УДК 004.052.42

Верификация политик разграничения доступа на основе атрибутов в облачных инфраструктурах с помощью метода проверки на модели

Котенко И. В., Левшун Д. С., Саенко И. Б.

Актуальность работы: *Разграничение доступа на основе атрибутов представляет собой перспективную модель контроля доступа для облачных инфраструктур, поскольку подобная инфраструктура включает в себя большое количество пользователей, ресурсов и динамически изменяемых прав доступа, задача верификации систем доступа, основанных на данной модели, не исследована.* ***Цель работы*** *заключается в разработке моделей политик разграничения доступа на основе атрибутов и подхода к их экспериментальной проверке с помощью метода проверки на модели.* ***Используемые методы:*** *темпоральная логика, метод проверки на модели.* ***Новизна*** *полученных результатов заключается в следующем: предложен теоретический фундамент применения метода проверки на модели для верификации политик разграничения доступа на основе атрибутов; выполнена реализация предлагаемого подхода на примере небольшой организации; продемонстрирован порядок применения предлагаемого подхода для выявления и устранения противоречий в политиках доступа на основе атрибутов.* ***Результат:*** *возможность применения метода проверки на модели для верификации политик доступа на основе атрибутов продемонстрирована на основе проведенных экспериментов. Реализация выполнена с использованием инструмента верификации UPPAAL.* ***Практическая значимость:*** *сложность верификации политик доступа на основе атрибутов увеличивается экспоненциально с ростом количества правил, а потому ручная проверка сложных систем контроля неприемлема. Для автоматизации данного процесса предложен и экспериментально проверен подход, основанный на методе проверки на модели.*

Ключевые слова: *разграничение доступа, проверка на модели, темпоральная логика, разграничение доступа на основе атрибутов, облачная инфраструктура.*

Введение

Разграничение доступа играет важнейшую роль в обеспечении компьютерной и сетевой безопасности в облачных инфраструктурах, пользователи которых должны обладать разными полномочиями по выполнению различных действий над информационными ресурсами [1, 2]. Облачные инфраструктуры лежат в основе как больших информационных систем коллективного пользования, так и многих киберфизических систем (умный город, умный дом, автоматизированное производство, робототехника и т.д.) [3, 4]. Для решения задач разграничения доступа в этих системах разработано несколько моделей контроля доступа, которые считаются традиционными. Такими моделями являются: дискреционное управление доступом (discretionary access control, DAC), мандатное управление доступом (mandatory access control, MAC), а также

Библиографическая ссылка на статью:
Котенко И. В., Левшун Д. С., Саенко И. Б. Верификация политик разграничения доступа на основе атрибутов в облачных инфраструктурах с помощью метода проверки на модели // Системы управления, связи и безопасности. 2019. № 4. С. 421-436. DOI: 10.24411/2410-9916-2019-10417.

Reference for citation:
Kotenko I. V., Levshun D. S., Saenko I. B. Verification of Attribute-based Access Control Policies in Cloud Infrastructures based on Model Checking. *Systems of Control, Communication and Security*, 2019, no. 4, pp. 421-436. DOI: 10.24411/2410-9916-2019-10417 (in Russian).

управление доступом на основе ролей (role-based access control, RBAC). Однако опыт использования традиционных моделей контроля доступа показал, что в условиях высокой динамики изменения требуемых полномочий, возникающих при изменении характеристик (атрибутов) пользователей, ресурсов или среды, данные модели становятся неэффективными. Возникает потребность в использовании новых, более гибких моделей контроля доступа.

Одной из таких достаточно гибких моделей контроля доступа, которая появилась сравнительно недавно, является модель разграничения доступа на основе атрибутов (attribute-based access control, ABAC) [5]. Эта модель может успешно заменить традиционные модели контроля доступа [6]. Разрешение на выполнение тех или иных действий над ресурсами (объектами) в этой модели выдается на основании проверки корректности выполнения множества логических условий (правил), которые определяют используемую политику контроля доступа. Правила формируются в виде логических выражений, в которых используются значения атрибутов. Все множество атрибутов состоит из трех групп: атрибутов пользователей (субъектов), атрибутов ресурсов (объектов) и атрибутов компьютерного окружения. К последней группе относится время. По этой причине ABAC модель является более гибкой, чем другие модели контроля доступа, и способной быстро реагировать на изменения.

Однако, в отличие от традиционных DAC, MAC и RBAC моделей, ABAC модель еще во многом находится на исследовательском уровне. Многие вопросы, касающиеся разработки и использования политик на основе ABAC, еще не до конца исследованы. Поэтому разработчики средств защиты еще не перешли к широкому внедрению ABAC в своих продуктах. Одним из таких проблемных вопросов является верификация политик, основанных на ABAC. Задачами верификации ABAC политик является нахождение противоречий в правилах контроля доступа и способов устранения этих противоречий.

В настоящей статье исследуется возможность применения для верификации ABAC политик подхода на основе проверки на модели (model checking). Проверка на модели осуществляется с использованием темпоральной логики и ориентирована на анализ множества возможных состояний логической системы. Для практической реализации разработано множество программных средств, которые нашли успешное применение при решении задач верификации во многих сценариях. Однако для верификации ABAC политик данный метод еще не исследовался. Этим определяется теоретический вклад статьи. Новизна полученных результатов заключается в следующем:

1) предложен теоретический фундамент для применения метода проверки на модели к верификации ABAC политик;
2) выполнена реализация этого метода для фрагмента ABAC политики;
3) продемонстрирован порядок применения подхода для выявления и устранения противоречий в ABAC политиках.

