7. https://fstec.ru/normotvorcheskaya/informatsionnye-i-analiticheskie-materialy/2171-informatsionnoe-soobshchenie-fstek-rossii-ot-10-fevralya-2021-g-n-240-24-647. Информационное сообщение ФСТЭК России от 10.02.2021 № 240/24/647.
8. https://frama-c.com/html/acsl.html — ANSI/ISO C Specification Language. 2022.
9. https://www.ispras.ru/technologies/astraver_toolset/ — Система верификации AstraVer Toolset. 2022.
10. *Kirchner F., Kosmatov N., Prevosto V., et al.* Frama-C: a software analysis perspective // Formal Aspects of Computing. 2015. No. 27(3). P. 573–609.
11. *Filliatre J.-C. and Paskevich A.* Why3 — where programs meet provers // LNCS. 2013. V. 7792. P. 125–128.
12. *Marche C. and Moy Y.* The Jessie Plugin for Deductive Verification in Frama-C. INRIA Saclay Ile-de-France and LRI, CNRS UMR, 2012.
13. *Колисниченко Д. Н.* Разработка Linux-приложений. СПб.: БХВ-Петербург, 2012. 432 с.

УДК 004.056.5, 004.94 DOI 10.17223/2226308X/15/22

СРАВНЕНИЕ СПОСОБОВ МОДЕЛИРОВАНИЯ МЕХАНИЗМОВ УПРАВЛЕНИЯ ДОСТУПОМ ОС И СУБД НА ФОРМАЛИЗОВАННОМ ЯЗЫКЕ МЕТОДА Event-B С ЦЕЛЬЮ ИХ ВЕРИФИКАЦИИ ИНСТРУМЕНТАМИ Rodin И ProB

М. А. Леонова, П. Н. Девянин

В результате перевода описания формальной модели управления доступом промышленной ОССН Astra Linux Special Edition (МРОСЛ ДП-модели) из математической в формализованную нотацию на языке метода Event-B, её автоматизированной верификации инструментами Rodin и ProB и проведённых в ГК Astra Linux научных исследований разработаны два способа моделирования взаимодействующих между собой систем, самостоятельно реализующих развитые механизмы управления доступом, такие, как ОС и СУБД. Эти способы основаны на использовании различных вариантов построения иерархии спецификаций МРОСЛ ДП-модели в формализованной нотации с применением техники пошагового уточнения. Сравнение способов показало как их достоинства, так и недостатки в части сложности написания спецификаций, необходимости повторения доказательств при верификации инструментом Rodin, возможности устранения эффекта «комбинаторного взрыва» при верификации инструментом ProB. По результатам сравнения сделан вывод, что рассмотренные способы могут быть полезны при разработке других формальных моделей управления доступом и их верификации с применением соответствующих средств.

Ключевые слова: *формальная модель управления доступом, МРОСЛ ДП-модель, Event-B, верификация, дедуктивная верификация, Rodin, метод проверки модели, ProB.*

Введение

В средствах защиты информации (СЗИ), таких, как ОС и СУБД, механизм управления доступом выполняет одну из основных функций по обеспечению их безопасности. Для достижения доверия к корректности этого механизма, создания условий для научного обоснования выполнения им заданных для СЗИ требований безопасности уже многие десятилетия разрабатываются формальные модели управления доступом [1, 2].

Мандатная сущностно-ролевая ДП-модель управления доступом и информационными потоками в ОС семейства Linux (МРОСЛ ДП-модель) [2] изначально создана для реализации в защищённой операционной системе специального назначения Astra Linux Special Edition (ОССН) [3, 4]. Данная модель, изложенная на математическом языке (в математической нотации), имеет иерархическое представление и состоит из восьми уровней (рис. 1) — четырёх уровней, моделирующих механизм управления доступом самой ОССН, и четырёх аналогичных уровней для штатной СУБД PostgreSQL. Уровни ОССН уточняются (наследуются и дополняются) строго последовательно (линейно) от первого к четвёртому, но иерархия уровней СУБД имеет более сложный вид: каждый последующий уровень СУБД уточняет предыдущий, а также соответствующий ему уровень ОССН, что делается для моделирования взаимодействия механизмов управления доступом данных систем.

