Использование управления доступом на основе атрибутов в протоколе OAuth 2.0

А.В. Беловодов, О.Р. Лапонина

***Аннотация* — данная статья представляет собой исследовательскую работу на тему возможностей применения управления доступом на основе ролей (RBAC) и управления доступом на основе атрибутов (ABAC) совместно с использованием открытого протокола авторизации OAuth 2.0. В статье рассматриваются основные понятия, связанные с моделями авторизации, атрибутами, и тем, как существующие решения сталкиваются с рядом проблем в современном мире, а также возможные методы их решения. В статье предложена модель, предназначенная для управления доступом на основе атрибутов для кросс-доменных источников с использованием API. Модель включает базовые архитектурные решения и принципы ABAC и OAuth. Служба авторизации ABAC рассматривается как микросервис или набор микросервисов. Это обеспечит совместимость архитектуры и развертывания с приложениями на основе микросервисов. Комбинирование возможностей OAuth 2.0 и ABAC позволит внедрить сквозную модель безопасности, которая способна защитить конфиденциальность клиентов и сотрудников, наиболее важные для бизнеса транзакции и наиболее конфиденциальные данные по API шлюзу. Также существует возможность фильтрования ответного сообщения. Это важно, если вызов API предназначен для извлечения записи данных из студенческой, банковской или медицинской карты, так как они могут содержать конфиденциальные или приватные элементы данных, которые следует отфильтровать в зависимости от полномочий вызывающего абонента.**

***Ключевые слова* – RBAC, ABAC, OAuth 2.0, Open Authorization, PDP, PEP, PIP, PAP.**

I. Введение

Авторизация и управление доступом являются одними из наиболее важных частей современных автоматизированных систем, поскольку они напрямую влияют на безопасность и контроль доступа к определенным частям системы для различных групп пользователей. Особенно это актуально в растущей тенденции нахождения уязвимостей, связанных с контролем доступа в отношении наиболее важных рисков безопасности веб-приложений [1].

В связи с ростом угрозы со стороны злонамеренных инсайдеров возникла необходимость принятия мер предосторожности для защиты информации [2]. Организации в значительной степени полагаются на свою информацию и технологии, которые ее защищают и используют. Из-за зависимости организаций от информации и технологий они становятся уязвимыми для киберугроз: поэтому менеджеры этих организаций должны принимать надлежащие меры предосторожности и внедрять соответствующие передовые методы для снижения риска инсайдерских угроз и других киберугроз [3]. Когда учетная запись пользователя создается в сети, последнее, что организация хочет сделать, это предоставить этому пользователю привилегии для всей сети. Сотрудники не должны иметь полного доступа, особенно если им нужна только базовая учетная запись для выполнения определенной задачи. Предоставление пользователям привилегий для всей сети может иметь печальные последствия, поскольку недовольные сотрудники могут явиться источником инсайдерских угроз и использовать плохие политики безопасности, чтобы обойти все уровни безопасности, существующие в организации [4]. Влияние извне на сотрудников организации также может иметь место, кроме того, акцент делается на восприимчивых людях внутри организаций, а недовольные сотрудники являются легкой мишенью [5]. Развертывание строгой политики безопасности может быть упущенным из виду процессом, и часто его не рассматривают, пока не становится слишком поздно.

II. Модели управления доступом

Управление доступом — это процесс, использующий механизмы предоставления полномочий на доступ к определенным ресурсам, приложениям или системе. Архитекторы и администраторы компьютерной безопасности развертывают механизмы контроля доступа (Access Control Mechanisms – ACM) в соответствии с требованиями к защите своих объектов при обработке запросов субъектов. ACM могут использовать различные методы для обеспечения соблюдения политики управления доступом, применяемой к объектам. То, как функционируют эти системы управления доступом, можно описать в терминах различных логических моделей контроля доступа. Эти модели контроля доступа обеспечивают структуру и набор условий, на основе которых объекты, субъекты, операции и правила могут быть объединены для выработки и обеспечения выполнения решения об управлении доступом. Каждая модель имеет свои преимущества и ограничения, в настоящее время наиболее популярными являются две основные модели управления доступом: модель управления доступом на

Статья получена 10 мая 2023.

