Лекция 5: Модели управления доступом

Тема: Управление доступом и управление идентификацией

Технический университет Молдовы

Лектор: Максим Масютин

Введение

Здравствуйте. Сегодня мы погружаемся в одну из наиболее фундаментальных концепций информационной безопасности — модели управления доступом. Это основа всего, что мы делаем в сфере безопасности, и я не могу не подчеркнуть, насколько важно глубоко понимать эти концепции.

Начну с простого вопроса: что произойдёт, если каждый сотрудник организации получит доступ ко всему? Хаос, верно? Представьте, что любой сотрудник может читать данные о зарплатах, изменять финансовые записи или удалять критически важные системные файлы. Результаты были бы катастрофическими — как с точки зрения безопасности, так и с операционной точки зрения.

Управление доступом — это механизм, с помощью которого мы регулируем, кто или что может просматривать, использовать или изменять ресурсы в вычислительной среде. Это привратник наших цифровых активов.

Согласно отчёту Verizon Data Breach Investigations Report за 2025 год, более 74% всех нарушений безопасности были связаны с человеческим фактором, включая атаки с применением социальной инженерии, ошибки и злоупотребления. Многих из них можно было бы избежать или смягчить их последствия при наличии надлежащих механизмов управления доступом.

Рассматривайте управление доступом как четырёхэтапный процесс, который мы подробно изучим сегодня. Во-первых, нам нужно знать, кто вы — это идентификация. Во-вторых, нам нужно подтвердить, что вы действительно тот, за кого себя выдаёте — это аутентификация. В-третьих, нам нужно определить, что вам разрешено делать — это авторизация. И в-четвёртых, нам нужно отследить, что вы фактически сделали — это подотчётность.

Эти четыре концепции формируют основу фреймворка IAAA (первые буквы английских слов Identification, Authentication, Authorization, Accountability — что переводится как идентификация, аутентификация, авторизация, подотчётность).

Этот фреймворк управляет доступом практически в каждой защищённой системе.

Часть 1: Управление идентификацией и доступом (IAM) и структура IAAA

Что такое Управление идентификацией и доступом (IAM)?

Управление идентификацией и доступом, или IAM (Identity and Access Management), — это комплекс бизнес-процессов, политик и технологий, облегчающий управление электронными или цифровыми идентификационными данными (цифровыми личностями). При наличии внедренной системы IAM менеджеры по информационным технологиям (ИТ) могут контролировать доступ пользователей к критически важной информации внутри своей организации.

По своей сути, IAM отвечает на три фундаментальных вопроса для каждого запроса доступа:

- **Кто** является действительным пользователем? (Управление идентификацией)
- **Что** им разрешено делать? (Управление доступом)
- **Когда** и **Как** они это делают? (Аудит и отчетность)

Системы IAM предоставляют администраторам инструменты и технологии для изменения роли пользователя, отслеживания действий пользователей, создания отчетов об этих действиях и постоянного обеспечения соблюдения политик. Это дисциплина, которая позволяет нужным людям получать доступ к нужным ресурсам в нужное время и по нужным причинам.

Структура IAAA: Идентификация, Аутентификация, Авторизация и Подотчетность

Идентификация: кто вы?

Идентификация — это процесс заявления о своей личности. Когда вы подходите к стойке охраны и говорите: "Я Иван Иванов из отдела ИТ", вы идентифицируете себя. В цифровом мире это обычно принимает форму имени пользователя, адреса электронной почты, номера сотрудника или другого уникального идентификатора.

Ключевая характеристика идентификации заключается в том, что это заявление, а не доказательство. Любой может заявить, что он — кто-то другой. Я мог бы войти в здание и заявить, что я генеральный директор, но это не сделает это правдой.

Именно поэтому одной только идентификации никогда не достаточно для предоставления доступа к конфиденциальным ресурсам.

