Секция 4

МАТЕМАТИЧЕСКИЕ ОСНОВЫ КОМПЬЮТЕРНОЙ БЕЗОПАСНОСТИ

УДК 004.94 DOI 10.17223/2226308X/12/45

О МОДЕЛИРОВАНИИ В РАМКАХ МРОСЛ ДП-МОДЕЛИ МАНДАТНЫХ КОНТРОЛЯ ЦЕЛОСТНОСТИ И УПРАВЛЕНИЯ ДОСТУПОМ В СУБД PostgreSQL

П. Н. Девянин

Интеграция в операционную систему специального назначения (ОССН) Astra Linux Special Edition прикладного программного обеспечения, включающего собственные механизмы управления доступом, требует, во-первых, соответствующей инженерной реализации по их сопряжению с базовыми для ОССН мандатными управлением доступом и контролем целостности, а во-вторых, обеспечения при этом доверия к безопасности такого сочетания механизмов управления доступом, в том числе для предотвращения информационных потоков (скрытых каналов) по памяти или по времени. Важным примером такого штатного для ОССН программного обеспечения является СУБД *PostgreSQL*, изначально реализующая развитый механизм ролевого управления доступом. Сертификация ОССН по требованиям утверждённого ФСТЭК России профиля защиты операционных систем общего назначения первого (высшего) класса защиты, в ходе которой разрабатывалась и верифицировалась формальная модель управления доступом, а также сертификация ОССН по требованиям других отечественных регуляторов, говорят о целесообразности подготовки к выполнению аналогичных требований применительно к СУБД. В этой связи рассматриваются результаты завершения формирования уровней мандатных управления доступом и контроля целостности СУБД *PostgreSQL* на базе иерархического представления мандатной сущностно-ролевой ДП-модели (МРОСЛ ДП-модели), являющейся научной основой разработки механизма управления доступом ОССН.

Ключевые слова: *компьютерная безопасность, формальная модель, управление доступом, PostgreSQL.*

Система требований к средствам защиты информации, формируемая отечественными регуляторами, непрерывно развивается, вбирая в себя достижения науки и передовой мировой опыт. Так, в применяемых с 1 июня 2019 г. «Требованиях по безопасности информации, устанавливающих уровни доверия к средствам технической защиты информации и средствам обеспечения безопасности информационных технологий», утверждённых ФСТЭК России [1], указывается на необходимость разработки модели безопасности (с четвёртого уровня доверия) и её верификации с применением инструментальных средств (с третьего уровня доверия), а также идентификации и анализа скрытых каналов (информационных потоков) по памяти (с пятого уровня доверия) и по времени (с третьего уровня доверия).

Поскольку отечественная защищённая ОССН Astra Linux Special Edition [2, 3] сертифицирована по требованиям утвержденного ФСТЭК России профиля защиты опе-

рационных систем общего назначения самого высокого — первого класса защиты (соответствующего первому уровню доверия) [4], то при реализации её механизма управления доступом использована верифицированная мандатная сущностно-ролевая ДП-модель (МРОСЛ ДП-модель) в её иерархическом представлении [5, 6]. Вместе с тем логично предположить, что в дальнейшем требования соответствующих уровней доверия будут распространены на прикладное программное обеспечение, включающее собственные механизмы управления доступом, например на штатную для ОССН СУБД *PostgreSQL*. С целью удовлетворения этим требованиям заблаговременно [7] проведены исследования по развитию МРОСЛ ДП-модели для «охвата» ею механизма управления доступом СУБД *PostgreSQL*. В настоящее время они завершены, их итогом стало формирование в дополнение к четырём уровням в иерархическом представлении модели для ОССН: ролевого управления доступом (1.1), мандатного контроля целостности (2.1), мандатного управления доступом с информационными потоками по памяти (3.1) и по времени (4.1) и аналогичных взаимосвязанных с ними уровней (1.2–4.2) для СУБД (рис. 1).