Статья имеет следующую структуру. Во втором разделе приводится анализ современного состояния исследований. Третий и четвертый разделы посвящены теоретическим основам разграничения доступа на основе атрибутов. Предлагаемый подход к верификации представлен в пятом разделе. В шестом

разделе описаны результаты проведенных экспериментов. В заключении содержатся основные выводы, и представлены направления дальнейших исследований.

Анализ релевантных работ

Рассмотрим исследования в области моделирования ABAC политик и применения подходов на основе проверки на модели к их верификации более подробно.

В [7] представлен подход к разграничению доступа на основе атрибутов для улучшения обмена данными внутри организации с учетом планирования, проектирования, внедрения и эксплуатации. В данной работе представлены стандарты ABAC, область применения разграничения доступа на основе атрибутов, а также нерешенные проблемы, связанные с его использованием и верификацией. Однако вопросы автоматизированной верификации ABAC политик в данной работе не были рассмотрены.

В [8] рассмотрена одна из важнейших задач – решение проблемы оптимизации структуры ABAC модели. В качестве одного из возможных решений предложено использование методов глубокого обучения (deep learning). При этом в работе предполагается, что используемая модель контроля доступа не содержит аномалий, а задача верификации является одним из направлений дальнейших исследований.

Анализ текущих проблем в области моделирования ABAC представлен в [9]. Среди различных проблем выделяется проблема формального анализа безопасности ABAC модели. Верификация ABAC политик является частью этой проблемы. При этом в данной работе подчеркивается, что во многих работах, например, в [10–12], анализ политик осуществлен независимо от формальной модели контроля доступа. Хотя многие из этих решений применимы к ABAC политикам, сами по себе они не могут обеспечить полный анализ безопасности модели ABAC без учета свойств формальной модели и способа комбинирования и применения политик. В этой связи задача верификации политик ABAC модели приобретает достаточно большое значение. Однако эта задача в настоящее время решается, в основном, за счет доверенной третьей стороны [13].

В [14] предлагается решать задачу верификации политик контроля доступа посредством применения заранее разработанных шаблонов. Этот подход упрощает проектирование системы безопасности. Однако он не подходит для анализа безопасности и верификации в режиме реального времени.

В [15] представлена графическая модель для упрощения спецификации ограничений и верификации. Визуализация в настоящее время является достаточно перспективным направлением анализа безопасности. Она приводит к появлению новых моделей контроля доступа, в которых используются графические элементы, например, модель визуализации контроля доступа на основе треугольных матриц [16]. Однако ни одна из графических моделей контроля доступа не может обеспечить требуемую скорость верификации.

Достижение требуемого качества верификации ABAC политик видится в использовании методов автоматической верификации. Среди этих методов достаточно распространенным и хорошо разработанным является метод проверки на модели. Так, в [17] рассматривается применение данного метода для верификации правил авторизации политики безопасности для мобильных систем. В [18] предлагается подход, основанный на проверке на модели, для обнаружения аномалий фильтрации. В [19] рассматривается подход для формального моделирования и анализа реализации атак на компьютерную сеть. Эти работы демонстрируют достаточно высокую эффективность применения проверки на модели для анализа и верификации различных систем безопасности и дают основания полагать, что этот метод может также успешно применяться для верификации ABAC моделей.

Модель разграничения доступа на основе атрибутов

В отличие от RBAC, в ABAC разграничение доступа пользователей обеспечивается на основе атрибутов, а не ролей. Атрибуты, участвующие в формировании условий доступа, сгруппированы в три категории: атрибуты субъектов доступа, атрибуты информационных ресурсов и атрибуты окружающей среды. Значения этих атрибутов участвуют в формировании правил, на основании которых принимается решение на разрешение или запрет доступа. Операции доступа относятся к информационным ресурсам и их атрибутам. В результате ABAC позволяет строить более гибкие схемы доступа, чем RBAC, которые отличаются способностью хорошо адаптироваться к высокой динамике изменения политики безопасности, свойственной современным крупномасштабным информационным системам.

Как для RBAC, так и ABAC существует проблема формирования схемы разграничения доступа. В модели RBAC эта проблема получила название Role Mining Problem (RMP). В модели ABAC некоторые исследователи предлагают называть эту проблему ABAC Policies Mining Problem (APMP). Ее суть заключается в следующем.

Пусть даны множество пользователей (U), ресурсов (R) и операций (O), которые пользователи могут выполнять над ресурсами. Атрибуты разделяются на два типа: для пользователей (A_u) и ресурсов (A_r). Атрибут a пользователя u или ресурса r может принимать пустое значение или значение из своего домена D_a. Это значение обозначается с помощью отношений $a(u)$ или $a(r)$. Правило политики $p = <e;o>$ в модели ABAC задается выражением, которое определяет условие применимости (e) и выполняемое действие (o). Покажем это на следующем примере.

Пусть атрибут пользователя A_u имеет имя «отдел» - «*Department*» ($A_u = Department$) и атрибут ресурса A_r имеет имя «Владелец» - «*Owner*» ($A_r = Owner$). Пользователь u может выполнять над ресурсом r действие $o = read$, если выполняется одно из двух условий: либо пользователь u работает в отделе управления («*Management*»), либо он является владельцем ресурса r. Формальное представление этого правила имеет следующий вид:

$$p = \langle A_u(u) = Management\ OR\ A_r(r) = u;\ read(u,r) \rangle \quad (1)$$

где $read(u,r)$ – операция чтения, выполняемая u над r.

Проблема нахождения политики (схемы) разграничения доступа в модели ABAC (АРМР) формулируется следующим образом. Пусть имеются журнал событий L, состоящий из записей вида <u, r, o, t>, обозначающих тот факт, что пользователь u выполняет над ресурсом r действие o в момент времени t. Требуется найти такую политику, которая максимизирует ее показатель качества.