С технической точки зрения идентификаторы должны быть уникальными в пределах области действия системы. Невозможно иметь двух пользователей с одинаковым именем в одном домене, точно так же как невозможно иметь двух сотрудников с одинаковым номером в одной организации. Эта уникальность критически важна для подотчётности — мы должны иметь возможность отслеживать действия вплоть до конкретных лиц.

Современные системы управления идентификацией часто используют множественные идентификаторы. У пользователя может быть имя для входа в систему, адрес электронной почты для переписки, номер сотрудника для кадровых систем и уникальное имя в службе каталогов. Все они указывают на одного и того же человека, но служат разным целям.

Аутентификация: докажите это

Аутентификация — это процесс проверки заявленной личности. Это момент, когда мы переходим от "я заявляю, что я Иван Иванов" к "вот доказательство, что я Иван Иванов". Факторы аутентификации делятся на три традиционные категории, к которым в последние годы добавились ещё две.

Первая категория — то, что вы знаете

Сюда входят пароли, ПИН-коды, секретные вопросы и парольные фразы. Это самая старая и наиболее распространённая форма аутентификации. Древние римляне использовали пароли для идентификации дружественных сил, и мы до сих пор используем ту же базовую концепцию. Однако факторы знания имеют существенные слабости. Их можно забыть, угадать, украсть или выманить у человека.

Вторая категория — то, что вы имеете

Сюда входят физические токены, смарт-карты, мобильные телефоны, аппаратные ключи безопасности или любые другие физические устройства. Идея в том, что даже если кто-то знает ваш пароль, он не сможет пройти аутентификацию без физического токена. Удалённая кража такого фактора значительно сложнее, хотя физическая кража, разумеется, остаётся возможной.

Третья категория — то, что вы есть, то есть биометрия

Отпечатки пальцев, распознавание лица, сканирование радужной оболочки, голосовые паттерны и даже поведенческая биометрия, такая как характер набора текста — всё это — биоме́трия. Она удобна тем, что вы всегда носите её с собой и не можете забыть. Однако она сопряжена с собственными проблемами — её нельзя

изменить в случае компрометации, и она порождает вопросы о конфиденциальности.

Четвёртая категория, получившая признание относительно недавно, — место, где вы находитесь

Это — аутентификация на основе местоположения. Ваши GPS-координаты, IP-адрес или близость к определённой сети могут служить факторами аутентификации. Это становится всё более важным в архитектурах с нулевым доверием.

Пятая категория — то, что вы делаете, то есть поведенческая биометрия

Это биометрия, анализирующая паттерны вашего взаимодействия с системами. Сюда входят динамика нажатия клавиш, паттерны движения мыши и жесты на сенсорном экране. Эти факторы часто используются как механизмы непрерывной аутентификации, а не как разовая проверка.

Авторизация: что вы можете делать?

Как только мы узнали, кто вы, и подтвердили вашу личность, нам нужно определить, что вам разрешено делать. Это авторизация — процесс определения прав доступа и привилегий, предоставляемых субъекту.

Решения об авторизации основываются на политиках. Эти политики могут быть простыми — "администраторы имеют доступ к панели администрирования" — или чрезвычайно сложными, включающими множество условий, временных ограничений и контекстных факторов. Ключевой момент состоит в том, что авторизация всегда должна следовать принципу минимальных привилегий, который мы подробно обсудим позже.

Авторизацию часто путают с аутентификацией, но это фундаментально различные понятия. Аутентификация подтверждает личность; авторизация предоставляет разрешения. Вы можете быть полностью аутентифицированы — система точно знает, кто вы, — но при этом вам может быть отказано в доступе к ресурсу, потому что вы не авторизованы для его использования.

Современные системы авторизации часто используют подходы на основе утверждений (claims). При аутентификации вы получаете токен, содержащий утверждения о вашей личности — ваше имя, отдел, роль, членство в группах. Система авторизации затем оценивает эти утверждения в соответствии со своими политиками для определения того, какой доступ предоставить.