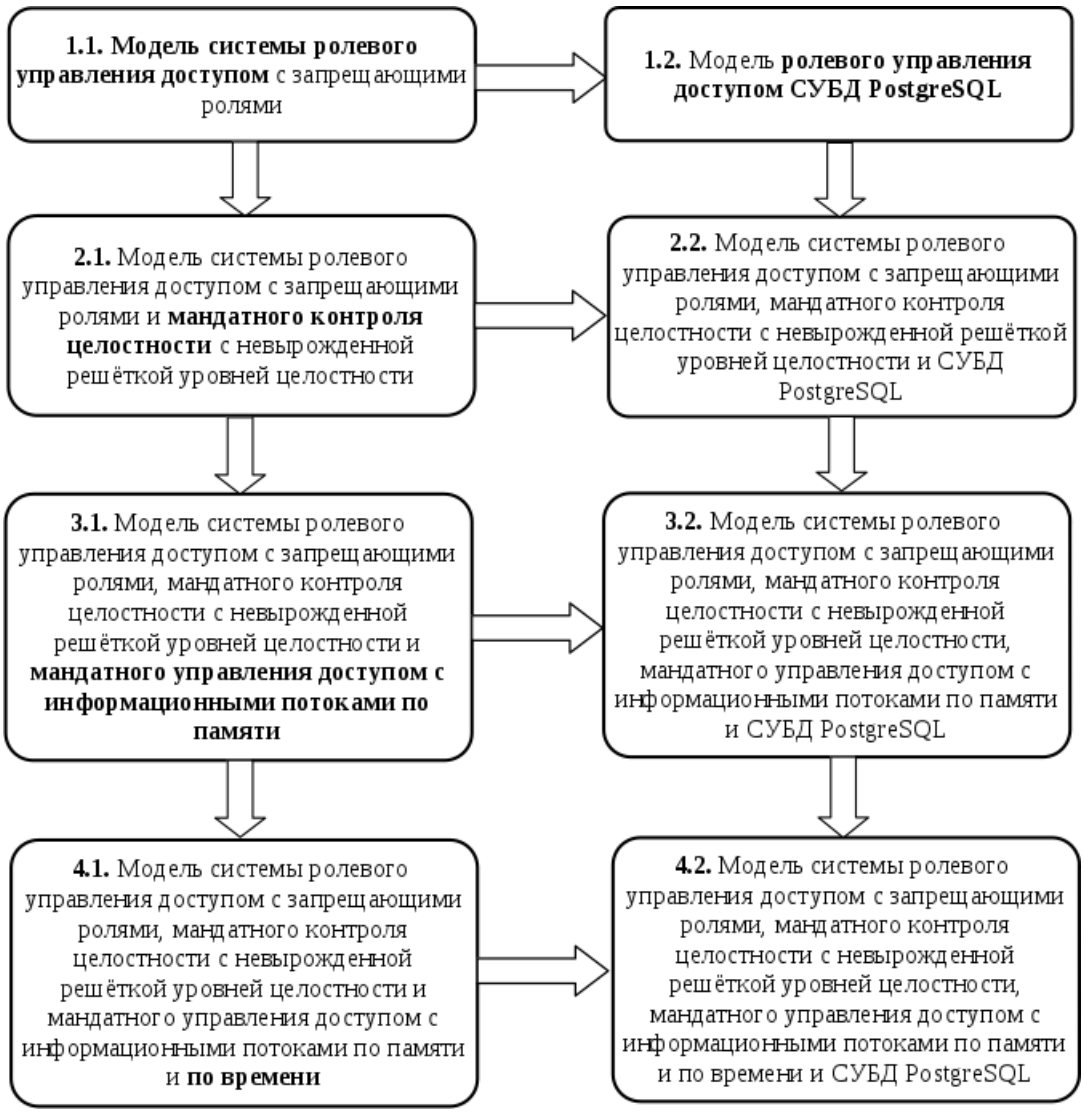

Рис. 1. Схема иерархического представления МРОСЛ ДП-модели

Хотя изначально модель формировалась в математической нотации, по мере увеличения объёма её описания (сейчас около 600 страниц) для повышения эффективности разработки, проверки корректности и отсутствия ошибок в самой модели для её описания стал применяться формализованный (машиночитаемый) язык метода Event-B [5],

т. е. представление модели в формализованной нотации [6, 7]. Для автоматизированной проверки корректности и верификации описания модели [8] в этой нотации используется инструмент дедуктивной верификации Rodin [9]. Кроме того, для повышения качества верификации МРОСЛ ДП-модели, расширения спектра применяемых для этого методов и инструментов, моделирования и в перспективе автоматизированного тестирования на соответствие этой модели её реализации непосредственно в программном коде и настройках конфигурации механизмов управления доступом ОССН и СУБД [10] осуществляется верификация модели с использованием инструмента проверки моделей ProB [11, 12].