А.В. Беловодов – МГУ имени М.В. Ломоносова (email: belovodoff@gmail.com)

О.Р. Лапонина – МГУ имени М.В. Ломоносова (email: laponina@oit.cmc.msu.ru)

основе ролей (RBAC) и модель управления доступом на основе атрибутов (ABAC).

RBAC является наиболее предпочтительной моделью управления доступом для локального домена. В этом случае права доступа пользователя определяются ролями, заданными для данного пользователя.

Роли – это не что иное, как абстракции поведения пользователей и назначенных им обязанностей. Для обеспечения контроля доступа и безопасности в конкретных программных системах выгодно использовать концепцию ролей. Она также снижает затраты на управление полномочиями [6].

Можно выделить следующие недостатки модели управления доступом на основе ролей:

– Требует дополнительных усилий для поддержания существующих ролей пользователей;

– В больших системах может происходить стремительное увеличение количества ролей в системе;

– Не может поддерживать ограничения, связанные с учетом текущих параметров окружения.

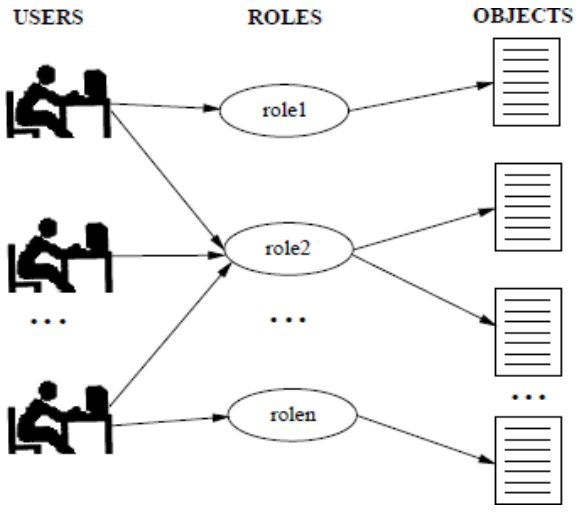

Рисунок 1. Архитектура RBAC

ABAC – это модель авторизации следующего поколения, которая обеспечивает динамическое, контекстно-зависимое управление доступом. В ABAC разрешения на доступ к объектам не предоставляются субъекту напрямую, для этого используются три ключевых элемента:

- Атрибуты;
- Политики;
- Архитектура развертывания.

Атрибуты — это основа для ABAC. Иначе говоря, атрибуты — это пары ключ-значение, где ключ представляет идентификатор атрибута. Атрибуты могут иметь несколько значений. Примеры атрибутов:

- Время обращения;
- Место обращение;
- Идентификатор обратившегося.

Следует заметить, что атрибутная модель может содержать в себе ролевую модель.

Одних атрибутов недостаточно для выражения логики авторизации. Они должны быть связаны друг с другом с помощью политик. Ключевым отличием ABAC является возможность задать сложный логический набор политик, с помощью которого будет оцениваться множество различных атрибутов [7]. Эти политики являются прямым отражением требований к авторизации, предъявляемых владельцами приложений. Политики естественного языка (Natural language processing – NLP) – это требования высокого уровня, которые определяют, как управляется доступ к информации и кто, при каких обстоятельствах, может получить доступ к какой информации. NLP выражены в понятных для человека терминах и могут быть недоступны для прямой реализации в ACM. NLP одновременно могут быть специфичны для конкретного приложения и, следовательно, должны учитываться поставщиком приложения, и NLP также описывать возможные действия субъекта в контексте корпоративных политик. Например, NLP могут описывать использование объектов внутри или между подразделениями организации или могут основываться на факторах необходимости знать, компетентности, полномочий, обязательств или конфликта интересов. Такие политики могут охватывать несколько вычислительных платформ и приложений.

NLP требуются для кодификации в алгоритмы или механизмы цифровой политики (Digital Policy – DP). Для повышения эффективности работы и простоты спецификации может потребоваться декомпозиция NLP и перевод его в различные версии DP, соответствующие инфраструктуре операционных подразделений предприятия.

Затем для этих различных версий DP могут потребоваться метаполитики (Meta Policy – MP), или политики, определяющие использование нескольких DP и управление ими для обработки иерархических полномочий нескольких DP, устранения конфликтов нескольких DP, а также хранения и обновлений DP. Таким образом, MP используются для управления DP. В зависимости от уровня сложности могут потребоваться иерархические структуры для выбора стратегий приоритета и комбинирования определенных NLP.

