) \in APA(ar)$, то $i_r(ar) = i\_high$, где $\alpha_r \in \{own_r, write_r\}$.

У с л о в и е 6. Субъект-сессия может создать, удалить, переименовать запрещающую роль, получить административный доступ на запись к запрещающей роли, изменить иерархию запрещающих ролей или изменить права доступа административных ролей к ней, только если эта запрещающая роль обладает высоким уровнем целостности: для субъект-сессии $s \in S$, запрещающей роли $ur \in UR$, если $(s,\ ur,\ write_a) \in AA$, то $i_r(ur) = i_s(s) = i\_high$.

Все остальные условия мандатного управления доступом и мандатного контроля целостности для запрещающих ролей задаются аналогично существующим условиям для ролей в МРОСЛ ДП-модели.

Замечание 14. Условия 5 и 6 предположения 3 не запрещают субъект-сессии, обладающей любым уровнем целостности, получить доступ на чтение к запрещающей роли. Однако, согласно условию 6, добавить или удалить соответствующее право доступа для административной роли может только субъект-сессия с высоким уровнем целостности (функционирующая от имени учётной записи пользователя с высоким уровнем целостности), что позволяет данной субъект-сессии изменять права доступа запрещающих ролей к сущностям с любым уровнем целостности.

3. Де-юре правила преобразования состояний

Зададим правила преобразования состояний, в исходном или результирующем состоянии которых используются или изменяются функции и множества, содержащие элементы, заданные для различных рабочих станций, функционирующих в рамках сети. Помимо этого приведём пример того, как изменяются правила преобразования состояний отдельных рабочих станций сети, заданные в МРОСЛ ДП-модели [1], с учётом новых элементов модели.

Де-юре правила преобразования состояний МРОКС ДП-модели

Правило: $dest\_access\_read(x_i, x_j, y_j)$
Исходное состояние: $1 \leqslant i \leqslant \omega$, $1 \leqslant j \leqslant \omega$, $x_i, x_j \in SN$, $wsn_i, wsn_j \in CSN : csn\_accessory(x_i) = = wsn_i$, $csn\_accessory(x_j) = wsn_j$, $y_i \in E_j$, [существуют $os_i \in OS_i$ и $os_j \in OS_j$: $(os_i, os_j) \in \in csn\_connected(wsn_i, wsn_j)$, $\{(x_i, os_i, \alpha_a) : \alpha_a \in \{read_a, write_a\}\} \subset A$, $\{(x_j, os_j, \alpha_a) : \alpha_a \in \{read_a,$ $write_a\}\} \subset A$], [не существует $ur \in UR$, $network\_role(ur) =$ `true`, такой, что $(x_i, ur, read_a) \in AA$ и $(y_i, read_r) \in PA_j(ur)$] и [существует $r \in R \cup AR$, $network\_role(r) =$ `true`, такая, что $(x_i, r, read_a) \in AA$ и $(y_j, read_r) \in PA_j(r)$], [и либо (существует контейнер $c_j \in C_j$, такой, что $execute\_container(x_j, c_j, y_j) =$ `true` и $f_e(y_j) \leqslant f_s(x_j)$), либо $(x_j, downgrade\_admin\_role,$ $read_a) \in AA$]
Результирующее состояние: $A' = A \cup \{(x_j, y_j, read_a), (x_i, y_j, read_a)\}$ $(A'_j = A_j \cup \{(x_j, y_j, read_a)\},$ $A'_{ij} \cup \{(x_i, y_j, read_a)\})$
Правило: $dest\_access\_write(x_i, x_j, y_j)$
Исходное состояние: $1 \leqslant i \leqslant \omega$, $1 \leqslant j \leqslant \omega$, $x_i, x_j \in SN$, $wsn_i, wsn_j \in CSN$: $csn\_accessory(x_i) = = wsn_i$, $csn\_accessory(x_j) = wsn_j$, $y_i \in E_j$, [существуют $os_i \in OS_i$ и $os_j \in OS_j$: $(os_i, os_j) \in \in csn\_connected(wsn_i, wsn_j)$, $\{(x_i, os_i, \alpha_a) : \alpha_a \in \{read_a, write_a\}\} \subset A$, $\{(x_j, os_j, \alpha_a) : \alpha_a \in \{read_a,$ $write_a\}\} \subset A$], [не существует $ur \in UR$, $network\_role(ur) =$ `true`, такой, что $(x_i, ur, read_a) \in AA$ и $(y_i, write_r) \in PA_j(ur)$] и [существует $r \in R \cup AR$, $network\_role(r) =$ `true`, такая, что $(x_i, r, read_a) \in AA$ и $(y_j, write_r) \in PA_j(r)$], [$i_e(y_j) \leqslant i_s(x_i)$ и либо (существует контейнер $c_j \in C_j$, такой, что $execute\_container(x_j, c_j, y_j) =$ `true` и, если $y_j \in E\_HOLE$, то $f_s(x_i) \leqslant f_e(y_j)$, иначе $f_e(y_j) = f_s(x_i)$), либо $(x_i, downgrade\_admin\_role, read_a) \in AA$], [если $i_e(y_j) = i\_high$, то $(x'_i, f_s(x'_i)\_i\_entity, write_a) \in A$]