Подотчётность: что вы сделали?

Подотчётность — это четвёртый и завершающий столп фреймворка IAAA. Она обеспечивает запись всех действий, выполняемых аутентифицированными и авторизованными пользователями, и возможность их отслеживания до

конкретных лиц. Без подотчётности у нас нет способа обнаружить злоупотребления, расследовать инциденты или обеспечить соблюдение нормативных требований.

Подотчётность реализуется через журналирование, аудит и мониторинг. Каждая попытка доступа — успешная или неуспешная — должна регистрироваться с указанием того, кто выполнил действие, к какому ресурсу был осуществлён доступ, когда это произошло и откуда. Эти журналы аудита служат множеству целей: сдерживают несанкционированное поведение, поддерживают расследование инцидентов, удовлетворяют нормативные требования и предоставляют доказательства для дисциплинарных или судебных разбирательств.

Рассмотрим сценарий, в котором администратор базы данных, имеющий легитимный доступ, решает экспортировать и продать клиентские данные. Без механизмов подотчётности записи об этом действии не было бы, и оно могло бы остаться необнаруженным. При наличии надлежащего журналирования и мониторинга, необычный экспорт данных был бы отмечен, расследован и отслежен до ответственного лица.

Подотчётность также напрямую связана с идентификацией — именно поэтому уникальные идентификаторы так важны. Если несколько человек совместно используют одну учётную запись, подотчётность разрушается, поскольку мы не можем определить, кто именно выполнил то или иное действие. Это один из самых сильных аргументов против совместного использования учётных записей в корпоративных средах.

Современные системы подотчётности выходят за рамки простого журналирования. Они включают мониторинг в реальном времени с автоматическими оповещениями, поведенческую аналитику, способную обнаруживать аномальные паттерны, и системы SIEM (Security Information and Event Management, управление информацией и событиями безопасности), которые коррелируют события из множества источников для выявления угроз.

Аббревиатура AAA в сетевых протоколах

В сетевой и специальной литературе по безопасности вы часто будете встречать аббревиатуру AAA — Authentication, Authorization, Accounting (аутентификация, авторизация, учёт). По сути, это тот же фреймворк IAAA, но без начального этапа идентификации. В модели AAA фаза идентификации считается неявной — к тому моменту, когда пользователь отправляет учётные данные сетевому устройству, он уже себя идентифицировал, поэтому фреймворк сосредоточен на трёх оставшихся этапах.

Наиболее известные протоколы, реализующие AAA, — это RADIUS (от английского Remote Authentication Dial-In User Service, что переводится как Служба удалённой аутентификации пользователей по коммутируемым линиям) и

TACACS+ (от английского Terminal Access Controller Access-Control System Plus, что переводится как Система управления доступом контроллера доступа к терминалу). Протокол RADIUS, описанный в RFC 2865 и RFC 2866, изначально был разработан для удалённого коммутируемого доступа к сетям в 1990-х годах, но эволюционировал в широко используемый протокол управления сетевым доступом. Он объединяет аутентификацию и авторизацию в один обмен и обрабатывает учёт отдельно. RADIUS является основой аутентификации в корпоративных сетях Wi-Fi (802.1X), VPN-доступа и управления абонентами интернет-провайдеров.

TACACS+, разработанный компанией Cisco, разделяет аутентификацию, авторизацию и учёт на три независимые операции. Такое разделение обеспечивает более детальный контроль — например, можно аутентифицировать сетевого администратора один раз, но авторизовать каждую выполняемую им команду отдельно. TACACS+ также шифрует всю полезную нагрузку пакета, тогда как RADIUS шифрует только поле пароля. Это делает TACACS+ предпочтительным выбором для управления административным доступом к сетевой инфраструктуре: маршрутизаторам, коммутаторам и межсетевым экранам.