Теоретические основы проверки на модели

Верификация политик контроля доступа на предмет аномалий правил с помощью метода проверки на модели сводится к следующим действиям. Вначале осуществляется построение модели информационной системы, в которой применяются политики безопасности. Затем задается спецификация этой системы с помощью линейной темпоральной логики.

Модель информационной системы предназначена для представления взаимосвязей пользователей, ресурсов и действий, их атрибутов и задействованных информационных процессов. Она включает в себя два базовых компонента: конфигурацию системы и политики контроля доступа. Конфигурация системы представляется множеством пользователей с установленными между ними логическими связями и множеством информационных ресурсов.

Верификация политик разграничения доступа включает в себя следующие этапы [20]:

1) построение модели информационной системы во внутреннем формате системы верификации в виде конечного автомата;
2) построение спецификации на проверяемую систему, задающей свойства корректности на языке темпоральной логики;
3) вычисление модели с помощью программного средства;
4) обработка результатов верификации и построенных контрольных примеров, показывающих, каким образом система может перейти в некорректное состояние;
5) сравнение и оценка результатов верификации в соответствии с требованиями к их эффективности.

Для построения модели информационной системы в методе проверки на модели принято использовать модель Крипке [21]. Она состоит из множества состояний, множества переходов между состояниями и функции, которая помечает каждое состояние набором свойств, истинных в этом состоянии. Детализация данной модели приведена в более ранних работах авторов [22].

Подход к верификации

Рассмотрим небольшую организацию, состоящую из пяти сотрудников: двух начальников отделов и трех рабочих (рис. 1). Начальники могут работать только в своем отделе, в то время как рабочие могут быть переведены из одного

отдела в другой. Каждый сотрудник может создавать файлы и работать с ними. Начальники могут также работать со всеми файлами, созданными в их отделе.

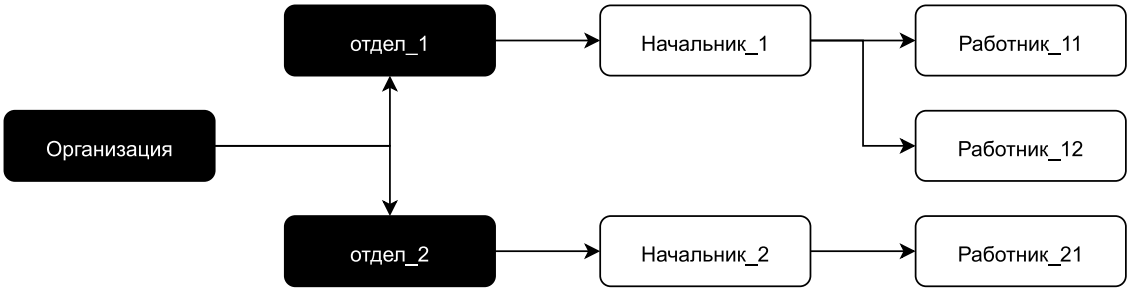

Рис. 1. Иерархия сотрудников организации

У каждого сотрудника есть три атрибута: *personal_id* – уникальный идентификатор сотрудника в организации (помогает отличить одного сотрудника от другого); *role_id* – идентификатор роли сотрудника (*Рабочий* или *Начальник* в нашем примере); *department_id* – идентификатор отдела, в котором работает сотрудник (*отдел_1* или *отдел_2* в нашем примере).

У *Файла*, созданного сотрудником, есть два атрибута: *owner_id* – идентификатор сотрудника (его или её *personal_id*); *department_id* – идентификатор отдела, в котором работал сотрудник в момент создания файла (*отдел_1* или *отдел_2* в нашем примере).

В соответствии с политикой безопасности, доступ к *Файлу* для сотрудника, роль которого – *Начальник*, предоставляется, если *department_id* сотрудника и *Файла* совпадают. Для сотрудников, роль которых – *Рабочий*, доступ к *Файлу* предоставляется, если *personal_id* сотрудника и *owner_id Файла* совпадают. Во всех остальных случаях в доступе должно быть отказано.

Подобная политика безопасности позволит избежать ситуаций, в которых сотрудники *отдела_2* будут иметь возможность работать с файлами, созданными сотрудниками *отдела_1* и наоборот. Кроме того, если *Рабочий_11* из *отдела_1* создаст *Файл*, и затем *Рабочий_11* будет переведен в *отдел_2*, то *Начальник_1* все еще будет иметь доступ к созданному *Файлу* (за счет атрибута *department_id*).

Недостаток подобной политики заключается в том, что *Рабочий_11* сохранит доступ к созданному *Файлу* за счет атрибута *owner_id* даже будучи сотрудником *отдела_2*. Для предотвращения подобных ситуаций, политика безопасности может быть изменена следующим образом: для сотрудников, роль которых – *Рабочий*, доступ к *Файлу* предоставляется тогда и только тогда, когда *personal_id* сотрудника и *owner_id Файла* совпадают точно также как их *department_id*.

Отметим, что сложность верификации политик доступа растет с каждым добавленным правилом, а потому их ручная проверка занимает все больше временных ресурсов. Для автоматизации данного процесса используется под-

№4. 2019
ISSN 2410-9916

ход, основанный на проверке на модели. В следующем разделе смоделируем представленный пример в среде UPPAAL и верифицируем его.