Продолжение таблицы

Результирующее состояние: $A' = A \cup \{(x_j, y_j, write_a), (x_i, y_j, write_a)\}$ ($A'_j = A_j \cup \{(x_j, y_j, write_a)\}$, $A'_{ij} \cup \{(x_i, y_j, write_a)\}$)
Правило: $csn\_connect(x_i, os_i, x_j, y_j, os_j)$
Исходное состояние: $\upsilon < \omega$, $1 \leqslant i \leqslant \omega$, $1 \leqslant j \leqslant \omega$, $y_j \in OS_j$, [$wsn_i, wsn_j \in CSN$: $csn\_accessory(x_i) = wsn_i, csn\_accessory(x_j) = wsn_j, csn\_initialized(wsn_i) =$ `true`], [существует $nr \in NR$, такая, что $(wsn_i, nr, read_a) \in AA_i$ и $(em_j, execute_r) \in PA_j(nr)$, $CCRI(em_j) =$ = `false`, $CCR(em_j) =$ `false`], [не существует $nur \in NUR$, такой, что $(csn_i, nur, read_a) \in AA_i$ и $(em_j, execute_r) \in PA_j(nur)$], [существует $nr' \in NR$, такая, что $(csn_j, nr', read_a) \in AA_j$ и $(em_i, execute_r) \in PA_i(nr')$, $CCRI(em_i) =$ `false`, $CCR(em_i) =$ `false`], [не существует $nur' \in NUR$, такой, что $(wsn_j, nur', read_a) \in AA_j$ и $(em_i, execute_r) \in PA_i(nur')$], [$\{(x_i, os_i, \alpha_a) : \alpha_a \in \{read_a, write_a\}\} \subset A_i$, $(x_j, y_j, read_a) \subset A_i$], [либо $f_e(os_i) = f_e(y_j) = f_s(x_j)$, либо $f_e(os_i) < f_e(y_j) \leqslant f_s(x_j)$ и $(x_j, downgrade\_admin\_role, read_a) \in AA_j$], $Constraint_{PA_j}(PA'_j) =$ `true`
Результирующее состояние: $csn\_connected'(wsn_i, wsn_j) = csn\_connected(wsn_i, wsn_j) \cup (os_i, os_j)$, $OS_j = OS_j \cup os_j$, $f_e(os_j) = f_e(os_i)$, $PA'_j(user(x_j)\_c\_f_s(x_j)\_i_s(x_j)) = PA_j(user(x_j)\_c\_f_s(x_j)\_i_s(x_j)) \cup \{(os_j, own_r)\}$, $H'_{E_j}(H^{-1}_{E_j}(y_j)) =$ $= H_{E_j}(H^{-1}_{E_j}(y_j)) \cup os_j, H'_{E_j}(os_j) = \varnothing$, [если $csn\_initialized(wsn_i) =$ `false`,то $S' = S \cup S_i, E' = E \cup E_i$, $\upsilon = \upsilon + 1$, $csn\_initialized'(wsn_i) =$ `true`, $A' = A \cup A_i$, $AA' = AA \cup AA_i$]
Правило: $csn\_disconnect(x_i, os_i, x_j, os_j)$
Исходное состояние: $1 \leqslant i \leqslant \omega$, $1 \leqslant j \leqslant \omega$, $x_i \in S_i$, $x_j \in S_j$, $os_i \in OS_i$, $os_j \in OS_j$, [$wsn_i, wsn_j \in CSN : csn\_accessory(x_i) = csn_i, csn\_accessory(x_j) = wsn_j, csn\_initialized(wsn_i) =$ $= csn\_initialized(wsn_j) =$ `true`], $(os_i, os_j) \in csn\_connected(wsn_i, wsn_j)$, $i_e(os_i) \leqslant i_s(x_i)$ и либо $f_e(os_i) = f_s(x_i)$, либо $(x_i, downgrade\_admin\_role, read_a) \in AA$
Результирующее состояние: $OS_i = OS_i \setminus \{os_i\}, A' = A \setminus \{(s, os_i, \alpha_a) : s \in S, \alpha_a \in R_a\}$, $csn\_connected'(wsn_i, wsn_j) = csn\_connected(wsn_i, wsn_j) \setminus \{(os_i, os_j)\}$, [если $i \neq j$ и для любого $1 \leqslant k \leqslant n$, $k \neq j$ выполняется $csn\_connected(wsn_k, wsn_j) = \varnothing$, то $S' = S \setminus S_j, E' = E \setminus E_j$, $AA' = AA \setminus AA_j$, $\upsilon = \upsilon - 1$, $A' = A \setminus (A_j \cup A_{i_j} \cup A_{k_j})$, для $r \in R \cup UR \cup AR$ выполняется $PA'(r) = PA(r) \setminus 2^{P_j}$, $csn\_initialized'(wsn_j) =$ `false`], для $s \in S'$ выполняются равенства $user'(s) = user(s)$ и $HS'(s) = HS(s)$
Правило: $listen\_socket(x, y)$
Исходное состояние: $x \in S$, $y \in OS$, $(x, listen\_socket) \in AA$, существует $r \in R$, такая, что $network\_role(r) =$ `true`, $(x, r, read_a) \in AA$ и $(y, read_r) \in PA(r)$ и [для любой $ur \in UR$: $network\_role(ur) =$ `true`, $(x, ur, read_a) \in AA$ верно $(y, read_r) \notin PA(ur)$], [либо $f_s(x) = f_e(y)$, либо $(x, downgrade\_admin\_role, read_a) \in AA$]
Результирующее состояние: $is\_listening(y) =$ `true`
Правило: $access\_read(x, x', y)$
Исходное состояние: $x, x' \in S$, $y \in E \cup R \cup UR \cup AR$, существует $r \in R \cup AR : (x, r, read_a) \in AA$ и для любой $ur \in UR$: $(x, ur, read_a) \in AA$, [если $y \in E$, то $(y, read_r) \in PA(r)$, $(y, read_r) \notin PA(ur)$ и либо (существует контейнер $c \in C$, такой, что $execute\_container(x, c, y) =$ `true` и $f_e(y) \leqslant$ $\leqslant f_s(x)$), либо $(x, downgrade\_admin\_role, read_a) \in AA$, и, если $y \in OS$, то $network\_role(r) =$ $network\_role(ur) =$ `true`], [если $y \in R \cup UR \cup AR$, то $(y, read_r) \in APA(r)$, $Constraint_{AA}(AA') =$ = `true`, (если $y \in UR$, то для $e \in]y[$ либо $(x, e, read_a) \in A$, либо $(x, e, write_a) \in A$), (либо $f_r(y) \leqslant$ $\leqslant f_s(x)$, либо $(x, downgrade\_admin\_role, read_a) \in AA$) и если $y \in R \cup AR$, то $i_r(y) \leqslant i_s(x)$], [если $y \in R \cup AR$ и $i_r(y) = i\_high$, то $(x', f_s(x)\_i\_entity, write_a) \in A$], [если $y \in NR \cup NUR$, то $x \in CSN$]
Результирующее состояние: если $y \in E$, то $A' = A \cup \{(x, y, read_a)\}$, если $y \in R \cup UR$, то $AA' = AA \cup \{(x, y, read_a)\}$, если $y \in AR$, то $AA' = AA \cup \{(x, y, read_a)\} \cup \{(x, ur, read_a)$: [$x \in S$, при этом если $ur \in NUR$, то $x \in CSN$], $(ur, read_r) \in APA(y)$ и $f_r(ur) \leqslant f_s(x)\}$
Правило: $access\_write(x, x', y)$
Исходное состояние: $x, x' \in S$, $y \in E \cup R \cup UR \cup AR$, существует $r \in R \cup AR$: $(x, r, read_a) \in AA$ и для любой $ur \in UR$: $(x, ur, read_a) \in AA$, [если $y \in E$, то $(y, write_r) \in PA(r)$, $(y, read_r) \notin PA(ur)$, $i_e(y) \leqslant i_s(x)$ и либо (существует контейнер $c \in C$, такой, что $execute\_container(x, c, y) =$ `true`, и $f_e(y) = f_s(x)$), либо $(x, downgrade\_admin\_role, read_a) \in AA$, и, если $y \in OS$, то $network\_role(r) =$ $= network\_role(ur) =$ `true`], [если $y \in R \cup UR \cup AR$, то $(y, write_r) \in APA(r)$, $i_r(y) \leqslant i_s(x)$, $Constraint_{AA}(AA') =$ `true`, (для $e \in]y[$ либо $(x, e, read_a) \in A$, либо $(x, e, write_a) \in A$), (либо $f_r(y) =$ $= f_s(x)$, либо $(x, downgrade\_admin\_role, read_a) \in AA$)], [если ($y \in E$ и $i_e(y) = i\_high$) или ($y \in$ $\in R \cup UR \cup AR$ и $i_r(y) = i\_high$), то $(x', f_s(x)\_i\_entity, write_a) \in A$]
Результирующее состояние: если $y \in E$, то $A' = A \cup \{(x, y, write_a)\}$, $AA' = AA$, если $y \in R \cup UR \cup AR$, то $AA' = AA \cup \{(x, y, write_a)\}$, $A' = A$