Diameter — преемник RADIUS (название является игрой слов: diameter, то есть диаметр, вдвое больше radius, то есть радиуса) — был разработан для современных сетевых сред и устраняет многие ограничения RADIUS. Он поддерживает надёжную передачу данных по TCP или SCTP ("Stream Control Transmission Protocol" — протоко́л управле́ния переда́чей пото́ков). Diameter имеет встроенные механизмы отказоустойчивости и широко применяется в мобильных сетях четвёртого и пятого поколений для аутентификации абонентов и управления политиками.

Поэтому, когда вы видите аббревиатуру AAA в контексте сетевой безопасности, помните, что она означает аутентификацию, авторизацию и учёт — те же концепции, что и в IAAA, но с подразумеваемой идентификацией и подотчётностью, сфокусированной именно на учёте использования ресурсов: продолжительности сессий, объёме переданных данных и временных метках подключений.

Часть 2: Дискреционное управление доступом (DAC)

Теперь давайте рассмотрим основные модели управления доступом, начиная с дискреционного управления доступом, или DAC (Discretionary Access Control). Это модель, с которой вы, скорее всего, уже знакомы, даже если не знали её название.

Дискреционный означает «предоставленный на собственное усмотрение», «необязательный» или «произвольный».

В модели DAC владелец ресурса определяет, кто может к нему обращаться. Когда вы создаёте файл на своём компьютере, вы становитесь его владельцем и решаете, кто ещё может его читать, изменять или выполнять. Эта модель используется в большинстве потребительских операционных систем, включая Windows, macOS и Linux.

Определяющая характеристика DAC — дискреционность: владелец обладает полным контролем над решениями о доступе. Нет центрального органа, предписывающего, кто и к чему должен иметь доступ. Если я создаю документ, я могу поделиться им с Алисой, запретить доступ Бобу и передумать завтра.

DAC обычно реализуется с помощью списков управления доступом, по английски: Access Control List (или ACL).

ACL прикрепляется к каждому ресурсу и указывает, какие субъекты могут выполнять какие операции. Например, ACL файл может указывать, что Алиса имеет права на чтение и запись, Боб имеет права только на чтение, а все остальные не имеют доступа.

Преимущества DAC — гибкость и простота. Пользователям не нужно проходить бюрократические процедуры для обмена файлами с коллегами. Системному администратору не нужно утверждать каждое решение о доступе. Это делает DAC весьма практичным инструментом для совместных рабочих сред.

Однако DAC имеет существенные ограничения в области безопасности. Поскольку владельцы обладают полным контролем, они могут принимать неверные решения — умышленно или случайно. Пользователь может поделиться конфиденциальными данными с неавторизованными лицами — как злонамеренно, так и просто потому, что не понимает степени конфиденциальности информации.

Ещё одна серьёзная слабость DAC — уязвимость перед вредоносным ПО. Если пользователь запускает вредоносный код, этот код выполняется с правами пользователя и может получить доступ ко всему, к чему имеет доступ пользователь. Если этот пользователь является администратором с широкими правами доступа, вредоносное ПО может нанести огромный ущерб.

DAC также страдает от проблемы "избыточного доступа". Со временем пользователи склонны накапливать разрешения. Они получают доступ к ресурсам для проекта, проект завершается, но доступ остаётся. Это накопление ненужных разрешений называется "накопление привилегий" и является серьёзной проблемой безопасности в средах DAC.

Несмотря на эти ограничения, DAC остаётся чрезвычайно популярным благодаря удобству использования. Ключ в том, чтобы сочетать DAC с другими средствами контроля — сильной аутентификацией, мониторингом, обучением пользователей и дополнительными мандатными контролями там, где это уместно.

Часть 3: Мандатное управление доступом (MAC)

Мандатное управление доступом представляет собой противоположную DAC философию. В модели MAC (Mandatory Access Control) центральный орган — обычно операционная система или администратор безопасности — определяет права доступа, и отдельные пользователи не могут отменить эти решения.