Описание МРОСЛ ДП-модели в формализованной нотации на языке метода Event-B представляет собой последовательность связанных между собой спецификаций двух видов: контекстов (context) и машин (machine). Контексты определяют неизменяемую, базовую часть модели, в них задаются несущие множества (sets), константы (constants) и аксиомы (axioms). Машины являются динамической частью, в которых описываются переменные (variables), инварианты на них (invariants) и события (events), в свою очередь состоящие из параметров (parameters), охранных условий (guards) и действий (actions), изменяющих значения переменных машин. Для каждого уровня модели создаются машина и, при необходимости, контекст, после написания которых Rodin автоматически генерирует утверждения для доказательства (proof obligations). Модель считается дедуктивно верифицированной с применением инструмента Rodin, если для всех её уровней выполнены доказательства сгенерированных утверждений.

Последовательности спецификаций в инструменте Rodin задаются следующим образом: контексты могут расширяться (extends) несколькими контекстами, а для машин используется техника пошагового уточнения (refinement) [13], которая позволяет машине уточнять не более одной машины, другими словами, возможно составить только линейную последовательность машин. Ещё одной особенностью является невозможность изменения значений определяемых инвариантами функций на любых уровнях спецификаций, кроме того, на котором они задаются. Хотя, следует отметить, это может быть необходимо для корректного моделирования взаимодействия систем (например, возникновения информационных потоков между элементами ОС и СУБД).

При переводе описания модели из математической в формализованную нотацию накладываемые refinement ограничения не препятствовали моделированию иерархии уровней для ОССН, так как их последовательность линейна. Но при моделировании нелинейной иерархии динамической части уровней СУБД потребовался поиск способов обхода данных ограничений ввиду того, что машина каждого уровня (кроме первого) должна уточнять две другие машины. При этом необходимо на данных уровнях изменять значения функций, задаваемых на уровнях ОССН.

Предлагаются два способа моделирования нелинейной иерархии МРОСЛ ДП-модели. Первый способ заключается в непосредственном использовании техники refinement и построении линейной последовательности всех спецификаций восьми уровней, когда первый уровень СУБД уточняет четвёртый уровень ОССН. Второй способ также состоит в использовании техники refinement, но с построением двух ветвей уточнений, корнем которых является машина, описывающая первый уровень ОССН. Каждый из способов обладает достоинствами и недостатками как самого процесса моделирования, так и при осуществлении дедуктивной верификации инструментом Rodin и верификации по методу проверки моделей инструментом ProB получаемого представления

модели в формализованной нотации. Анализу и сравнению этих двух способов моделирования посвящена настоящая работа.

1. Первый способ

Первый способ описания нелинейной иерархии МРОСЛ ДП-модели в формализованной нотации заключается в прямом использовании техники refinement и построении линейной последовательности всех спецификаций восьми уровней, согласно которой первый уровень СУБД уточняет четвёртый уровень ОССН (рис. 2), но с переопределением на них необходимых для логического согласования модели в математической и формализованной нотациях функций и событий.