Наконец, рассмотрим распределение нескольких ACM и управление ими. В зависимости от потребностей пользователей, размера предприятия, распределения ресурсов и уровней конфиденциальности объектов, к которым необходимо получить доступ или совместно использовать их, использование нескольких ACM может иметь решающее значение для успеха внедрения ABAC. Функциональные компоненты ACM могут быть физически и логически разделены и распределены внутри предприятия.

Внутри ACM существует несколько функциональных «точек», которые являются основой для поиска политики и управления ею, наряду с некоторыми логическими компонентами для обработки контекста или процесса поиска политики, различных атрибутов и оценки результатов. Основные функциональные точки: точка применения политики (Policy Enforcement Point – PEP), точка принятия политических решений (Policy Decision Point – PDP), точка информации о политике (Policy Information Point – PIP) и точка администрирования политики (Policy Administration

Point – PAP). Когда эти компоненты находятся в едином окружении, они должны функционировать согласованно, чтобы принимать решения по управлению доступом.

PDP выполняет оценку DP и нескольких MP для принятия решения об управлении доступом.

Следующая функция, которую необходимо выполнить в рамках этих компонентов, заключается в обеспечении соблюдения этих решений, принятых PDP. Эту роль выполняет PEP.

PDP и PEP могут быть физически и логически разделены на предприятии. Например, предприятие может создать централизованно управляемую корпоративную службу принятия решений, которая оценивает атрибуты и политику и выносит решения, которые затем передаются в PEP в качестве утверждений. Это позволяет осуществлять централизованное управление атрибутами субъекта и политикой, но предоставляет частичный контроль доступа к объекту со стороны владельца локального объекта. Для проектирования и распространения компонентов ACM требуется функция управления, обеспечивающая координацию возможностей ABAC. Для того чтобы PDP и PEP могли выполнять свои роли, они должны иметь возможность располагать информацией об атрибутах и политиках, которые должны быть применены. Эти функции выполняет PIP.

Прежде чем эти правила могут быть применены, они должны быть тщательно протестированы и оценены, чтобы убедиться, что они отвечают предполагаемым потребностям. Это действие осуществляется PAP.

Алгоритм функционирования модели:

Когда от субъекта отправляется запрос на доступ для выполнения определенного действия над объектом, он перехватывается PEP, который преобразует запрос приложения в запрос авторизации и отправляет его в PDP, являющийся механизмом принятия решений об авторизации, который использует информацию, предоставленную в запросе, и политики, чтобы решить, должен ли запрос быть разрешен или нет. PDP использует PIP для поиска атрибутов, на которые ссылаются политики и, следовательно, необходимы для принятия решение по запросу на авторизацию. Точка администрирования политики (PAP), как следует из названия, является архитектурным объектом, который используется для управления политиками, которые позже оценивает PDP. Это позволяет создавать, развертывать и управлять изменениями политик.

Подробное рассмотрение данной модели контроля доступа необходимо для понимания работы модели, предложенной далее.

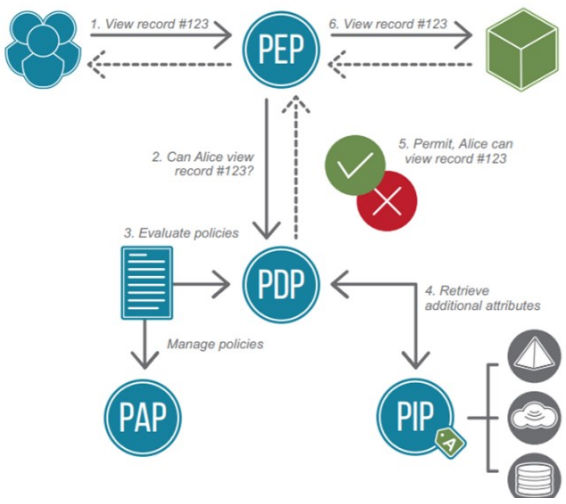

Рисунок 2. Архитектура ABAC

Традиционно администрирование безопасности больших систем упрощалось с помощью подхода к управлению доступом на основе ролей, который масштабируется лучше, чем другие предыдущие модели. Рассматривая роли и идентификационные данные как характеристики участника, управление доступом на основе атрибутов полностью охватывает функциональность как ABAC, так и RBAC и может определять разрешения на основе практически любых характеристик, относящихся к безопасности. Более гибкое применение политики может быть достигнуто с помощью гибких служб контроля доступа к данным. В связи с этим некоторые подходы использовали модели, объединенные в некоторых случаях со второй версией протокола OAuth 2.0.