Окончание таблицы

Правило: $grant\_rights(x, x', r, (r, \alpha_{r_j}) : 1 \leqslant j \leqslant k)$
Исходное состояние: $x, x' \in S$, $y \in E$, $r \in R \cup UR \cup AR$, $\alpha_{r_j} \in \{write_r, read_r, execute_r\}$, $(x, r, write_a) \in AA$, $i_e(y) \leqslant i_s(x)$, [существует $r' \in R \cup AR$: $(x, r', read_a) \in AA$, $(y, own_r) \in PA(r')$, и, если $y \in OS$, то $network\_role(r) = \texttt{true}$], [либо (существует контейнер $c \in C$, такой, что $execute\_container(x, c, y) = \texttt{true}$, и $f_e(y) = f_s(x)$), либо $(x, downgrade\_admin\_role, read_a) \in AA$], [если $\alpha_{r_j} = write_r$, то $i_e(y) \leqslant i_r(r)$], $Constraint_{PA}(PA') = \texttt{true}$, [если $i_e(y) = i\_high$, то $(x', f_s(x)\_i\_entity, write_a) \in A$], [если $r \in NR \cup NUR$, то $(x, admin\_net, read_a) \in AA$], где $1 \leqslant j \leqslant k$
Результирующее состояние: $PA'(r) = PA(r) \cup \{(y, \alpha_{r_j}) : 1 \leqslant j \leqslant k\}$, и для $r' \in (R \cup AR) \setminus \{r\}$ выполняется равенство $PA'(r') = PA(r')$
Правило: $grant\_admin\_rights(x, x', r, (ar, \alpha_{r_j}) : 1 \leqslant j \leqslant k)$
Исходное состояние: $x, x' \in S$, $r \in R \cup UR \cup AR$, $ar \in AR$, $\alpha_r \in \{write_r, read_r\}$, $(x, ar, write_a) \in$ $\in AA$, $i_r(r) \leqslant i_s(x)$, [если $r \in R \cup AR$ или ($r \in UR$ и существует $j : \alpha_{r_j} = write_r$), то $i_r(r) \leqslant i_r(ar)$], [если $r \in R \cup UR$, то $(x, roles\_admin\_role, read_a) \in AA$, если $r \in AR$, то $(x, admin\_roles\_admin\_role, read_a) \in AA$, если $r \in NR \cup NUR$, то $(x, admin\_net, read_a) \in AA$ и (если существует $j : \alpha_{r_j} = read_r$, то $ar \in \{u\_admin\_i\_high : u \in WS\}$)], [либо $f_r(r) = f_s(x)$, либо $(x, downgrade\_admin\_role, read_a) \in AA$], $Constraint_{APA}(APA') = \texttt{true}$, $Constraint_{AA}(AA') =$ $= \texttt{true}$, [если $i_r(r) = i\_high$, то $(x', f_s(x)\_i\_entity, write_a) \in A$], где $1 \leqslant j \leqslant k$
Результирующее состояние: $APA'(ar) = APA(ar) \cup \{(r, \alpha_{r_j}) : \alpha_{r_j} = write_r, 1 \leqslant j \leqslant k\} \cup$ $\cup \{(r', \alpha_{r_j})$: $\alpha_{r_j} = read_r, r' \leqslant r, 1 \leqslant j \leqslant k\}$, для $ar' \in AR \setminus \{ar\}$ выполняется равенство $APA'(ar') =$ $= APA(ar')$, если $r \in R \cup AR$, то $AA' = AA$, если $r \in UR$ и $\alpha_{r_j} = read_r$, то $AA' = AA \cup \{(s, r, read_a)$: [$s \in S$, при этом (если $r \in NUR$, то $s \in CSN$)], $(s, ar, read_a) \in AA$ и $f_r(r) \leqslant f_s(s)\}$
Правило: $create\_first\_session(x, x', u, y, z, zc, zi)$
Исходное состояние: $x, x' \in S$, $u \in U$, $y \in E$, $z \in S$, [существует $r \in R \cup AR$, такая, что $(x, r, read_a) \in AA$ и $(y, execute_r) \in PA(r)$, для любой $ur \in UR : (x, ur, read_a) \in AA$ и $(y, execute_r) \notin PA(ur)$], существует контейнер $c \in C$, такой, что $execute\_container(x, c, y) =$ $= \texttt{true}$, $f_e(y) \leqslant min(f_s(x), f_u(u))$, $zc \leqslant f_u(u)$, $zi \leqslant min(i_u(u), i_e(y))$, [либо $f_s(x) \leqslant zc$, либо $(x, downgrade\_admin\_role, read_a) \in AA$], [для $e \in]u[$ либо $(x, e, read_a) \in A$, либо $(x, e, write_a) \in A$], [если $z_i = i\_high$, то $(x', f_s(x)\_i\_entity, write_a) \in A$], [если $u \in WS$ и $z \in S' \cap CSN$, то $z_i = i\_high$ и $z_c \leqslant f_e(em)$ для сущности-«макроконтейнера» $em \in EM$]