MAC основано на метках. Каждому субъекту (пользователю, процессу) и каждому объекту (файлу, ресурсу) присваивается метка безопасности. Решения о доступе принимаются путём сравнения этих меток в соответствии с фиксированным набором правил, которые не могут быть изменены обычными пользователями.

Наиболее известная модель MAC — модель Белла-Лападулы, разработанная Дэвидом Беллом и Леонардом Лападулой, исследователями компьютерной безопасности, в 1970-х годах для Министерства обороны США. Модель Белла-Лападулы ориентирована на конфиденциальность и реализует два ключевых правила. Первое — "нет чтения вверх": субъект не может читать объекты с более высоким уровнем безопасности. Второе — "нет записи вниз": субъект не может записывать данные в объекты с более низким уровнем безопасности.

Приведу пример. Представьте систему с тремя уровнями безопасности: несекретно, секретно и совершенно секретно. Пользователь с допуском "секретно" может читать несекретные и секретные документы, но не совершенно секретные. Тот же пользователь может записывать данные в секретные и совершенно секретные документы, но не в несекретные. Это предотвращает перетекание информации с высоких уровней безопасности на низкие.

Модель Биба, предложенная Кеннетом Биба, исследователем компьютерной безопасности, — это ещё одна модель MAC, но она ориентирована на целостность, а не на конфиденциальность. Модель Биба по существу инвертирует модель Белла-Лападулы: "нет чтения вниз" и "нет записи вверх". Это предотвращает загрязнение высокоцелостных ресурсов информацией с низким уровнем целостности. Модель Биба полезна в средах, где критически важна точность данных, например в финансовых системах.

MAC является мандатным в том смысле, что правила применяются независимо от желания владельца ресурса. Даже если я владею совершенно секретным документом и хочу поделиться им с пользователем, имеющим несекретный допуск, система не позволит этого. Это обеспечивает сильную защиту как от внутренних угроз, так и от скомпрометированных учётных записей.

Основные реализации MAC встречаются в специализированных защищённых операционных системах, таких как SELinux (Security-Enhanced Linux), разработанная Агентством национальной безопасности, или NSA (National Security Agency), и операционная система Trusted Solaris. Windows также

реализует форму MAC через механизм обязательного контроля целостности (Mandatory Integrity Control).

Недостаток MAC — его жёсткость и административная нагрузка. Каждый ресурс должен быть помечен. Каждому пользователю должен быть назначен уровень допуска. Политики должны быть тщательно спроектированы и поддерживаться. Это делает MAC непрактичным для большинства коммерческих сред, поэтому он используется преимущественно в военных и государственных контекстах, где требования безопасности оправдывают эти накладные расходы.

Часть 4: Управление доступом на основе ролей (RBAC)

Управление доступом на основе ролей появилось в 1990-х годах как практический компромисс между гибкостью DAC и строгостью MAC. RBAC (Role-Based Access Control) стал доминирующей моделью управления доступом в корпоративных средах.

Основная концепция RBAC проста: вместо того чтобы назначать разрешения непосредственно пользователям, вы назначаете разрешения ролям, а затем назначаете роли пользователям. Роль представляет собой должностную функцию внутри организации — "бухгалтер", "системный администратор", "руководитель проекта" и так далее.

Такое опосредование даёт несколько важных преимуществ. Во-первых, оно упрощает администрирование. Когда новый сотрудник поступает в бухгалтерию, вам не нужно определять все индивидуальные разрешения, которые ему необходимы. Вы просто назначаете ему роль "бухгалтер", и он автоматически получает все разрешения, связанные с этой ролью.

Во-вторых, RBAC обеспечивает единообразие. Все бухгалтеры имеют одинаковый базовый набор разрешений, что означает меньший риск того, что у одного бухгалтера будет избыточный доступ из-за исторического накопления разрешений.