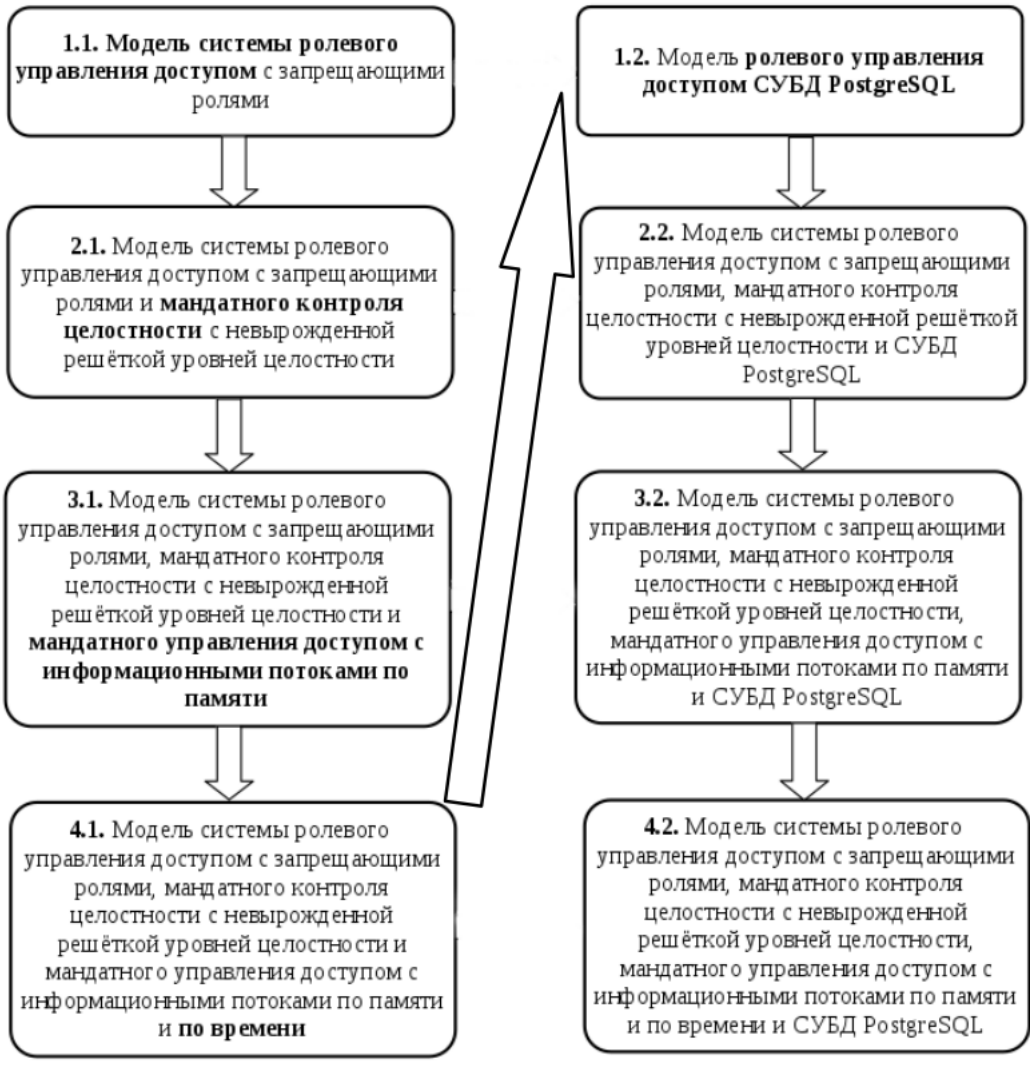

Рис. 2. Последовательность уточнения спецификаций уровней модели при использовании первого способа

Например, при описании машины второго уровня СУБД (шестого в последовательности) необходимо иметь возможность изменять значение функции информационных потоков по памяти от субъектов к сущностям ОССН SEMFlows при выполнении дефакто события создания информационного потока по памяти от субъекта к сущности ОССН при наличии субъекта-посредника find_entity_m. Однако данная функция определяется в машине аналогичного уровня ОССН. В результате использования

первого способа в машине второго уровня СУБД задаются функция dbSEFlows (переопределяющая функцию SEMFlows) и изменяющее её значение де-факто событие os_find_entity (переопределяющее событие find_entity_m). При этом для анализа информационных потоков по памяти от субъектов к сущностям СУБД определяются новые функция S_DBEFlows и событие db_find_entity (листинг 1).

```
os_find_entity
ANY
x, y, z, flow

WHERE
grd1: x ∈ Subjects
grd2: y ∈ Subjects
grd3: z ∈ Entities
grd4: flow ∈ ℙ1({1,2})
//‘‘1’’ - информационный поток по памяти, ‘‘2’’ - информационный поток по времени
grd5: 1 ∈ flow ⇒ z ∉ EHole
grd6: 1 ∈ flow ⇒ y ↦ 1 ∈ dbSSFlows(x) ∧ z ↦ 1 ∈ dbSEFlows(y)

THEN
act1: dbSEFlows(x) := dbSEFlows(x) ∪ ({z} × flow)
END

db_find_entity
ANY
x, y, z, flow

WHERE
grd1: x ∈ Subjects
grd2: y ∈ Subjects
grd3: z ∈ Entities
grd4: flow ∈ ℙ1({1,2})
grd5: 1 ∈ flow ⇒ z ∉ DBEHole
grd6: 1 ∈ flow ⇒ y ↦ 1 ∈ dbSSFlows(x) ∧ z ↦ 1 ∈ S_DBEFlows(y)

THEN
act1: S_DBEFlows(x) := S_DBEFlows(x) ∪ ({z} × flow)
END
```