Результирующее состояние: $S' = S \in z$, $f'_s(z) = zc$, $i'_s(z) = zi$, $user'(z) = u$, если $u \in L_U$, то $L'_S = L_S \cup z$, иначе $N'_S = N_S \cup z$, для $s \in S$ выполняется $user'(s) = user(s)$, $f'_s(s) = f_s(s)$, $i'_s(s) = i_s(s)$, если $z \in CSN$, то $csn\_initialized(z) = \texttt{false}$, $AA' = AA \cup \{(z, u\_admin\_i\_zi,$ $read_a), (z, u\_c\_zc\_zi, write_a), (z, common\_zc\_zi, write_a)\} \cup \{(z, u\_c\_l\_li, read_a), (z, common\_l\_li,$ $read_a)$: $l \leqslant zc$ и $li \leqslant zi\} \cup \{(z, u\_nr\_zc\_zi, read_a)$: $u \in WS$ и $z \in CSN\} \cup \{(z, ur, read_a)$: [$ur \in UR$, при этом (если $ur \in NUR$, то $z \in CSN$)], $(ur, read_r) \in APA(user(z)\_admin\_i\_zi)$ и $f_r(ur) \leqslant f_s(z)\}$, $PA'(u\_c\_zc\_zi) = PA(u\_c\_zc\_zi) \cup \{(z, own_r)\}$, и для $r' \in R \setminus \{u\_c\_zc\_zi\}$ выполняется $PA'(r') = PA(r')$, $[z] = fa(u, y)$, $]z[= fp(u, y)$, $HS'(z) = \varnothing$, для $s \in S$ выполняется $HS(s) = HS(s)$
Правило: $create\_role(x, x', r, rc, ri, re, name, rz)$
Исходное состояние: $x, x' \in S$, $r \in R \cup UR \cup AR$, [если $rz \in R \cup UR$, то $(x, roles\_admin\_role, \alpha_a) \in$ $\in AA$, если $rz \in AR$, то $(x, admin\_roles\_admin\_role, \alpha_a) \in AA$, где $\alpha_a \in \{read_a, write_a\}$], $(x, rz, write_a) \in AA$, $re \subset RE$, $name \in NAME \setminus (\{\text{“”}\} \cup \{role\_name(rz, r') : r' \in HR(rz)\})$, [либо $r = f_r(rz) = f_s(x)$, либо $r \leqslant f_r(rz)$ и $(x, downgrade\_admin\_role, read_a) \in AA$], [$ri \leqslant$ $\leqslant \min(i_r(rz), i_s(x))$ и если $rz \in UR$, то $ri = i\_high$], [для $e \in re$ выполняется $f_e(e) = rc$, $i_e(e) = ri$], $Constraint_{APA}(APA') = \texttt{true}$, $Constraint_{PA}(PA') = \texttt{true}$, [если $i_r(rz) = i\_high$, то $(x', f_s(x)\_i\_entity, write_a) \in A$], [если $rz \in NR \cup NUR$, то $(x, admin\_net, read_a) \in AA$]
Результирующее состояние: если $rz \in R$, то $R' = R \in \{r\}$, $APA'(roles\_admin\_role) =$ $= APA(roles\_admin\_role) \cup \{(r, own_r), (r, execute_r)\} \cup \{(r, read_r) : (rz, read_r) \in$ $\in APA(roles\_admin\_role)$ и $r \notin NR \cup NUR\}$, если $rz \in AR$, то $AR' = AR \cup \{r\}$, $APA'(admin\_roles\_admin\_role) = APA(admin\_roles\_admin\_role) \cup \{(r, own_r), (r, execute_r)\} \cup$ $\cup \{(r, read_r) : (rz, read_r) \in APA(admin\_roles\_admin\_role)$ и $r \notin NR \cup NUR\}$, для $ar \in AR \setminus \{admin\_roles\_admin\_role, roles\_admin\_role\}$ выполняется $APA'(ar) = APA(ar) \cup$ $\cup \{(r, execute_r)\} \cup \{(r, read_r) : (rz, read_r) \in APA(ar)$ и (если $r \in NR \cup NUR$, то $ar \in \{u\_admin\_li :$ $u \in U, li \in LI\}$)$\}$, $f'_r(r) = rc$, $i'_r(r) = ri$, $CCR'(r) = \texttt{false}$, $CCRI'(r) = \texttt{false}$, $role\_name'(rz, r) =$ $= name$, $]r[= re$, $shared\_container'(r) = \texttt{true}$, $PA'(r) = \varnothing$, $HR'(rz) = HE(z) \cup \{r\}$, $HR'(r) = \varnothing$, для $r' \in (R \cup AR) \setminus \{rz\}$ выполняется равенство $HR'(r') = HR(r')$