Экспериментальная проверка

UPPAAL представляет собой инструмент для моделирования и верификации систем реального времени. При этом системы представляются в виде конечных автоматов [23]. Конечные автоматы *Сотрудника* и *Файла*, основанные на примере из предыдущего раздела, представлены на рис. 2 и рис. 3 соответственно. Используемые на рисунках обозначения соответствуют синтаксису среды UPPAAL: окружности – состояния (двойная окружность – начальное состояние), направленные дуги – переходы между состояниями, зеленый текст – условия перехода между состояниями (равенство "==", логическое И "&&", логическое ИЛИ "||"), синий текст – изменение значений переменных, голубой текст – параметры синхронизации между конечными автоматами.

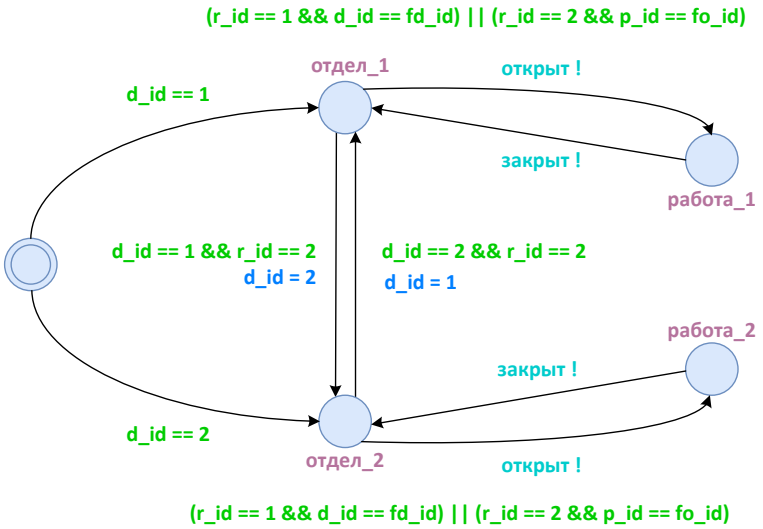

Рис. 2. Конечный автомат сотрудника в среде UPPAAL

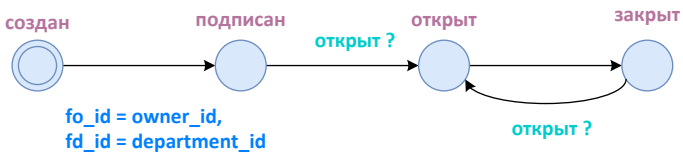

Рис. 3. Конечный автомат файла в среде UPPAAL

Конечный автомат *Сотрудника* состоит из пяти состояний: начального состояния; *отдел_1* (сотрудник работает в первом отделе); *отдел_2* (сотрудник работает во втором отделе); *работа_1* (процесс работы с файлами первого отдела); *работа_2* (процесс работы с файлами второго отдела).

Кроме того, у сотрудника есть три параметра: *p_id* (соответствует *personal_id*), *r_id* (соответствует *role_id*) и *d_id* (соответствует *department_id*). Переходы между состояниями представлены в виде направленных ребер с соответ-

DOI: 10.24411/2410-9916-2019-10417
URL: https://sccs.intelgr.com/archive/2019-04/17-Kotenko.pdf

ствующими правилами доступа и синхронизации (например, работа с файлом на основе *открыть ! закрыть !*).

Конечный автомат *Файла* состоит из четырех состояний: создан (начальное состояние); подписан (привязка к сотруднику через *owner_id* и отделу через *department_id*); открыт; закрыт. Переходы между состояниями представлены в виде направленных ребер с соответствующими правилами синхронизации (например, работа с файлом на основе *открыт ? закрыт ?*). Использованные при моделировании параметры *Файла* отражают, что он был создан *Рабочим_11* в то время как данный сотрудник работал в *отделе_1*.

Для моделирования и верификации, в среде UPPAAL была создана система, состоящая из пяти сотрудников и одного файла (рис. 4).