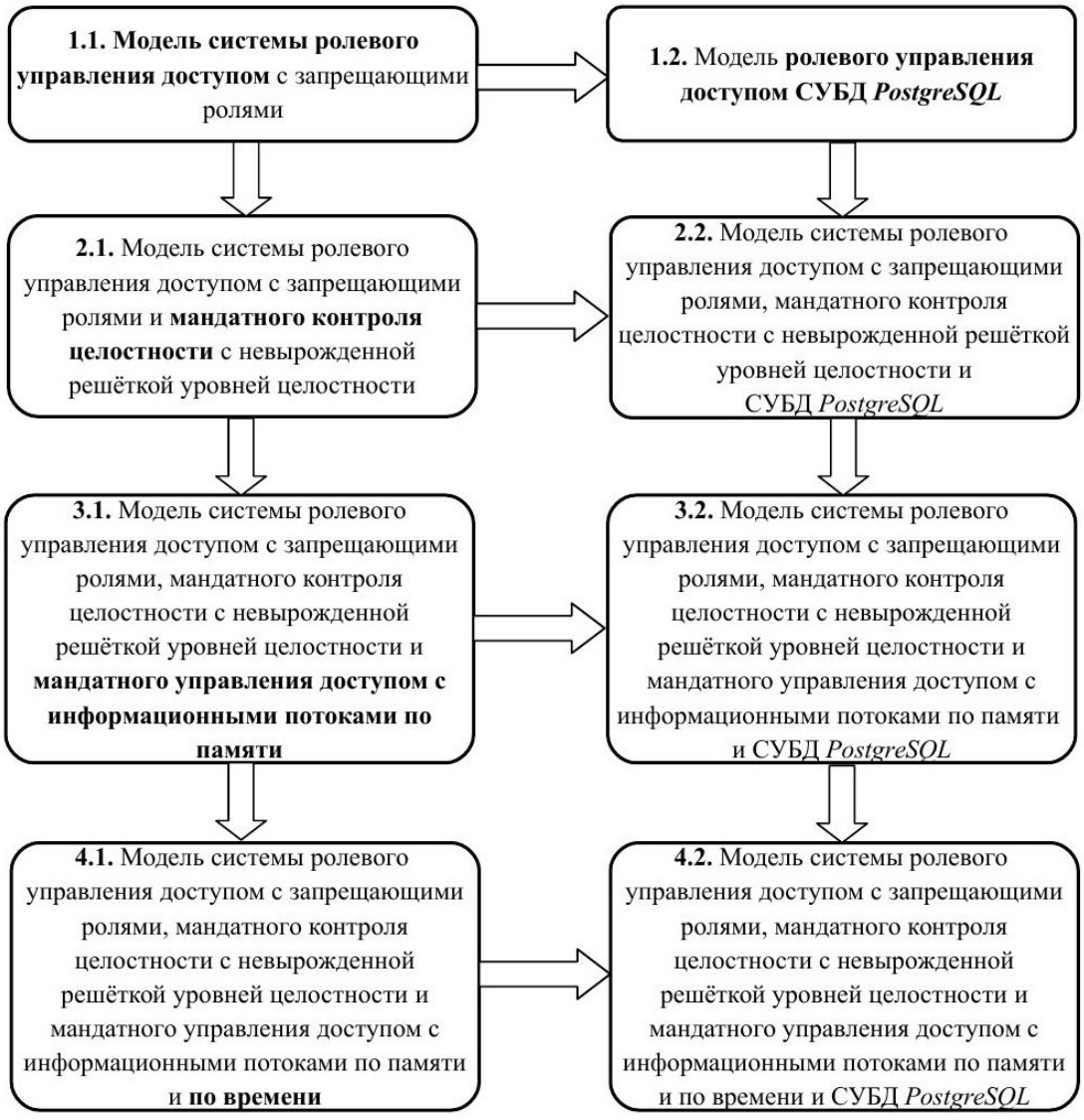

Рис. 1. Иерархическое представление МРОСЛ ДП-модели для ОССН и СУБД *PostgreSQL*

С учётом существенных отличий основанного на стандарте SQL механизма управления доступом СУБД *PostgreSQL* от реализованного в ОССН, начиная с уровня ролевого управления доступом (1.2), были использованы:

— множества: ролей СУБД, административных привилегий СУБД (*SUPERUSER*, *CREATEROLE*, *CREATEDB*, *LOGIN*, *REPLICATION*, *INHERIT*), административных ролей СУБД, специальных ролей СУБД, общих ролей СУБД, элементов СУБД, элементов-объектов СУБД (каталоги по событию, расширения, сопоставления, домены, конфигурации, словари, парсеры, шаблоны, функции, последовательности, строки, ограничения, индексы, правила, триггеры, триггерные функции, репликации), элементов-контейнеров СУБД (кластеры, базы данных, схемы, таблицы, столбцы, представления, табличные пространства), видов привилегий СУБД (*SELECT*, *INSERT*, *UPDATE*, *DELETE*, *TRUNCATE*, *REFERENCES*, *TRIGGER*, *USAGE*, *CREATE*, *CONNECT*, *TEMPORARY*, *TEMP*, *EXECUTE*, *OWN*), сущностей СУБД (на этом уровне модели сущностями являются элементы СУБД от её схем и далее выше по иерархии, например, базы данных, кластеры), привилегий к элементам СУБД;