Де-юре правила $dest\_access\_read(x_i, x_j, y_j)$, $dest\_access\_write(x_i, x_j, y_j)$ позволяют субъект-сессии x_i, функционирующей на рабочей станции wsn_i, получить доступ на чтение или запись к удалённому ресурсу y_j сети, расположенному на рабочей станции wsn_j, через удалённое сетевое соединение с субъект-сессией x_j. Для возможности обращения к ресурсам рабочей станции wsn_j необходимо существование сокетов os_i и os_j, через которые осуществляется сетевое соединение субъект-сессий x_i и x_j. При этом субъект-сессии x_i и x_j должны обладать доступом на чтение и запись к сокетам os_i и os_j соответственно. Вместе с тем субъект-сессия x_j должна обладать сетевой ролью r, имеющей соответствующее право доступа ($read_r$ или $write_r$) к сущности y_j, и не должна содержать в качестве текущей какую-либо сетевую запрещающую роль ur, обладающую тем же рассматриваемым правом доступа к y_j, что и роль r. Получение запрашиваемого доступа к y_j возможно только с учётом прав доступа субъект-сессии x_j к сущностям-контейнерам, содержащим y_j. Для получения доступа на запись к сущности y_j уровень целостности субъект-сессии x_i должен быть не меньше уровня целостности y_j, а её уровень конфиденциальности должен совпадать с уровнем доступа x_j, за исключением случая, когда предоставляется доступ к объекту-«дырке», и уровень кофиденциальности y_j должен быть не ниже уровня доступа субъект-сессии x_j. При получении доступа на чтение уровень доступа субъект-сессии x_j должен быть не ниже уровня конфиденциальности сущности y_j. Для возможности нарушения требований мандатного управления доступом субъект-сессия x_i должна обладать доступом на чтение к административной роли $downgrade\_admin\_role$.

Де-юре правило $csn\_connect(x_i, os_i, x_j, y_j, os_j)$ позволяет субъект-сессии x_i, соответствующей рабочей станции сети wsn_i, установить сетевое соединение с субъект-сессией x_j, соответствующей рабочей станции сети wsn_j, посредством сокетов os_i и os_j соответственно и «слушающего» сокета y_j. При этом необходимо, чтобы в множестве текущих ролей субъект-сессии wsn_i содержалась роль удалённого доступа ur и отсутствовала запрещающая роль удалённого доступа nur, обладающие правом доступа $execute_r$ к сущности-«макроконтейнеру» удалённой рабочей станции em_j. Субъект-сесиия x_i должна обладать доступами на чтение и запись к объекту-сокету os_i, а субъект-сессия x_j — только правом доступа на чтение к «слушающему» сокету y_j. Для возможности установления сетевого соединения уровень доступа субъект-сессии x_j должен быть не ниже уровня конфиденциальности «слушающего» сокета субъект-сесии y_j. При этом либо уровень конфиденциальности x_i должен быть равен уровню доступа субъект-сесии x_j, а также уровню конфиденциальности «слушающего» сокета y_j, либо для возможности нарушения этого требования субъект-сессия x_j должна обладать доступом на чтение к административной роли $downgrade\_admin\_role$. Вместе с тем множество текущих ролей субъект-сессий wsn_j должно содержать роль удалённого доступа ur и не включать запрещающую роль удалённого доступа nur, обладающие правом доступа $execute_r$ к сущности-«макроконтейнеру» удалённой рабочей станции em_i (разрешать ответный входящий трафик от удалённой рабочей станции wsn_j). В силу того, что атрибуты CCR и $CCRI$ для сущностей-«макроконтейнеров» имеют значение `false`, их уровни конфиденциальности и целостности в дальнейшем не учитываются при получении доступа внутрь этих контейнеров. При выполнении всех условий на рабочей станции wsn_j создаётся сокет os_j, участвующий в дальнейшем в сетевом соединении; его уровни кофиденциальности и целостности совпадают с уровнями конфиденциальности и целостности сокета os_i. В случае, если субъект-сессия wsn_i не была инициализирована в сети (устанавливается первое сетевое соеди-

нение, $csn\_initialized(wsn_i) = \texttt{false}$), то множества, описывающие состояние сети, дополняются множествами инициализируемой рабочей станции.