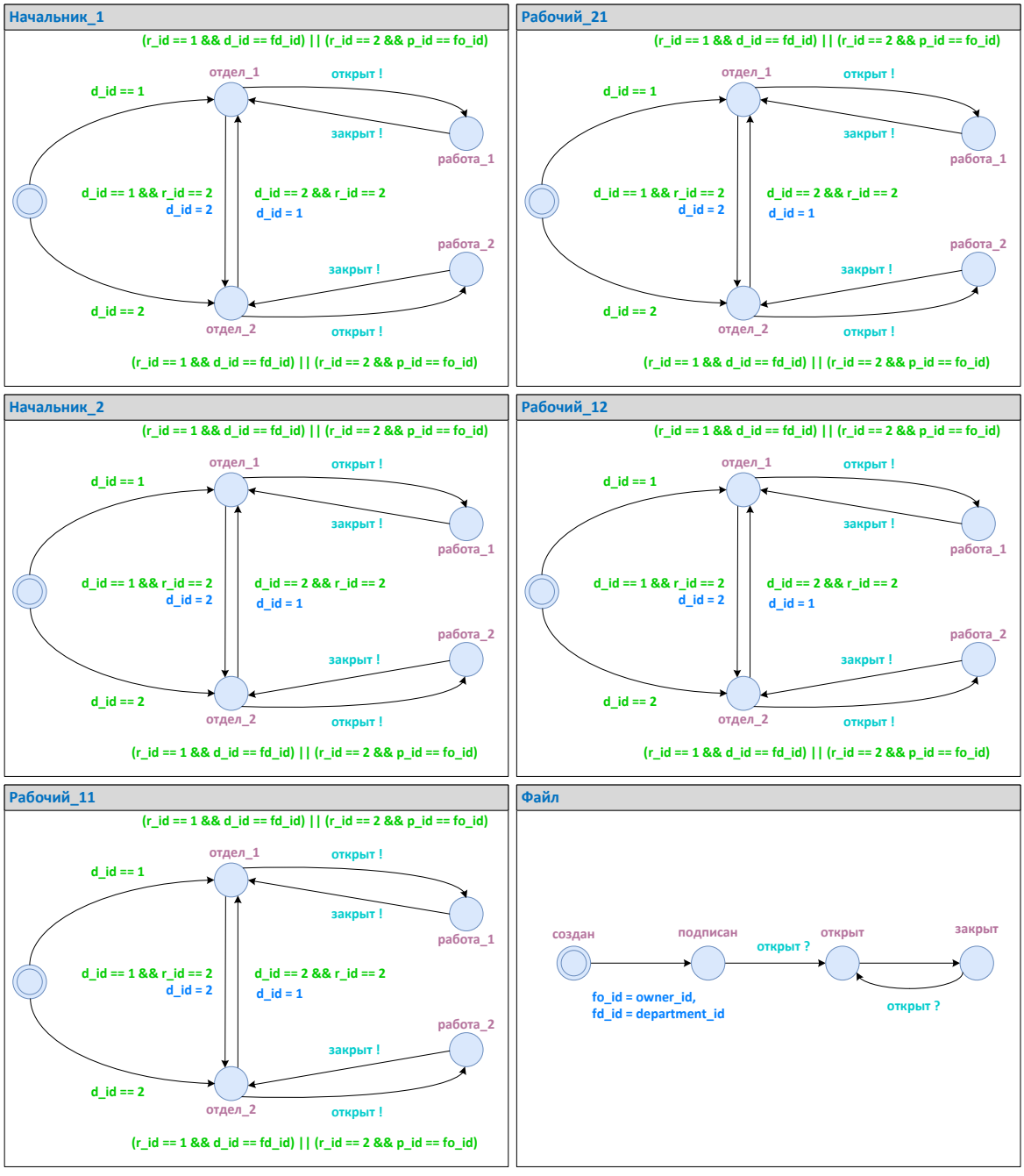

Рис. 4. Моделирование системы в среде UPPAAL

Листинг исходного кода представлен ниже:

```
// Place template instantiations here.
Boss_1 = Employee(1, 1, 1);
Worker_11 = Employee(2, 2, 1);
Worker_12 = Employee(3, 2, 1);

Boss_2 = Employee(4, 1, 2);
Worker_21 = Employee(5, 2, 2);

// List one or more processes to be composed into a
system.
system Boss_1, Worker_11, Worker_12, Boss_2, Work-
er 21, File;
```

Правило 1. Доступ к *Файлу* для сотрудника, роль которого – *Начальник*, предоставляется тогда и только тогда, когда *department_id* сотрудника и *Файла* совпадают. В среде UPPAAL данное правило представимо следующим образом: *r_id* == 1 && *d_id* == *fd_id* и является правилом перехода между состояниями *отдел_1* → *работа_1*, *отдел_2* → *работа_2*.

Правило 2. Доступ к *Файлу* для сотрудника, роль которого – *Рабочий*, предоставляется тогда и только тогда, когда *personal_id* сотрудника совпадает с *owner_id Файла*. В среде UPPAAL данное правило представимо следующим образом: *r_id* == 2 && *p_id* == *fo_id* и является правилом перехода между состояниями *отдел_1* → *работа_1*, *отдел_2* → *работа_2*.

Для верификации политики безопасности были проверены следующие параметры: (1) переход сотрудников между отделами и (2) возможность работы с созданными файлами (таблица 1).

Таблица 1 – Проверка политики безопасности

Правило	**Результат**
E<> not *Начальник_1.работа_1* and not *Рабочий_11.работа_1* and not *Рабочий_11.работа_2* and *Файл.открыт*	*False*
E<> not *Начальник_1.работа_1* and *Рабочий_11.работа_2* and *Файл.открыт*	*True*
E<> not *Начальник_1.работа_1* and *Рабочий_11.работа_1* and *Файл.открыт*	*True*
E<> *Начальник_1.работа_1* and not *Рабочий_11.работа_1* and not *Рабочий_11.работа_2* and *Файл.открыт*	*True*
E<> *Рабочий_11.отдел_2*	*True*
E<> *Начальник_1.отдел_2*	*False*
A[] not deadlock	*True*

Рассмотрим выражение «E<> not *Начальник_1.работа_1* and not *Рабочий_11.работа_1* and not *Рабочий_11.работа_2* and *Файл.открыт*» более подробно. Данное выражение дает ответ на следующий вопрос: может ли кто-то

работать с файлом (*and Файл.открыт*), при условии, что с ним в данный момент не работают: *Начальник_1* из *отдела_1* (not *Начальник_1.работа_1*); *Рабочий_11* из *отдела_1* (and not *Рабочий_11.работа*_1); *Рабочий 11* из *отдела_2* (and not *Рабочий_11.работа_2*).

Так как по результатам проверки на модели получен ответ *False*, может быть сделан вывод, что доступ к *Файлу* может быть предоставлен только *Начальнику_1*, который является руководителем *отдела_1*, и *Рабочему_11*, который является создателем *Файла* и в тот момент работал в *отделе_1*.

Выражение «E<> not *Начальник_1.работа_1* and *Рабочий_11.работа_2* and *Файл.открыт*» дает ответ на вопрос может ли *Файл* быть открыт, если *Начальник_1* на данный момент с ним не работает, а *Рабочий_11* работает с этим *Файлом* во втором отделе.