Листинг 1. Общие для ОССН и СУБД события при моделировании первым способом

Описание машины на каждом уровне СУБД состоит из двух частей. Первая (и основная) часть описывает его отличия от предыдущего уровня, касающиеся моделируемого управления доступом (при этом необходимый уровень ОССН уже наследуется данным уровнем СУБД ввиду линейной последовательности всех спецификаций). Так, второй уровень СУБД (шестой в последовательности) задаёт мандатный контроль целостности, а значит, на третьем уровне СУБД (седьмом в последовательности) добавится описание мандатного управления доступом с информационными потоками по памяти с учётом наследования третьего уровня ОССН. Вторая часть ввиду особенностей техники refinement содержит переопределения некоторых функций и событий аналогичного уровня ОССН. Например, для третьего уровня СУБД (седьмого в последовательности) это затронет несколько функций и событий третьего уровня ОССН. Однако в целом по сравнению с общим объёмом кода описания каждого уровня СУБД объём такого переопределяемого («избыточного») кода невелик — приблизительно десятая часть. Соответственно столь же невелико число «избыточных» (повторяемых

для переопределенных функций и событий на уровне СУБД, уже выполненных на уровне ОССН) доказательств относительно их общего числа для каждого уровня, что является основным достоинством первого способа. Для примера на втором уровне СУБД переопределены 5 функций и 13 событий из общего числа описанных на нём, соответственно 20 функций и 63 событий, а также выполнены 156 «избыточных» доказательств из 1099, что не так много по сравнению с описанным далее вторым способом.

Однако у включения на каждом уровне СУБД всех уровней ОССН есть существенные недостатки. Во-первых, несоответствие сути постепенного для обеих систем добавления механизмов защиты, таких, как мандатный контроль целостности и мандатное управление доступом. Во-вторых, трудности непосредственного применения инструментов Rodin и ProB, одной из причин этого является большой объём описания уровней модели для СУБД в её формализованной нотации в результате наследования всего описания модели для ОССН. Если для Rodin это не так критично, то для ProB зачастую делает невозможным даже начальную инициализацию модели при верификации.

При использовании инструмента Rodin с каждым последующим уровнем увеличивается сложность верификации с точки зрения автоматизации ввиду того, что при доказательстве утверждения в качестве гипотез используются все аксиомы, инварианты и охранные условия спецификаций всех предыдущих уровней. Это затрудняет встроенным пруверам (provers) и солверам (solvers) поиск необходимых гипотез для вывода гипотезы-цели, являющейся смыслом каждого доказательства. Данную задачу приходится решать ручным удалением избыточных гипотез.

При верификации Rodin определённого уровня требуется выполнить также доказательства утверждений, сгенерированных только для этого уровня, считая, что уточнённые им уровни верифицированы (на них выполнены доказательства всех утверждений), даже если это на данный момент не так (например, определённый набор утверждений остался недоказанным). Другими словами, уровни верифицируются отдельно друг от друга, что при выявлении впоследствии ошибок в спецификациях предыдущего уровня может потребовать переработку спецификаций и повторное доказательство утверждений последующего уровня.

В свою очередь, применяемый для верификации по методу проверки моделей инструмент ProB позволяет для заданной модели с конечным (для МРОСЛ ДП-модели очень большим) числом состояний проверить, выполняются ли в рассматриваемых состояниях модели условия их корректности (инварианты). Однако основная трудность, которую приходится здесь преодолевать, связана с эффектом «комбинаторного взрыва» в пространстве состояний. Это характерно для моделирования систем (в том числе ОС и СУБД), состоящих из многих компонент, взаимодействующих друг с другом, и описываемых структурами данных, способными принимать большое число значений.

При работе ProB спецификации верифицируемого уровня включают спецификации всех предыдущих уровней. Соответственно при инициализации модели и переходе её в каждое следующее состояние необходимо проверить выполнение инвариантов для значений переменных в спецификациях текущего и всех предыдущих уровней. В результате для верификации, например, второго уровня СУБД (шестого в последовательности) необходимо также по сути избыточно верифицировать третий и четвёртый уровни ОССН, что вследствие проявления «комбинаторного взрыва» завершало работу ProB с ошибкой вида timeout, вызванной превышением допустимого интервала времени, установленного для выполнения переборных алгоритмов.