III. Авторизация в веб-приложениях, протокол OAuth 2.0

Сегодня все больше и больше конфиденциальных транзакций и данных доступно через API-интерфейсы, ставшими фактически наиболее часто используемым способом, позволяющим клиентам, сотрудникам, деловым партнерам и службам подключаться к внутренним процессам, сервисам и данным.

API-шлюз – это инструмент управления, который находится между клиентом и набором внутренних сервисов и часто является важным компонентом архитектуры защиты API. Базовых возможностей безопасности шлюза может быть достаточно для простых или базовых вариантов использования, но организациям, которым необходимо безопасно предоставлять доступ к особо важным данным и обмениваться ими, скорее всего, понадобятся дополнительные возможности. Большинство шлюзов API предоставляют возможности интеграции с решениями безопасности, поддерживающими различные стандартизированные механизмы контроля доступа, такие как OAuth, для проверки удостоверений, управления токенами и поддержки других сценариев.

OAuth 2.0 – это открытый протокол авторизации, позволяющий выдать одному сервису (приложению) права на доступ к ресурсам пользователя на другом

сервисе. Протокол избавляет от необходимости передавать приложению логин и пароль, а также позволяет выдавать ограниченный набор прав, а не все сразу [7], предоставляя сервису токен доступа.

OAuth определяет четыре роли:

- Владелец ресурса
- Клиент
- Сервер ресурсов
- Авторизационный сервер

Авторизационный сервер проверяет подлинность информации, предоставленной клиентом, а затем создаёт авторизационные токены для приложения, с помощью которых клиент будет осуществлять доступ к серверу ресурсов.

Необходимые термины для рассмотрения протокола:

Авторизационный грант – креденциал, выданным собственником ресурса для доступа к защищенным ресурсам, и используется клиентом для получения токена доступа.

Токен доступа – креденциал, используемый клиентом для доступа к защищенным ресурсам. Токен доступа является строкой, представляющей собой авторизацию, выпущенную для клиента. Строка обычно рассматривается клиентом как неформатированные данные. Токены предназначены для определенной области и определенного способа доступа, выданные собственником ресурса и используются (учитываются) сервером ресурса и сервером авторизации.

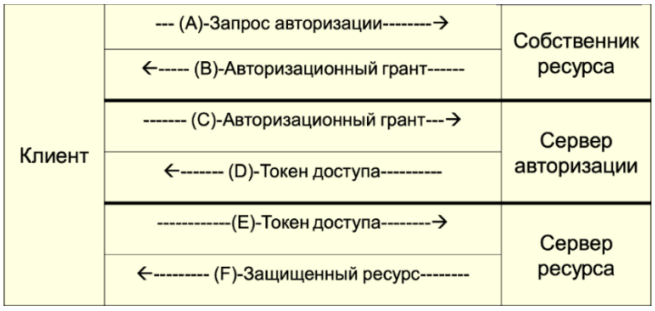

Рисунок 3. Абстрактное представление потока данных протокола

Протокол OAuth 2.0 обычно используется для авторизации третьей стороной, чтобы получить разрешение на доступ к серверу ресурса. OAuth 2.0 использует механизм области действия для ограничения доступа к источнику информации из приложения [8]. Затем область действия будет помещена на сгенерированный токен, когда аутентификация будет выполнена, с этого момента источник будет проверять только токен и встроенную в него область действия, чтобы проверить, разрешено ли пользователю получать необходимую исходную информацию [9].

IV. Модель управления доступом на основе атрибутов в протоколе OAuth 2.0

Чтобы обеспечить защиту ресурсов тех, кто может выполнять определенные транзакции или, кто может получать доступ к определенным данным и использовать их по API-интерфейсу, решение API-шлюзов должно быть дополнено и расширено решением для авторизации на основе динамической политики, организации смогут обеспечить контроль доступа к конкретным ресурсам. Это означает, что контроль доступа, например, может быть применен к отдельным документам, банковским счетам (личным, общим или делегированным), журналам пациентов с согласия пациента или без него, или к страховой претензии, с которыми может работать только назначенный или обученный персонал.