Так как по результатам проверки на модели получен ответ *True,* то доступ к *Файлу* может быть предоставлен его владельцу, даже если он был переведен в другой отдел (проблема политики безопасности, которая упоминалась ранее).

Для предотвращения подобных ситуаций, второе правило политики безопасности должно быть изменено следующим образом: доступ к *Файлу* для сотрудника, роль которого – *Рабочий*, предоставляется тогда и только тогда, когда *personal_id* сотрудника совпадает с *owner_id Файла* точно также как *department_id* сотрудника и *Файла*.

В среде UPPAAL данное правило представимо следующим образом: *r_id* == 2 && *p_id == fo_id* и является правилом перехода между состояниями *отдел_1 → работа_1*, *отдел_2 → работа_2*. Проведем повторную верификацию политики безопасности (таблица 2).

Таблица 2 – Повторная проверка политики безопасности

Правило	**Результат**
E<> not *Начальник_1.работа_1* and not *Рабочий_11.работа_1* and not *Рабочий_11.работа_2* and *Файл.открыт*	*False*
E<> not *Начальник_1.работа_1* and *Рабочий_11.работа_2* and *Файл.открыт*	*False*
E<> not *Начальник_1.работа_1* and *Рабочий_11.работа_1* and *Файл.открыт*	*True*
E<> *Начальник_1.работа_1* and not *Рабочий_11.работа_1* and not *Рабочий_11.работа_2* and *Файл.открыт*	*True*
E<> *Рабочий_11.отдел_2*	*True*
E<> *Начальник_1.отдел_2*	*False*
A[] not deadlock	*True*

Изменение политики безопасности привело к тому, что по результатам проверки на модели выражения «E<> not *Начальник_1.работа_1* and *Рабочий_11.работа_2* and *Файл.открыт*» был получен ответ *False*. Т.е. если *Начальник_1* не работает с *Файлом*, а *Рабочий_11* на данный момент работает в *отделе_2*, то *Файл* не может быть в состоянии *открыт*. При этом результаты

проверки остальных выражений остались прежними. Это означает, что внесенные изменения, с одной стороны, решили существующую проблему, а с другой – не добавили новых проблем.

Для удобства работы с представленным в данном разделе экспериментом, разработанные в среде UPPAAL модели доступны для загрузки в GitHub репозитории [24].

Заключение

В данной работе представлен новый подход к верификации политик разграничения доступа на основе атрибутов в облачных инфраструктурах посредством проверки на модели. На основе представленной модели ABAC и применения метода проверки на модели, был построен эксперимент в среде UPPAAL.

Данный эксперимент заключался в спецификации политики безопасности небольшой компании в терминах ABAC и демонстрации применимости предлагаемого подхода к верификации. Верификация политики безопасности в среде UPPAAL позволила обнаружить в ней недостатки, а также подтвердить, что после ее доработки обнаруженные недостатки устраняются, а новые недостатки отсутствуют.

При этом важно отметить, что конечные автоматы, используемые для моделирования в среде UPPAAL, отлично подходят для графического представления небольших политик доступа. Однако данный инструмент не подходит для моделирования более сложных политик в виду большого количества ручной работы. И хотя для проведения текущих экспериментов возможностей среды UPPAAL было достаточно, в рамках дальнейших исследований планируется переход к инструментам, позволяющим верифицировать распределенные модели (например, SPIN [25]).

Работа выполнена при частичной финансовой поддержке РФФИ (проекты 18-07-01488 и 18-29-22034) и бюджетной темы 0073-2019-0002.

Литература

1. Subashini S., Kavitha V. A survey on security issues in service delivery models of cloud computing // Journal of network and computer applications. 2011. Vol. 34. № 1. P. 1-11.

2. Karatas G., Akbulut A. Survey on Access Control Mechanisms in Cloud Computing // Journal of Cyber Security and Mobility. 2018. Vol. 7. № 3. P. 1-36.

3. Lopez J., Rubio J. E. Access control for cyber-physical systems interconnected to the cloud // Computer Networks. 2018. Vol. 134. P. 46-54.

4. Котенко И. В., Десницкий В. А., Чечулин А. А. Исследование технологии проектирования безопасных встроенных систем в проекте Европейского сообщества SecFutur // Защита информации. Инсайд. 2011. № 3 (39). С.68-75.

5. Hu V. C., Ferraiolo D., Kuhn R., Schnitzer A., Sandlin K., Miller R., Scarfone K. Guide to attribute based access control (ABAC) definition and considerations // NIST special publication. 2014. Vol. 800. № 162. 54 p.

6. Kuhn D. R., Coyne E. J., Weil T. R. Adding attributes to role-based access control // Computer. 2010. Vol. 43. № 6. P. 79-81.

7. Hu V. C., Kuhn R., Ferraiolo D., Voas J. Attribute-based access control // Computer. 2015. Vol. 48. № 2. P. 85-88.

8. Mocanu D., Turkmen F., Liotta A. Towards ABAC policy mining from logs with deep learning // Proceedings of the 18th International Multiconference, IS2015. 2015. P. 124-128.

9. Servos D., Osborn S. L. Current research and open problems in attribute-based access control // ACM Computing Surveys (CSUR). 2017. Vol. 49. № 4. P. 65.

10. Fisler K., Krishnamurthi S., Meyerovich L. A., Tschantz M. C. Verification and change-impact analysis of access-control policies // Proceedings of the 27th international conference on Software engineering. ACM, 2005. P. 196-205.

11. Kolovski V., Hendler J., Parsia B. Analyzing web access control policies // Proceedings of the 16th international conference on World Wide Web. ACM, 2007. P. 677-686.