Де-юре правило $csn\_disconnect(x_i, os_i, x_j, os_j)$ позволяет субъект-сессии x_i, функционирующей на рабочей станции wsn_i, удалить (закрыть) сокет os_i и разорвать сетевое соединение, осуществляемое через сокет os_j, с субъект-сессией x_j, функционирующей на рабочей станции сети wsn_j. Для успешного выполнения запрашиваемого действия необходимо, чтобы уровень доступа субъект-сессий x_i был равен уровню конфиденциальности сокета os_i. Для возможности нарушения этого требования субъект-сессия x_i должна обладать ролью $downgrade\_admin\_role$. Вместе с тем уровень целостности сокета os_i не должен превосходить уровня целостности субъект-сессии x_i. В случае выполнения всех условий, а также если разорванное соединение было последним для рабочей станции сети wsn_j, из множеств, описывающих состояние сети, удаляются множества рабочей станции wsn_j.

Обоснуем утверждения о том, что способ задания запрещающих ролей с учетом предъявляемых требований в рамках МРОКС ДП-модели не повлияет на корректность задания правил преобразования состояний МРОСЛ ДП-модели [1] и что условия и результаты заданных правил преобразования соответствуют предположениям 1–3.

Утверждение 1. Пусть G_0 — начальное состояние системы $\Sigma(G^*, OP, G_0)$, удовлетворяющее условиям предположений 1 и 2. Тогда для любой траектории $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$, где $N \geqslant 1$, в состоянии G_N выполняются условия предположений 1 и 2 и переход $G_{N-1} \vdash_{op_N} G_N$ удовлетворяет условиям этих предположений.

Доказательство.

Индукция по длине N траектории функционирования системы.

Пусть $N = 0$, тогда по условию утверждения состояние G_0 удовлетворяет условиям предположений 1 и 2.

Пусть $N > 0$ и утверждение верно для всех траекторий длины $0 \leqslant L < N$. Пусть $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$ — траектория функционирования системы. По предположению индукции состояние G_{N-1} удовлетворяет условиям предположений 1 и 2, а также каждый переход $G_{i-1} \vdash_{op_i} G_i$ удовлетворяет условиям этих предположений, где $1 \leqslant i < N$.

Рассмотрим правило преобразования состояний op_N. Если условия его применения не выполняются в состоянии G_{N-1}, то по определению правил преобразования состояний справедливо равенство $G_{N-1} = G_N$ и по предположению индукции состояние G_N удовлетворяет условиям предположений 1 и 2 и переход $G_{N-1} \vdash_{op_N} G_N$ удовлетворяет условиям этих предположений. Пусть условия применения правила преобразования состояний op_N выполняются в состоянии G_{N-1}.

Если op_N является де-факто правилом (за исключением правила $de\_facto\_op(s, op(x, x', \ldots))$), то в результате его применения могут измениться только значение функции $de\_facto\_own_{N-1}$ и множество информационных потоков F_{N-1}. В предположениях 1 и 2 не содержится требований к множеству информационных потоков и фактическому владению субъект-сессий. Таким образом, если op_N является де-факто правилом (за исключением правила $de\_facto\_op(s, op(x, x', \ldots))$), то по предположению индукции состояние G_N удовлетворяет условиям предположений 1 и 2 и переход $G_{N-1} \vdash_{op_N} G_N$ удовлетворяет условиям этих предположений.

Если op_N является де-факто правилом $de\_facto\_op(s_i,\ s_j, op(x_i, x_j, \ldots))$, то результаты его применения совпадают с результатами применения де-юре правила $op(x_i, x_j, \ldots)$, за исключением множества информационных потоков F_{N-1}, в которое

добавляются информационные потоки по времени. Однако к таким информационным потокам также не предъявляются требования в предположениях 1 и 2, поэтому далее при обосновании шага индукции будем считать, что op_N — де-юре правило.

Пусть op_N — де-юре правило. Обоснуем выполнение условий предположений 1 и 2 в состоянии G_N и при переходе $G_{N-1} \vdash_{op_N} G_N$.

Административное право доступа $read_r$ к ролям или запрещающим ролям удалённого доступа задаётся при применении правил вида $create\_user(x, x', u, uc, ui, ue)$ (административное право доступа $read_r$ к индивидуальным ролям удалённого доступа задаётся аналогично индивидуальным ролям), $set\_user\_labels(x, x', u, uc, ui, ue)$ (для индивидуальных ролей удалённого доступа задаётся аналогично индивидуальным ролям), $grant\_admin\_rights(x, x', ar, \{(r, \alpha_{r_j}) : 1 \leqslant j \leqslant k\})$, $create\_role(x, x', r, rc, ri, re, name, rz)$, $create\_hard\_link\_role(x, x', r, name, rz)$ (аналогично правилу преобразования $create\_role$). В состоянии G_{N-1} при применении правила $grant\_admin\_rights$ и в состоянии G_N в результате применения остальных правил проверяется условие назначения права доступа $read_r$ для ролей и запрещающих ролей удалённого доступа только для индивидуальных административных ролей учётных записей пользователей. Следовательно, с учётом предположения индукции в состоянии G_N и при переходе $G_{N-1} \vdash_{op_N} G_N$ выполнено условие 1 предположения 1.

Заметим, что задание (и удаление) права доступа $read_r$ к индивидуальным ролям удалённого доступа организуется аналогично заданию (удалению) права доступа $read_r$ к индивидуальным ролям в МРОСЛ ДП-модели [1].

Таким образом, выполнение условия 2 предположения 1 МРОКС ДП-модели следует из утверждения 1 МРОСЛ ДП-модели [2].

Субъект-сессии могут получить административный доступ на чтение к запрещающим ролям удалённого доступа с использованием правил вида $create\_session(x, x', y, z)$, $create\_first\_session(x, x', u, y, z, zc, zi)$, $grant\_admin\_rights(x, x', ar, \{(r, \alpha_{r_j}) : 1 \leqslant \leqslant j \leqslant k\})$, $access\_read(x, x', y)$, а к ролям удалённого доступа — только посредством правила $access\_read(x, x', y)$. В результате применения этих правил получить доступ $read_a$ к запрещающим ролям удалённого доступа могут только субъект-сессии, входящие в множество CSN. Вместе с тем в условиях применения правила последнего вида проверяется, что доступ на чтение к роли (а также и к запрещающей роли) удалённого доступа может быть предоставлен только субъект-сессиям рабочих станций сети.