A. ABAC как микросервис

Рассмотрим службу авторизации ABAC как микросервис или набор микросервисов. Это обеспечит совместимость архитектуры и развертывания с приложениями на основе микросервисов. Сервис ABAC обладает типичными характеристиками микросервиса:

- Не поддерживает какое-либо состояние, обрабатывает запросы без сохранения какой-либо информации о состоянии
- Является неизменяемым
- Имеет четко определенный интерфейс
- Поддерживает REST/JSON
- Имеет ограниченный контекст
- Является отказоустойчивым
- Удобна для контейнеров

Комбинирование возможностей OAuth 2.0 и ABAC позволит внедрить сквозную модель безопасности, которая способна защитить конфиденциальность клиентов и сотрудников, наиболее важные для бизнеса транзакции и наиболее конфиденциальные данные по API шлюзу.

Однако области OAuth довольно статичны и не поддерживают какой-либо язык для выражения политики авторизации. Написание логики политики авторизации с использованием OAuth приведет к тому, что организации будут встраивать логику авторизации в API. Это создает тесную связь между логикой API и логикой авторизации, что затрудняет их управление и аудит. Таким образом, области OAuth в основном подходят для детального и функционального контроля доступа, например какие пользователи могут выполнять платежные транзакции или просматривать журналы пациентов.

Типичная конфигурация развертывания состоит из сервера авторизации OAuth 2.0 и шлюза API, играющего роль сервера ресурсов. Будем предполагать, что эта функция отделена от бизнес-службы. Можно думать об этом как о втором и третьем этапах потока предоставления кода авторизации, когда клиенту выдается токен доступа, который используется при последующих вызовах бизнес-службы. Шлюз API проверяет маркер доступа перед пересылкой запроса в бизнес-службу. Сам по себе OAuth не обеспечивает хорошего механизма для обеспечения соблюдения политики для бизнес-сервисов, содержащих конфиденциальные данные, что является распространенным сценарием в отрасли.

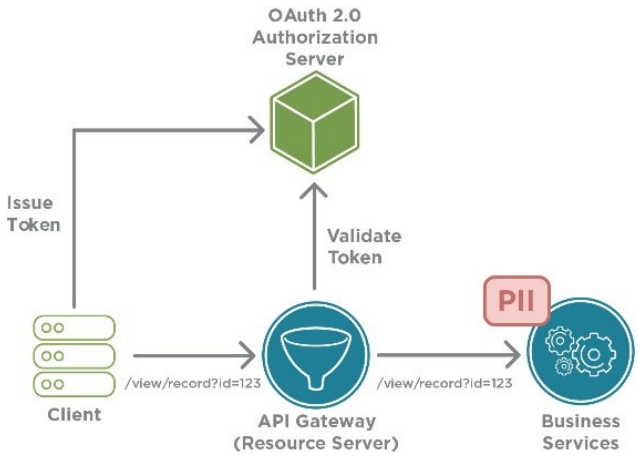

Рисунок 4. Шлюз API взаимодействует с OAuth 2.0

Шлюз API играет ключевую роль в развертывании API/микросервисов, предоставляя множество возможностей безопасности, управления и эксплуатации. Когда должны быть выполнены расширенные требования к контролю доступа, шлюз можно легко настроить для вызова службы ABAC, чтобы определить, следует ли предоставлять доступ к API или отказывать в нем, как показано на рисунке 4. Все политики доступа централизованно управляются и применяются в службе ABAC вместо жесткого кодирования этой логики в шлюзе – или, что еще хуже, в самом API. Может существовать несколько бизнес-служб, защищенных одним шлюзом API, и применяемая политика может быть динамической и применяться ко многим / всем из них.