12. Lin D., Rao P., Bertino E., Li N., Lobo J. EXAM: a comprehensive environment for the analysis of access control policies // International Journal of Information Security. 2010. Vol. 9. № 4. P. 253-273.

13. Lee A. J. Credential-based access control // Encyclopedia of cryptography and security. 2011. P. 271-272.

14. Deng Y., Wang J., Tsai J. J., Beznosov K. An approach for modeling and analysis of security system architectures // IEEE Transactions on knowledge and data engineering. 2003. Vol. 15. № 5. P. 1099-1119.

15. Jaeger T., Tidswell J. E. Practical safety in flexible access control models // ACM Transactions on Information and System Security (TISSEC). 2001. Vol. 4. № 2. P. 158-190.

16. Kolomeets M., Chechulin A., Kotenko I., Saenko I. Access Control Visualization Using Triangular Matrices // 27th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP-2019). IEEE, 2019. P. 348-355.

17. Braghin C., Sharygina N., Barone-Adesi K. A model checking-based approach for security policy verification of mobile systems // Formal Aspects of Computing. 2011. Vol. 23. № 5. P. 627-648.

18. Kotenko I., Polubelova O. Verification of security policy filtering rules by model checking // Proceedings of the 6th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems. IEEE, 2011. Vol. 2. P. 706-710.

19. Rothmaier G., Kneiphoff T., Krumm H. Using SPIN and Eclipse for optimized high-level modeling and analysis of computer network attack models // International SPIN Workshop on Model Checking of Software. – Springer, Berlin, Heidelberg, 2005. – P. 236-250.

20. Котенко И. В., Саенко И. Б. Методика верификации политик безопасности в многоуровневой интеллектуальной системе обеспечения комплексной безопасности железнодорожного транспорта // Технические науки – от теории к практике. Новосибирск: Изд. «СибАК», 2014. № 30. С.18-22.

21. Clarke E. M., Grumberg O., Peled D. Model Checking. MIT Press, 2000. 46 p.

22. Полубелова О. В., Котенко И. В. Верификация правил фильтрации с временными характеристиками методом «проверки на модели» // Труды СПИИРАН. 2012. Т. 3. № 22. С. 113-138.

23. Larsen K. G., Pettersson P., Yi W. UPPAAL in a nutshell // International journal on software tools for technology transfer. 1997. Vol. 1. № 1-2. P. 134-152.

24. Верификация политик разграничения доступа на основе атрибутов в облачных инфраструктурах с помощью метода проверки на модели // GitHub [Электронный ресурс]. 2019. – URL: https://github.com/levshun/SCCS-UPPAAL/ (дата обращения: 01.12.2019).

25. Holzmann G. J. The model checker SPIN // IEEE Transactions on software engineering. 1997. Vol. 23. № 5. P. 279-295.

References

1. Subashini S., Kavitha V. A survey on security issues in service delivery models of cloud computing. *Journal of network and computer applications*, 2011, vol. 34, no. 1, pp. 1-11.

2. Karatas G., Akbulut A. Survey on Access Control Mechanisms in Cloud Computing. *Journal of Cyber Security and Mobility*, 2018, vol. 7, no. 3, pp.1-36.

3. Lopez J., Rubio J. E. Access control for cyber-physical systems interconnected to the cloud. *Computer Networks*, 2018, vol. 134, pp. 46-54.

4. Kotenko I. V., Desnitsky V. A., Chechulin A. A. Research of the technology for designing secure embedded systems in a project of the European Community SecFutur. *Zasita informacii. Inside* [Protection of information. Inside]. 2011, no. 3 (39), pp.68-75. (in Russian).

5. Hu V. C., Ferraiolo D., Kuhn R., Schnitzer A., Sandlin K., Miller R., Scarfone K. Guide to attribute based access control (ABAC) definition and considerations. *NIST special publication*, 2014, vol. 800, no. 162. 54 p.

6. Kuhn D. R., Coyne E. J., Weil T. R. Adding attributes to role-based access control. *Computer*, 2010, vol. 43, no. 6, pp. 79-81.

7. Hu V. C., Kuhn R., Ferraiolo D., Voas J. Attribute-based access control. *Computer*, 2015, vol. 48, no. 2, pp. 85-88.

8. Mocanu D., Turkmen F., Liotta A. Towards ABAC policy mining from logs with deep learning. *Proceedings of the 18th International Multiconference*. IS2015, 2015, pp. 124-128.

9. Servos D., Osborn S. L. Current research and open problems in attribute-based access control. *ACM Computing Surveys (CSUR)*, 2017, vol. 49, no. 4, pp. 65.

10. Fisler K., Krishnamurthi S., Meyerovich L. A., Tschantz M. C. Verification and change-impact analysis of access-control policies. *Proceedings of the 27th international conference on Software engineering*. ACM, 2005, pp. 196-205.

11. Kolovski V., Hendler J., Parsia B. Analyzing web access control policies. *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 677-686.

12. Lin D., Rao P., Bertino E., Li N., Lobo J. EXAM: a comprehensive environment for the analysis of access control policies. *International Journal of Information Security*, 2010, vol. 9, no. 4, pp. 253-273.

13. Lee A. J. Credential-based access control. *Encyclopedia of cryptography and security*, 2011, pp. 271-272.

14. Deng Y., Wang J., Tsai J. J., Beznosov K. An approach for modeling and analysis of security system architectures. *IEEE Transactions on knowledge and data engineering*, 2003, vol. 15, no. 5, pp. 1099-1119.

15. Jaeger T., Tidswell J. E. Practical safety in flexible access control models. *ACM Transactions on Information and System Security (TISSEC)*, 2001, vol. 4, no. 2, pp. 158-190.