В-третьих, RBAC облегчает соответствие требованиям. Аудиторы могут изучить определения ролей, чтобы понять, какие разрешения имеют различные должностные функции. Это значительно проще, чем пытаться анализировать индивидуальные разрешения сотен или тысяч пользователей.

Стандарт RBAC, определённый в модели RBAC от NIST, включает несколько компонентов. Базовый RBAC включает пользователей, роли, разрешения и сессии. Иерархический RBAC добавляет наследование ролей, когда старшие роли могут наследовать разрешения от младших ролей. RBAC с ограничениями добавляет ограничения на разделение обязанностей, о которых мы поговорим позже.

Приведу практический пример. Рассмотрим приложение электронной коммерции с тремя ролями: покупатель, сотрудник и администратор. Покупатели могут просматривать товары и размещать заказы. Сотрудники наследуют все разрешения покупателей и дополнительно могут обрабатывать заказы и управлять складскими запасами. Администраторы наследуют все разрешения сотрудников и дополнительно могут управлять учётными записями пользователей и конфигурацией системы.

RBAC имеет и ограничения. В крупных организациях с множеством ролей модель может стать сложной — проблема, иногда называемая "взрывом ролей". Когда имеются тысячи ролей с незначительными различиями, преимущества простоты RBAC начинают снижаться. Кроме того, RBAC относительно статичен — он плохо приспособлен к динамическим или контекстным решениям о доступе.

Часть 5: Управление доступом на основе атрибутов (ABAC)

Управление доступом на основе атрибутов, или ABAC (Attribute-Based Access Control), представляет собой следующую ступень эволюции управления доступом. ABAC принимает решения на основе атрибутов, а не предопределённых ролей. Эти атрибуты могут описывать субъект, ресурс, действие и окружение.

Атрибуты субъекта могут включать отдел пользователя, должность, уровень допуска или любую другую характеристику. Атрибуты ресурса могут включать классификацию документа, владельца, дату создания или тип содержимого. Атрибуты действия описывают, что субъект хочет сделать — читать, записывать, удалять, утверждать. Атрибуты окружения фиксируют контекст — время суток, местоположение пользователя, используемое устройство.

Политики ABAC выражаются в виде правил, которые оценивают эти атрибуты. Например: "Разрешить доступ, если отдел пользователя совпадает с отделом документа И классификация документа не превышает уровня допуска пользователя И текущее время находится между 9:00 и 17:00".

Сила ABAC — в его гибкости. Вы можете выразить практически любую политику управления доступом в виде правила ABAC. Вы не ограничены предопределёнными ролями или метками безопасности. Если возникает новое бизнес-требование, вы можете создать новые атрибуты и правила для его реализации без перестройки всей системы управления доступом.

ABAC определяется стандартом XACML — eXtensible Access Control Markup Language (расширяемый язык разметки управления доступом). XACML предоставляет стандартный способ выражения политик ABAC, а также запроса и получения решений об авторизации. Хотя сам XACML достаточно многословен и

сложен, многие современные системы реализуют принципы ABAC с использованием более простых языков политик.

Сложность ABAC — его главная проблема. Политики ABAC могут быть трудны для написания, понимания и аудита. Вместе с гибкостью выражения любой политики приходит риск выражения некорректных политик. Системы ABAC требуют тщательного проектирования и всестороннего тестирования.

На практике многие организации используют гибридный подход. RBAC обеспечивает базовую структуру — роли определяют широкие категории доступа. ABAC обеспечивает уточнение — атрибуты могут дополнительно ограничивать или расширять доступ в рамках этих ролей. Такая комбинация позволяет использовать преимущества обеих моделей.

Часть 6: Управление доступом на основе отношений (ReBAC)

По мере того как системы становятся более социальными и взаимосвязанными, новую популярность приобретает модель управления доступом на основе отношений, или ReBAC (Relationship-Based Access Control). Наилучшим примером этой модели является система Zanzibar, разработанная Google для управления авторизацией миллиардов пользователей в таких продуктах, как Drive, YouTube и Photos.