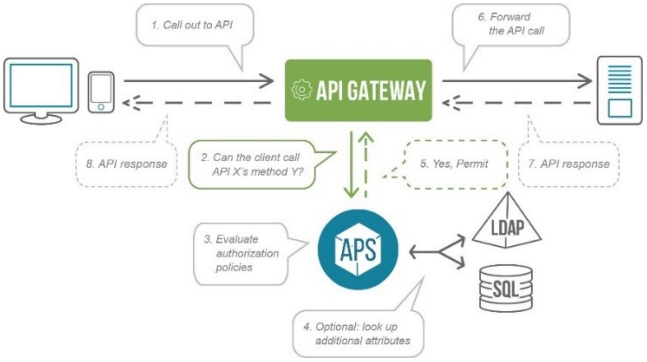

Рисунок 5. Архитектура ABAC со шлюзом API в качестве точки принудительного исполнения

Шлюз API можно настроить для вызова службы авторизации в сценариях, где требуется дополнительная оценка политики. Для этих сценариев шлюз создает форматированное сообщение JSON с соответствующими данными (SubjectID, метод, уровень аутентификации, запрошенный ресурс и т.д.) и отправляет его в конечную точку REST службы авторизации. Во время оценки политики служба авторизации может включать дополнительную контекстуальную информацию, такую как определение связи SubjectID с запрашиваемым ресурсом (если таковой имеется), проверка текущего статуса клиента (бронзовый, серебряный, золотой), оценка текущего риска и так далее. Полученное решение отправляется обратно на шлюз, где оно анализируется и приводится в исполнение. Для этого уровня интеграции не требуется никакого пользовательского кодирования, только настройка шлюза.

```
{
 "Request": {
  "AccessSubject": {
   "Attribute": [ { "AttributeId":
          "urn:oasis:names:tc:xacml:1.0:subject:subject-id",
          "Value": "%request.queryparam.token#" } ]
  },
  "Resource": {
   "Attribute": [
       { "AttributeId":"api.path","Value":"%message.path#"},
         {                                          "AttributeId":
      "urn:oasis:names:tc:xacml:1.0:resource:resource-id",
      "Value": "%message.path#" } ]
  },
  "Action": { "Attribute": [ { "AttributeId":
      "urn:oasis:names:tc:xacml:1.0:action:action-id",
      "Value": "%message.verb#" } ]
  }
 }
```

Листинг 1. Пример JSON запроса

B. Шлюз API вызывает службу ABAC, используя информацию OAuth

Все компоненты ABAC и OAuth объединяются в этом варианте конфигурации. На рисунке 6 шлюз API имеет в своем распоряжении много информации непосредственно перед выполнением вызова API от имени клиента. Например, шлюз имеет информацию об области действия и, возможно, используется конечная точка OAuth userinfo для сбора дополнительных данных о вошедшем в систему пользователе в веб-токене JSON (JWT). Атрибуты о пользователе вместе с информацией о вызываемом API упаковываются в сообщение для службы ABAC для оценки в точке принудительного исполнения (PEP). Токен JWT также может быть перенаправлен в точку принятия политических решений (PDP). Далее PDP использует точку информации о политике (PIP) для поиска атрибутов, на которые ссылаются политики и точку администрирования политики (PAP) для поиска необходимых политик, на основании которых принимает решение об удовлетворении запроса.

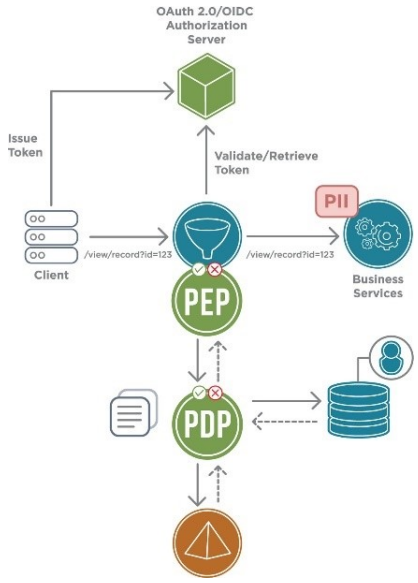

Рисунок 6. Шлюз API опосредует потоки данных ABAC – OAuth

```
{
 "alg": "HS256",
 "typ": "JWT"
}
{
 "uid": "Alice",
 "clearance": "top secret",
 "role": "manager",
 "department": "Sales"
}
HMACSHA256(
 base64UrlEncode(header) + "."
+
 base64UrlEncode(payload),
 my-secret
)
```

Листинг 2. Пример JWT токена

Как отмечалось ранее, служба ABAC может получить доступ к любому дополнительному контексту (оценка риска, информация об устройстве и.т.д.), чтобы принять решение об авторизации перед отправкой результата обратно на шлюз для принудительного исполнения.