16. Kolomeets M., Chechulin A., Kotenko I., Saenko I. Access Control Visualization Using Triangular Matrices. *27th Euromicro International Conference on Parallel, Distributed and Network-Based Processing* (PDP-2019). IEEE, 2019, pp. 348-355.

17. Braghin C., Sharygina N., Barone-Adesi K. A model checking-based approach for security policy verification of mobile systems. *Formal Aspects of Computing*, 2011, vol. 23, no. 5, pp. 627-648.

18. Kotenko I., Polubelova O. Verification of security policy filtering rules by model checking. *Proceedings of the 6th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems*. IEEE, 2011, vol. 2, pp. 706-710.

19. Rothmaier G., Kneiphoff T., Krumm H. Using Spin and Eclipse for optimized high-level modeling and analysis of computer network attack models. *International SPIN Workshop on Model Checking of Software*. Springer, Berlin, Heidelberg, 2005, pp. 236-250.

20. Kotenko I., Saenko I. The technique for verification of security policies in the multilevel intelligent system of integrated protection of railway transport. *Technical science - from theory to practice*. Novosibirsk, SibAK, 2014, no. 30, pp.18-22. (in Russian).

21. Clarke E. M., Grumberg O., Peled D. Model Checking. *MIT Press*, 2000. 46 p.

22. Polubelova O., Kotenko I. Verification of security policy filtering rules containing temporal parameters by Model Checking. *Proceedings of SPIIRAS*, 2012, vol. 3, no. 22, pp. 113-138. (in Russian).

23. Larsen K. G., Pettersson P., Yi W. UPPAAL in a nutshell. *International journal on software tools for technology transfer*, 1997, vol. 1, no. 1-2, pp. 134-152.

24. Verification of Access Control Policies based on Attributes in Cloud Infrastructures based on Model Checking. *GitHub*, 2019. Available at: https://github.com/levshun/SCCS-UPPAAL (accessed 01 December 2019).

25. Holzmann G. J. The model checker SPIN. *IEEE Transactions on software engineering*, 1997, vol. 23, no. 5, pp. 279-295.

Статья поступила 19 декабря 2019 г.

Информация об авторах

Котенко Игорь Витальевич – доктор технических наук, профессор. Заведующий лабораторией проблем компьютерной безопасности. Санкт-Петербургский институт информатики и автоматизации Российской академии наук (СПИИРАН). Область научных интересов: безопасность компьютерных сетей, искусственный интеллект, телекоммуникационные системы. E-mail: ivkote@comsec.spb.ru

Левшун Дмитрий Сергеевич – младший научный сотрудник лаборатории проблем компьютерной безопасности. Санкт-Петербургский институт информатики и автоматизации Российской академии наук (СПИИРАН). Область научных интересов: безопасность распределенных систем, встроенные устройства, корреляция событий безопасности. E-mail: levshun@comsec.spb.ru

Саенко Игорь Борисович – доктор технических наук, профессор. Ведущий научный сотрудник лаборатории проблем компьютерной безопасности. Санкт-Петербургский институт информатики и автоматизации Российской академии наук (СПИИРАН). Область научных интересов: безопасность компьютерных сетей, базы данных, искусственный интеллект, информационные и телекоммуникационные системы. E-mail: ibsaen@comsec.spb.ru

Адрес: 199178, Россия, г. Санкт-Петербург, 14 линия, д. 39.

Verification of Access Control Policies based on Attributes in Cloud Infrastructures based on Model Checking

I. V. Kotenko, D. S. Levshun, I. B. Saenko

Purpose. *Attribute-Based Access Control (ABAC) is a promising access control model for cloud infrastructures, since such an infrastructure includes a large number of users, resources, and dynamically changing access rights.* ***The purpose*** *is to evaluate the effectiveness of verification of ABAC policies by model checking approach.* ***Methods.*** *Temporal logics, model checking.* ***Novelty.*** *The theoretical background for application of the model-checking to ABAC policies verification is considered. The implementation of the model checking for a fragment of ABAC policy is developed. the identification and elimination of contradictions in ABAC policies is shown.* ***Results.*** *The possibility of using the model verification method to verify ABAC policies is validated by experiment. Implementation was performed using the UPPAAL verification tool.* ***Practical relevance.*** *The complexity of verification of attribute access policies grows with each rule added, and therefore, their manual verification takes up more and more time resources. To automate this process, an approach based on model verification was proposed and experimentally tested.*

Key words: *access control, model checking, temporal logics, ABAC, cloud infrastructure*

Information about Authors

Igor Vitalievich Kotenko – Dr. habil. of Engineering Sciences, Professor. Head of Laboratory of Computer Security Problems. St. Petersburg Institute for Informatics and Automation of the Russian Academy of Science (SPIIRAS). Field of research: information security, artificial intelligence, telecommunications. E-mail: ivkote@comsec.spb.ru

Dmitry Sergeevich Levshun – Junior Research Associate of Laboratory of Computer Security Problems. St. Petersburg Institute for Informatics and Automation of the Russian Academy of Science (SPIIRAS). Field of research: distributed system security, embedded devices, event correlation. E-mail: levshun@comsec.spb.ru

Igor Borisovich Saenko – Dr. habil. of Engineering Sciences, Professor. Leading Research Associate of Laboratory of Computer Security Problems. St. Petersburg Institute for Informatics and Automation of the Russian Academy of Science (SPIIRAS). Field of research: computer network security, databases, artificial intelligence, information and telecommunication systems. E-mail: ibsaen@comsec.spb.ru

Address: Russia, 199178, Saint-Petersburg, 14th Liniya, 39.