— функции: административных привилегий ролей СУБД, ролей входа субъект-сессий в СУБД, наследования привилегий ролей к элементам СУБД, управления подчинённостью ролей в иерархии, административных прав доступа административных ролей ОССН и СУБД к ролям СУБД, привилегий к элементам СУБД ролей СУБД, соответствия административных привилегий и видов привилегий к элементам СУБД правам доступа, эффективных прав доступа ролей СУБД.

Кроме того, на этом уровне переопределены: множество доступов субъект-сессий к ролям, запрещающим ролям, административным ролям и ролям СУБД; функции: имён сущностей и элементов СУБД, имён ролей, запрещающих ролей, административных ролей, ролей СУБД, доступа субъект-сессий к сущностям и элементам СУБД в контейнерах; иерархия сущностей и элементов СУБД; иерархия ролей, запрещающих ролей, административных ролей и ролей СУБД; состояние системы.

На последующих уровнях модели (2.2–4.2) добавлены множества: доверенных субъект-сессий СУБД, функций СУБД, доверенных субъект-сессий СУБД, корректных относительно информационных потоков по времени, и функция, определяющая для каждой функции СУБД режим её выполнения: `false` — выполнение функции с правами роли СУБД, являющейся владельцем функции СУБД (в СУБД *PostgreSQL* этому соответствует атрибут *SECURITY DEFINER*), `true` — выполнение функции с правами текущих ролей СУБД субъект-сессии, инициировавшей вызов функции (в СУБД *PostgreSQL* этому соответствует атрибут *SECURITY INVOKER*). Были также переопределены:

— множества: информационных потоков, сущностей, параметрически ассоциированных с ролями, запрещающими ролями, административными ролями и ролями СУБД;

— функции: уровней целостности и конфиденциальности ролей, мандатных атрибутов конфиденциальности и целостности контейнеров;

— состояние системы;

— траектория без кооперации доверенных и недоверенных субъект-сессий;

— безопасное состояние системы;

На уровне 1.2 ролевого управления доступом СУБД *PostgreSQL* по сравнению с уровнем 1.1 не задано новых де-юре правил преобразования состояний, хотя боль-

шая их часть была модифицирована путём добавления в правила новых параметров, условий и результатов применения, учитывающих специфику управления доступом в СУБД. Начиная с уровня мандатного контроля целостности СУБД, кроме внесения аналогичных изменений добавлено новое де-юре правило $db\_execute(x, x', y)$, позволяющее задать порядок активизации функций СУБД. Состав де-факто правил при моделировании управления доступом в СУБД не изменился. В итоге определены 39 де-юре и 10 де-факто правил преобразования состояний, общих для ОССН и СУБД, что создаёт условия для формирования единых подходов к их реализации, в том числе для противодействия запрещённым информационным потокам по памяти и по времени.

При задании требований к мандатным управлению доступом и контролю целостности в СУБД *PostgreSQL*, чтобы не снижать её производительности, предложено назначать уровни целостности и конфиденциальности элементам СУБД до схем включительно, т. е. не назначать их таблицам, функциям, триггерам.

Для каждого из уровней модели для СУБД сформулированы и обоснованы утверждения о соответствии (корректности) правил преобразования состояний системы условиям, которым должно удовлетворять соответственно ролевое управление доступом, мандатный контроль целостности, мандатное управление доступом с информационными потоками по памяти и по времени. С учётом внесённых на уровнях СУБД изменений переформулированы и доказаны утверждения о достаточных условиях безопасности системы в смыслах мандатного контроля целостности, контроля информационных потоков по памяти (в смысле Белла — ЛаПадулы) и контроля информационных потоков по времени. Именно для третьего случая на уровне 4.2 потребовались наибольшие изменения по сравнению с уровнем 4.1. Они заключались, во-первых, в запрете передачи роли СУБД права на дальнейшую передачу привилегии (с использованием в СУБД *PostgreSQL* права *WITH GRANT OPTION*), во-вторых, в отсутствии функций СУБД, содержащихся в сущностях-схемах СУБД, имеющих высокий уровень целостности и уровень конфиденциальности выше минимального.