C. Фильтрование сообщений на этапе ответа API

Безопасность API – это не односторонний процесс. Все предыдущие сценарии сосредоточены на применении аутентификации и контроля доступа к входящему вызову API. Следует также предусмотреть возможность фильтрования ответного сообщения. Если вызов API предназначен для извлечения записи данных из студенческой, банковской или медицинской карты, то они могут содержать конфиденциальные или приватные элементы данных, которые следует отфильтровать в зависимости от полномочий вызывающего абонента. Следовательно, необходимо иметь возможность настраивать шлюз API для вызова службы ABAC для определения любого фильтрования полей, которое должно применяться перед возвратом записи вызывающему пользователю или приложению. В этом случае шлюз отправляет метаданные (такие как регион, флаг PII и т.д.) из записи данных в службу ABAC для принятия решения о возможности пересылки. Служба ABAC возвращает набор решений или шаблон фильтрования, который шлюз применяет к записи данных. Этот сценарий проиллюстрирован на рисунке 7.

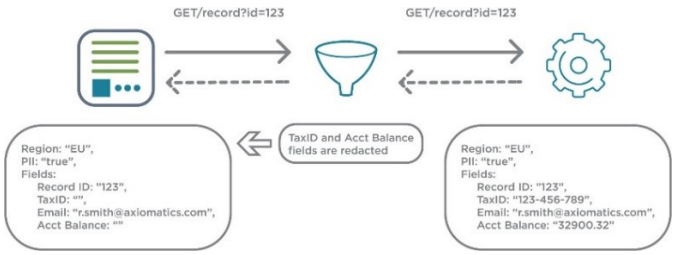

Рисунок 7. Фильтрация полей на этапе ответа API

V. Заключение

Авторизация и управление доступом являются одними из наиболее важных частей современных автоматизированных систем, поскольку они напрямую влияют на безопасность и контроль доступа к определенным частям системы для различных групп пользователей. Особенно это актуально в растущей тенденции нахождения уязвимостей, связанных с контролем доступа в отношении наиболее важных рисков безопасности веб-приложений.

В данной статье были рассмотрены возможностей применения управления доступом на основе ролей (RBAC) и управления доступом на основе атрибутов (ABAC) совместно с использованием открытого протокола авторизации OAuth 2.0. Рассмотрены основные понятия, связанные с моделями авторизации, атрибутами, и проблемами, с которыми сталкиваются существующие способы авторизации, а также возможные методы их решения.

В статье предложена модель, предназначенная для управления доступом на основе атрибутов для кросс-доменных источников с использованием API. Модель включает базовые архитектурные решения и принципы ABAC и OAuth, содержит все необходимые инструменты и инфраструктуру, необходимые для решения задач аутентификации, авторизации и контроля доступа на основе атрибутов.

Библиография

[1] OWASP Top Ten, “Top 10 Web Application Security Riskss”, https://owasp.org/www-project-top-ten/, accessed at 2021.

[2] Клейкомб, У., Хут, К. Л., Флинн, Л., Макинтайр, Д. М., & Левеллен, Т. (2012). Хронологический анализ саботажа с инсайдерской угрозой: предварительные наблюдения. J. Wirel. Mob. Сети Вездесущего Компьютера. Надежное приложение, 3, 4-20.

[3] Аль-Ахмад, У. (2013). Подробная стратегия управления безопасностью корпорации в кибервойне. Международный журнал кибербезопасности и цифровой криминалистики, 2 (4), 1-9.

[4] Солуаде, О. А., и Опара, Э. У. (2014). Нарушения безопасности, сетевые эксплойты и уязвимости: загадка и анализ. Международный журнал кибербезопасности и цифровой криминалистики, 3 (4), 246-261.

[5] Брикли, Дж. К., Тхакур, К., Камруззаман, А. С. (2021). Сравнительный анализ технических и нетехнических средств защиты от фишинга. Международный журнал кибербезопасности и цифровой криминалистики, 10 (1), 28-41.5. В. В. Лэмпсон. Защита. ACM SIGOPS Operating System Review, 8(1):18-24, январь 1974.