Таким образом, первый способ, основанный на построении линейной последовательности спецификаций уровней ОССН и СУБД МРОСЛ ДП-модели в формализованной нотации, обладает рядом отмеченных недостатков, которые сделали практически невозможной верификацию с использованием инструмента ProB уровней СУБД. Для устранения этих недостатков разработан альтернативный способ представления модели, предполагающий построение двух ветвей уточнений.

2. Второй способ

Второй способ описания нелинейной иерархии МРОСЛ ДП-модели в формализованной нотации заключается также в использовании техники refinement, но при этом строятся две ветви (последовательности) уточнений, корнем которых является машина, описывающая первый уровень ОССН (рис. 3):

— первая ветвь — линейно уточняющие друг друга спецификации четырёх уровней ОССН;
— вторая ветвь — линейно уточняющие друг друга машины первого уровня ОССН и четырёх уровней СУБД, но с ручным полным дублированием кода машин уровней ОССН в соответствующие им машины уровней СУБД и повторным доказательством сгенерированных утверждений для ОССН, начиная со второго уровня.

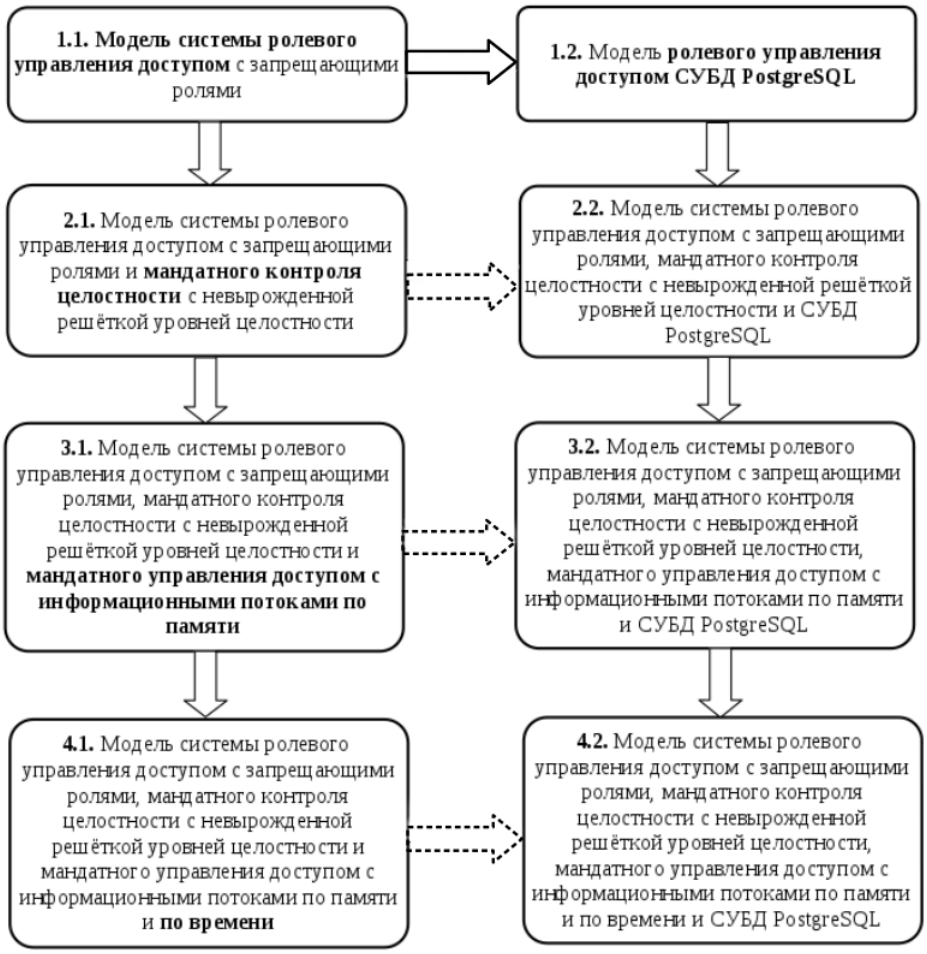

Рис. 3. Ветви (последовательности) уточнения спецификаций уровней модели при использовании второго способа

Построение второй ветви по сути является «ручным» уточнением машин первой ветви для ОССН соответствующими машинами второй ветви для СУБД. В отличие от машин, вследствие поддержки Rodin расширения контекстов несколькими контекстами иерархия спецификаций контекстов уровней полностью идентична исходной нелинейной иерархии МРОСЛ ДП-модели.