Таким образом, с учётом предположения индукции в состоянии G_N и при переходе $G_{N-1} \vdash_{op_N} G_N$ выполнено условие 3 предположения 1.

Для создания, удаления, переименования, изменения иерархии ролей удалённого доступа, назначения к ним прав доступа административных ролей, а также для изменения их прав доступа к сущностям-«макроконтейнерам» субъект-сессия может использовать все правила МРОСЛ ДП-модели, которые используются для тех же действий с ролями, в условиях применения которых проверяется наличие административной роли $admin\_net$ в множестве текущих ролей данной субъект-сессии (для них задаются условия аналогично правилу $grant\_rights$). Следовательно, с учётом предположения индукции в состоянии G_N и при переходе $G_{N-1} \vdash_{op_N} G_N$ выполнено условие 4 предположения 1.

Изменение уровней целостности и значений атрибутов $CCRI$ и CCR контейнера выполняется с помощью правил преобразования вида $set\_entity\_labels(x, x', y, yc, yi)$ и $set\_container\_attr(x, x', y, ccr, ccri, t)$, в условиях применения которых исключается возможность изменения уровней целостности и значений атрибутов CCR и $CCRI$ для сущности-«макроконтейнера».

Субъект-сессия может изменить уровень целостности учётной записи пользователя при применении правила вида $set\_user\_labels(x, x', u, uc, ui, ue)$, условия применения которого запрещают задавать для учётной записи-компьютера уровень целостности, отличный от i_high.

Таким образом, в силу условий, заданных в МРОСЛ ДП-модели, которые запрещают создавать сущность с уровнем конфиденциальности, превосходящим уровень конфиденциальности сущности-контейнера, её содержащего, и с учётом предположения индукции в состоянии G_N и при переходе $G_{N-1} \vdash_{op_N} G_N$ выполнено условие 1 предположения 2.

Создание субъект-сессии рабочей станции сети возможно с использованием правил вида $create\_session(x, x', y, z)$, $create\_first\_session(x, x', u, y, z, zc, zi)$, в условиях применения которых проверяется, что уровень целостности создаваемой субъект-сессии равен i_high, а уровень конфиденциальности (для правила $create\_session$ уровни конфиденциальности и целостности создаваемой субъект-сессии совпадают с соответствующими уровнями текущей субъект-сессии) не превосходит уровня конфиденциальности сущности-«макроконтейнера». В силу того, что мандатные атрибуты субъект-сессий не изменяются в процессе функционирования системы, и с учётом предположения индукции в состоянии G_N и при переходе $G_{N-1} \vdash_{op_N} G_N$ выполнено условие 2 предположения 2.

В соответствии с условиями правила $create\_role(x, x', r, rc, ri, re, name, rz)$, при создании ролей удалённого доступа им назначается уровень конфиденциальности i_high, и в модели отсутствуют правила, позволяющие изменять уровень целостности роли впоследствии. При создании индивидуальных ролей удалённого доступа учётных записей пользователей с использованием правил вида $create\_user(x, x', u, uc, ui, ue)$ и $set\_user\_labels(x, x', u, uc, ui, ue)$, в соответствии с условием 2 предположения 1 им присваивается уровень целостности, совпадающий с уровнем целостности учётной записи-компьютера, который по условию 1 предположения 2 равен i_high на всей траектории функционирования системы. Следовательно, с учётом предположения индукции в состоянии G_N и при переходе $G_{N-1} \vdash_{op_N} G_N$ выполнено условие 3 предположения 2.

Шаг индукции доказан. ■

Обоснуем также, что в МРОКС ДП-модели задание правил преобразования состояний системы корректо относительно требований, предъявляемых к мандатному и ролевому управлению доступом, связанных с новыми множествами запрещающих ролей UR и запрещающих ролей удалённого доступа NUR.

Утверждение 2. Пусть G_0 — начальное состояние системы $\Sigma(G^*, OP, G_0)$, соответствующей любой отдельной рабочей станции сети, удовлетворяющее условиям предположения 3 МРОКС ДП-модели. Тогда для любой траектории $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$, где $N \geqslant 1$, в состоянии G_N выполняются условия предположения 3 МРОКС ДП-модели, и переход $G_{N-1} \vdash_{op_N} G_N$ удовлетворяет условиям этого предположения.

Доказательство.

Индукция по длине N траектории функционирования системы.

Пусть $N = 0$, тогда по условию утверждения состояние G_0 удовлетворяет условиям предположения 3.

Пусть $N > 0$ и утверждение верно для всех траекторий длины $0 \leqslant L < N$. Пусть $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$ — траектория функционирования системы. По предположению индукции состояние G_{N-1} удовлетворяет условиям предположения 3, а так-

же каждый переход $G_{i-1} \vdash_{op_i} G_i$ удовлетворяет условиям этого предположения, где $1 \leqslant i < N$.

Рассмотрим правило преобразования состояний op_N. Если условия его применения не выполняются в состоянии G_{N-1}, то по определению правил преобразования состояний справедливо равенство $G_{N-1} = G_N$ и по предположению индукции состояние G_N и переход $G_{N-1} \vdash_{op_N} G_N$ удовлетворяют условиям предположения 3. Пусть условия применения правила преобразования состояний op_N выполняются в состоянии G_{N-1}.

Если op_N — де-факто правило, то в результате его применения могут измениться только значение функции $de\_facto\_own_{N-1}$ и множество информационных потоков F_{N-1}. Требований к множеству информационных потоков (как по памяти, так и по времени) и фактическому владению субъект-сессиями в предположениии 3 не содержится. Следовательно, если op_N — де-факто правило, то по предположению индукции состояние G_N и переход $G_{N-1} \vdash_{op_N} G_N$ удовлетворяют условиям предположения 3.