[6] Х. Л. Ф. Рави С. Сандху, Эдвард Дж. Койн и К. Э. Юман. Модели управления доступом на основе ролей. IEEE Компьютер, 29(2): 38-47, февраль 1996 года.

[7] Х. Джин, Р. Кришнан и Р. С. Сандху. Унифицированная модель управления доступом на основе атрибутов, охватывающая DAC, MAC и RBAC. DBSec, 12:41-55, 2012.

[8] D. Hardt, Ed, “The OAuth 2.0 Authorization Framework”, IETF, 2012.

[9] Bilbie, A., “A Guide To OAuth 2.0 Grants”, https://alexbilbie.com/guide-to-oauth-2-grants/, accessed at July 2019.

Using attribute-based access control in OAuth 2.0

A.V. Belovodov, O.R. Laponina

Abstract - **This article is a research paper on the possibilities of applying role-based access control (RBAC) and attribute-based access control (ABAC) together using the OAuth 2.0 open authorization protocol. The article discusses the basic concepts associated with authorization models, attributes, and how existing solutions face a number of problems in the modern world, as well as possible methods for solving them. The article proposes a model for attribute-based access control for cross-domain origins using an API. The model includes basic architectural solutions and principles of ABAC and OAuth. The ABAC authorization service is considered as a microservice or a set of microservices. This will ensure architecture and deployment compatibility with microservice-based applications. Combining the capabilities of OAuth 2.0 and ABAC will enable an end-to-end security model that can protect customer and employee privacy, business-critical transactions, and most sensitive data across the API gateway. It is also possible to filter the response message. This is important if the API call is to retrieve a record of data from a student, bank, or medical card, as these may contain sensitive or private data elements that should be filtered based on the caller's credentials.**

Keywords – **RBAC, ABAC, OAuth 2.0, Open Authorization, PDP, PEP, PIP, PAP.**

REFERENCES

[1] OWASP Top Ten, “Top 10 Web Application Security Riskss”, https://owasp.org/www-project-top-ten/, accessed at 2021.

[2] Klejkomb, U., Hut, K. L., Flinn, L., Makintajr, D. M., & Levellen, T. (2012). Hronologicheskij analiz sabotazha s insajderskoj ugrozoj: predvaritel'nye nabljudenija. J. Wirel. Mob. Seti Vezdesushhego Komp'jutera. Nadezhnoe prilozhenie, 3, 4-20.

[3] Al'-Ahmad, U. (2013). Podrobnaja strategija upravlenija bezopasnost'ju korporacii v kibervojne. Mezhdunarodnyj zhurnal kiberbezopasnosti i cifrovoj kriminalistiki, 2 (4), 1-9.

[4] Soluade, O. A., i Opara, Je. U. (2014). Narushenija bezopasnosti, setevye jeksplojty i ujazvimosti: zagadka i analiz. Mezhdunarodnyj zhurnal kiberbezopasnosti i cifrovoj kriminalistiki, 3 (4), 246-261.

[5] Brikli, Dzh. K., Thakur, K., Kamruzzaman, A. S. (2021). Sravnitel'nyj analiz tehnicheskih i netehnicheskih sredstv zashhity ot fishinga. Mezhdunarodnyj zhurnal kiberbezopasnosti i cifrovoj kriminalistiki, 10 (1), 28-41.5. B. V. Ljempson. Zashhita. ACM SIGOPS Operating System Review, 8(1):18-24, janvar' 1974.

[6] H. L. F. Ravi S. Sandhu, Jedvard Dzh. Kojn i K. Je. Juman. Modeli upravlenija dostupom na osnove rolej. IEEE Komp'juter, 29(2): 38-47, fevral' 1996 goda.

[7] H. Dzhin, R. Krishnan i R. S. Sandhu. Unificirovannaja model' upravlenija dostupom na osnove atributov, ohvatyvajushhaja DAC, MAC i RBAC. DBSec, 12:41-55, 2012.

[8] D. Hardt, Ed, “The OAuth 2.0 Authorization Framework”, IETF, 2012.

[9] Bilbie, A., “A Guide To OAuth 2.0 Grants”, https://alexbilbie.com/guide-to-oauth-2-grants/, accessed at July 2019.