Для примера в листинге 2 приведены два события из машины второго уровня СУБД 2-M-DBMS-MIC: access_read_entity (получение субъектом доступа на чтение к сущности ОССН) и access_read_db_entity (получение субъектом доступа на чтение к сущности СУБД). Событие access_read_entity уточняет охранными условиями grd5–grd7 одноимённое событие машины первого уровня ОССН, при этом данные предикаты дублируются из машины второго уровня ОССН, а событие access_read_db_entity уточняет охранными условиями grd6, grd7 одноимённое событие машины первого уровня СУБД, при этом данные предикаты являются новыми в рамках описания модели в формализованной нотации.

```
    access_read_entity
    ANY
    subject, entity

    WHERE
    grd1: subject ∈ Subjects
    grd2: entity ∈ Entities
    grd3: entity ↦ Read ∈ CheckRightE(subject)
    grd4: entity ∈ ExecuteContainer(subject)
    grd5: entity ↦ Read ∈ CheckRightEInt(subject)
    grd6: entity ∈ ExecuteContainerInt(subject)
    grd7: WithoutCoop = TRUE ⇒ SubjectInt(subject) ≠ HighI
    THEN
    act1: SubjectAccesses(subject) := SubjectAccesses(subject) ∪ {entity ↦ ReadA}
    END

    access_read_db_entity
    ANY
    subject, element, privilege

    WHERE
    grd1: subject ∈ Subjects
    grd2: elements ∈ DBElements
    grd3: privilege ∈ DBPrivileges
    grd4: Read ∈ dbRights(privilege)
    grd5: PostgresAdmRole ↦ ReadA ∈ DBSubjectAdmAccesses(subject) ∨
(dbEntity(element) ↦ Read ∈ dbCheckRightE(subject) ∧
dbEntity(element) ∈ dbExecuteContainer(subject))
    grd6: PostgresAdmRole ↦ ReadA ∈ DBSubjectAdmAccesses(subject) ∨
(dbEntity(element) ↦ Read ∈ dbCheckRightEInt(subject) ∧
dbEntity(element) ∈ dbExecuteContainerInt(subject))
    grd7: WithoutCoop = TRUE
    THEN
    act1: DBSubjectAccesses(subject) := DBSubjectAccesses(subject) ∪
{dbEntity(element) ↦ ReadA}
    END
```

Листинг 2. Примеры событий второго уровня СУБД, дублирующих события ОССН или являющихся новыми, при моделировании вторым способом

При использовании данного способа моделирования в формализованной нотации по аналогии с математической реализуется идея постепенного для обеих систем включения механизмов защиты, таких, как мандатный контроль целостности и мандатное управление доступом. Также упрощаются автоматизированные дедуктивная верификация модели с применением инструмента Rodin (устраняется избыточность используемых в доказательстве гипотез) и верификация по методу проверки моделей с применением инструмента ProB (устраняется избыточность необходимых для верификации машин), так как каждая машина уровня СУБД уточняет по технике refinement машину предыдущего уровня СУБД и «вручную» соответствующую ей машину уровня ОССН.

Недостатком второго способа моделирования относительно первого является его трудоёмкость. Вместо описания восьми уровней МРОСЛ ДП-модели на формализованном языке Event-B и их автоматизированной дедуктивной верификации необходимо по сути описать и верифицировать одиннадцать уровней — четыре уровня ОССН и четыре уровня СУБД, внутри себя содержащих полное дублирование трёх уровней ОССН (второго, третьего и четвёртого). Сравнительная статистика объёма избыточного кода при описании модели и доли её избыточной верификации в формализованной нотации каждым из двух способов представлены в таблице. Под «ручным» кодом здесь понимается код, который необходимо написать на каждом уровне модели без учёта кода, наследуемого в результате использования техники refinement с предыдущих уровней модели. Кроме того, в таблице отдельно приведена статистика для де-юре событий (используемых для моделирования функций механизмов защиты ОССН и СУБД) и де-факто событий (используемых для анализа информационных потоков или условий получения управления одним субъектом над другим).