Таким образом, в рамках иерархического представления МРОСЛ ДП-модели удалось полностью смоделировать управление доступом в СУБД *PostgreSQL* аналогично тому, как это было выполнено для ОССН Astra Linux Special Edition. В дальнейшем предполагается дедуктивная верификация соответствующих СУБД уровней модели с применением инструментальных средств, что соответствует утверждённым ФСТЭК России требованиям высоких уровней доверия к средствам защиты информации.

ЛИТЕРАТУРА

1. ФСТЭК России. Информационное сообщение от 29 марта 2019 г. № 240/24/1525. `https://fstec.ru/component/attachments/download/2286`.
2. Astra Linux — универсальная операционная система. `http://www.astralinux.ru`.
3. Astra Linux. `https://ru.wikipedia.org/wiki/Astra_Linux`.
4. Родина в кибербезопасности: российской ОС откроют все секреты. `https://iz.ru/871218/olga-kolentcova/rodina-v-kiberbezopasnosti-rossiiskoi-os-otkroiut-vse-sekrety/`.
5. *Буренин П. В., Девянин П. Н., Лебеденко Е. В. и др.* Безопасность операционной системы специального назначения Astra Linux Special Edition: учеб. пособие для вузов. / под ред. П. Н. Девянина. 3-е изд., перераб. и доп. М.: Горячая линия-Телеком, 2019. 404 с.
6. *Девянин П. Н., Кулямин В. В., Петренко А. К. и др.* Моделирование и верификация политик безопасности управления доступом в операционных системах. М.: Горячая линия-Телеком, 2019. 214 с.

7. *Девянин П. Н.* Подходы к моделированию управления доступом в СУБД PostgreSQL в рамках МРОСЛ ДП-модели // Прикладная дискретная математика. Приложение. 2018. № 11. С. 95–98.

УДК 004.056.53, 004.032.26 DOI 10.17223/2226308X/12/46

ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ КАК МЕХАНИЗМ ОБФУСКАЦИИ ВЫЧИСЛЕНИЙ

В. Л. Елисеев

Рассмотрены возможности использования искусственных нейронных сетей в качестве механизма строгой обфускации вычислительного алгоритма. Обсуждается задача обфускации, основные положения и современные подходы к её реализации. Вводится понятие нейросетевого обфускатора и доказываются его свойства. Приводятся достоинства и недостатки предложенного подхода.

Ключевые слова: *искусственная нейронная сеть, обфускация.*

Введение

Обфускация (сокрытие) представляет собой преобразование алгоритма к форме, затрудняющей изучение его свойств. Актуальность рассмотрения подобных механизмов основана на том факте, что программное обеспечение в подавляющем большинстве случаев представляет собой объект интеллектуальной собственности, но при этом используется в вычислительной среде, не контролируемой владельцем прав на программу. Наиболее часто обфускации подвергаются части программы, реализующие механизмы защиты от копирования и иного использования в обход лицензии. Примитивные механизмы обфускации применяются также для защиты от изучения человеком исходного кода библиотек, поставляемых на интерпретируемых языках (PHP, Python, JavaScript, Java).

Значительное число работ, посвящённых обфускации исходного кода программ (например, [1]), предлагает разнообразные эвристические методы, автоматизирующие эквивалентную трансформацию программы с целью запутывания оригинальной структуры алгоритма путём изменения представления используемых данных и констант. Такое преобразование даёт визуально достаточно впечатляющие результаты, однако для каждого из эвристических обфусцирующих алгоритмов есть деобфусцирующий, восстанавливающий исходную структуру. По этой причине эвристическая обфускация не может считаться надёжным подходом для защиты алгоритмов, содержащих конфиденциальную информацию, в том числе криптографические ключи и «ноу хау» (скоринговые алгоритмы страховых компаний и банков). Ряд задач, решение которых могло бы упроститься при создании стойких обфускаторов, приведено в [2].

1. Обзор

Начиная с момента появления до работы [3], все методы обфускации представляли собой эвристики, которые было трудно оценивать и сравнивать друг с другом в части стойкости к действиям злоумышленника, пытающегося восстановить исходный алгоритм. Общеизвестно, что любые эвристические подходы всегда имеют ограничения по области применения. В случае с обфускацией для каждого эвристического метода можно построить алгоритм деобфускации, восстанавливающий исходный алгоритм.