Пусть op_N — де-юре правило. Обоснуем выполнение условий предположения 3 в состоянии G_N и при переходе $G_{N-1} \vdash_{op_N} G_N$.

В исходном состоянии всех де-юре правил МРОКС ДП-модели, в которых проверяется наличие прав доступа (кроме права доступа own_r) у текущих ролей или административных ролей субъект-сессии, проверяется отсутствие текущих запрещающих ролей, обладающих идентичными правами доступа. Следовательно, с учётом предположения индукции в состоянии G_N и при переходе $G_{N-1} \vdash_{op_N} G_N$ выполнено условие 1 предположения 3.

Создание субъект-сессии может быть выполнено с использование одного из правил вида $create\_first\_session(x, x', u, y, z, zc, zi)$ и $create\_session(x, x', y, z)$, в результате применения которых субъект-сессия получает доступ на чтение ко всем запрещающим ролям с уровнем конфиденциальности, не превосходящим уровня доступа создаваемой субъект-сессии, и таким, что индивидуальная административная роль учётной записи пользователя, от чьего имени запускается субъект-сессия, обладает административным правом доступа на чтение к ним. Следовательно, с учётом предположения индукции в состоянии G_N и при переходе $G_{N-1} \vdash_{op_N} G_N$ выполнено условие 2 предположения 3.

В соответствии с условиями правила $create\_role(x, x', r, rc, ri, re, name, rz)$ при создании запрещающих ролей им назначается уровень конфиденциальности i_high, и в модели отсутствуют правила, позволяющие изменять уровень целостности роли впоследствии. Следовательно, с учётом предположения индукции в состоянии G_N и при переходе $G_{N-1} \vdash_{op_N} G_N$ выполнено условие 3 предположения 3.

В отличие от МРОСЛ ДП-модели, административная роль может содержать административное право доступа на чтение к запрещающей роли независимо от уровня целостности последней. Правило op_N, в результирующем состоянии которого административной роли назначается только право доступа на чтение (без права доступа на запись или владение) к запрещающей роли, может быть правилом вида $grant\_admin\_rights(x, x', ar, \{(r, \alpha_{r_j}) : 1 \leqslant j \leqslant k\})$, чьи условия и результаты применения не накладывают ограничений на уровень целостности запрещающей роли. Во всех остальных случаях административные права доступа к запрещающей роли задаются аналогично их заданию для ролей или административных ролей. Таким образом, с учётом предположения индукции в состоянии G_N и при переходе $G_{N-1} \vdash_{op_N} G_N$ выполнено условие 5 предположения 3.

Административный доступ на чтение к запрещающей роли может быть предоставлен субъект-сессии только с использованием правил вида $access\_read(x, x', y)$,

$create\_first\_session(x, x', u, y, z, zc, zi)$ и $create\_session(x, x', y, z)$, в условиях и результатах применения которых также не накладывается ограничений на уровень целостности запрещающей роли. Следовательно, с учётом предположения индукции в состоянии G_N и при переходе $G_{N-1} \vdash_{op_N} G_N$ выполнено условие 6 предположения 3.

Выполнение условия 4 предположения 3 доказывается аналогично условиям, заданным в МРОСЛ ДП-модели [1], в части, касающейся получения доступа к роли с уровнем целостности i_high.

Шаг индукции доказан. ■

Заключение

Рассмотрена разрабатываемая (на основе МРОСЛ ДП-модели) автором мандатная сущностно-ролевая ДП-модель управления доступом в компьютерных сетях, построенных на основе ОС семейства *Linux*. В её рамках заданы новые функции и множества, а также правила преобразования состояний, которые позволяют описывать спецификации механизмов управления доступом в ОС с учётом особенностей организации логического управления доступом в компьютерных сетях. Сформулированы и доказаны утверждения о корректности правил преобразования состояний системы относительно требований, предъявляемых к мандатному и ролевому управлению доступом.

Задание новых правил преобразования МРОКС ДП-модели, связанных с организацией логического управления доступом в рамках компьютерных сетей, нацелено на их реализацию в ОССН *Astra Linux Special Edition* [3] при построении сетевой подсистемы безопасности.

ЛИТЕРАТУРА

1. *Девянин П. Н.* Администрирование системы в рамках мандатной сущностно-ролевой ДП-модели управления доступом и информационными потоками в ОС семейства *Linux* // Прикладная дискретная математика. 2013. № 4(22). С. 22–40.
2. *Девянин П. Н.* Корректность правил преобразования состояний системы в рамках мандатной сущностно-ролевой ДП-модели ОС семейства *Linux* // Прикладная дискретная математика. Приложение. 2013. № 6. С. 58–59.
3. `http://www.astra-linux.ru` — Операционные системы *Astra Linux*. 2009.

REFERENCES

1. *Devyanin P. N.* Administrirovanie sistemy v ramkakh mandatnoy sushchnostno-rolevoy DP-modeli upravleniya dostupom i informatsionnymi potokami v OS semeystva *Linux* [System administration in MROSL DP-model]. Prikladnaya Diskretnaya Matematika, 2013, no. 4(22), pp. 22–40. (in Russian)
2. *Devyanin P. N.* Korrektnost' pravil preobrazovaniya sostoyaniy sistemy v ramkakh mandatnoy sushchnostno-rolevoy DP-modeli OS semeystva *Linux* [Correctness of state transformation rules in MROSL DP-model]. Prikladnaya Diskretnaya Matematika. Prilozhenie, 2013, no. 6, pp. 58–59. (in Russian)
3. `http://www.astra-linux.ru`