Восемь уровней модели для управления доступом в ОССН и СУБД	Количество де-юре событий, шт.	Количество де-факто событий, шт.	Объём «ручного» кода, тыс. строк	Объём избыточного кода, тыс. строк	Доля избыточной верификации, %
Первый способ	76	80	12,5	1,2	0,1
Второй способ	74	64	15,4	4,2	26,2

Сравнивая объёмы «ручного» и избыточного кода, полученные каждым из способов, можно сделать вывод, что на описание МРОСЛ ДП-модели в формализованной нотации с применением второго способа уходит больше времени, при этом доля избыточного кода по отношению к общему объёму кода, написанного «вручную», составляет около 27 % против 10 для первого способа. Длительность верификации модели с использованием второго способа также возрастает, так как доля избыточной верификации становится равной 26 % против 0,1 для первого способа. Однако использование второго способа значительно упрощает верификацию с применением инструмента ProB, которая при использовании первого способа для уровней СУБД практически невозможна.

Заключение

Проанализированы два способа перевода описания модели управления доступом промышленной ОССН Astra Linux Special Edition (МРОСЛ ДП-модели) из математической в формализованную нотацию на языке метода Event-B и её автоматизированной верификации инструментами Rodin и ProB. Оба способа основаны на применении техники пошагового уточнения refinement для представления иерархии уровней

МРОСЛ ДП-модели, соответствующих взаимодействующим между собой системам, самостоятельно реализующим развитые механизмы управления доступом, такие, как сама ОССН и штатная для неё СУБД PostgreSQL. Несмотря на отмеченные недостатки, каждый из предложенных способов имеет существенные достоинства и вместе они могут быть полезны при разработке других формальных моделей управления доступом и их верификации с применением соответствующих инструментов.

ЛИТЕРАТУРА

1. ГОСТ Р 59453.1-2021 «Защита информации. Формальная модель управления доступом. Ч. 1. Общие положения». М.: Стандартинформ, 2021. 16 с.
2. *Девянин П. Н.* Модели безопасности компьютерных систем. Управление доступом и информационными потоками: учеб. пособие для вузов. 3-е изд., перераб. и доп. М.: Горячая линия — Телеком, 2020. 352 с.
3. `https://astralinux.ru/products/astra-linux-special-edition/` — Операционная система специального назначения Astra Linux Special Edition. 2022.
4. *Буренин П. В., Девянин П. Н., Лебеденко Е. В. и др.* Безопасность операционной системы специального назначения Astra Linux Special Edition: учеб. пособие для вузов. 3-е изд., перераб. и доп. М.: Горячая линия — Телеком, 2019. 404 с.
5. *Abrial J.-R.* Modeling in Event-B: System and Software Engineering. Cambridge University Press, 2010.
6. *Девянин П. Н., Ефремов Д. В., Кулямин В. В. и др.* Моделирование и верификация политик безопасности управления доступом в операционных системах. М.: Горячая линия — Телеком, 2019. 214 с.
7. *Девянин П. Н., Леонова М. А.* Применение подтипов и тотальных функций формального метода Event-B для описания и верификации МРОСЛ ДП-модели // Программная инженерия. 2020. Т. 11. № 4. С. 230–241.
8. ГОСТ Р 59453.2-2021 «Защита информации. Формальная модель управления доступом. Ч. 2. Рекомендации по верификация формальной модели управления доступом». М.: Стандартинформ, 2021. 12 с.
9. *Abrial J.-R., Butler M., Hallerstede S., et al.* Rodin: An open toolset for modelling and reasoning in Event-B // Intern. J. Software Tools for Technology Transfer. 2010. V. 12. No. 6. P. 447–466.
10. *Девянин П. Н., Тележников В. Ю., Хорошилов А. В.* Формирование методологии разработки безопасного системного программного обеспечения на примере операционных систем // Труды ИСП РАН. 2021. Т. 33. № 5. С. 25–40.
11. *Leuschel M. and Butler M.* ProB: an automated analysis toolset for the B method // Int. J. Softw. Tools Technol. Transf. 2008. No. 10(2). P. 185–203.
12. *Девянин П. Н., Леонова М. А.* Приёмы по доработке описания модели управления доступом ОССН Astra Linux Special Edition на формализованном языке метода Event-B для обеспечения её автоматизированной верификации с применением инструментов Rodin и ProB // Прикладная дискретная математика. 2021. № 52. С. 83–96.
13. *Abrial J.-R. and Hallerstede S.* Refinement, decomposition, and instantiation of discrete models: Application to Event-B // Fundamenta Informaticae. 2007. V. 77. Iss. 1–2. P. 1–28.