В ReBAC доступ определяется отношениями между субъектами и объектами. Вместо проверок типа "Имеет ли Пользователь U Роль R?", ReBAC спрашивает "Связан ли Пользователь U с Объектом O?".

Отношения могут быть прямыми ("Пользователь U является *владельцем* Документа D") или косвенными/производными ("Пользователь U является *членом* Группы G, а Группа G является *зрителем* Папки F, а Документ D находится *внутри* Папки F"). ReBAC превосходно справляется с обходом этих графов отношений для принятия решений об авторизации.

Типичная политика ReBAC может выглядеть так: "Пользователь может просматривать документ, если он является владельцем, ИЛИ если он является редактором, ИЛИ если он является зрителем родительской папки". Это отличает её от RBAC (который плохо справляется с иерархией) и ABAC (который может испытывать трудности с глубоким обходом графов).

Ключевые технологии в этой области включают:

- **Google Zanzibar**: Документ, популяризировавший эту модель.
- **OpenFGA (Fine Grained Authorization)**: Реализация с открытым исходным кодом концепций Zanzibar.
- **Permify**: Ещё одно решение ReBAC с открытым исходным кодом.

ReBAC быстро становится стандартом для B2B SaaS-приложений, где клиенты имеют сложные организационные иерархии и нуждаются в возможностях детального обмена (например, "поделиться этим конкретным проектом с командой Инженерии").

Часть 7: Управление доступом на основе политик (PBAC)

Управление доступом на основе политик тесно связано с ABAC, но делает акцент на политике как на центральном организующем понятии. В модели PBAC (Policy-Based Access Control) решения о доступе управляются явными политиками, которые можно централизованно управлять, версионировать и подвергать аудиту.

Ключевая идея заключается в том, что политики являются первоклассными объектами системы. Они не встроены в код приложений и не разбросаны по конфигурационным файлам. Вместо этого ими управляют как дискретными объектами, которые могут создаваться, изменяться и удаляться через формальные процессы.

Современные системы PBAC часто используют Open Policy Agent, или OPA, который стал де-факто стандартом для применения политик в облачных средах. OPA использует специализированный язык политик Rego, который позволяет выражать сложные правила управления доступом в читаемом и тестируемом формате.

PBAC хорошо согласуется с современными практиками разработки программного обеспечения. Политики могут храниться в системе контроля версий, проверяться через запросы на слияние, автоматически тестироваться и развёртываться через конвейеры CI/CD. Подход "политика как код" привносит строгость программной инженерии в управление доступом.

PBAC особенно ценен в архитектурах микросервисов, где решения об управлении доступом необходимо принимать во многих различных точках. Вместо того чтобы реализовывать логику управления доступом отдельно в каждом микросервисе, вы можете вынести логику во внешний централизованный движок политик и заставить все сервисы обращаться к нему за решениями об авторизации.

Часть 8: Идентификация машин и рабочих нагрузок

Исторически мы фокусировались на идентификации людей. Но в современных облачных средах программное обеспечение создает больше запросов

идентификации, чем люди. Это область **Идентификации машин** (Machine Identity).

Когда контейнеру в кластере Kubernetes нужно обратиться к базе данных, он должен аутентифицироваться. Жесткое кодирование паролей (секретов) является риском безопасности. Вместо этого мы используем короткоживущие, автоматизированные идентификаторы.

Ключевые стандарты включают:

- **SPIFFE (Secure Production Identity Framework for Everyone)**: Стандарт для предоставления идентификаторов рабочим нагрузкам (программным компонентам).
- **SPIRE**: Реализация SPIFFE, которая выдает криптографически проверяемые документы идентификации (SVID) рабочим нагрузкам.
- **OAuth 2.0 Client Credentials Flow**: Часто используется для межсервисного взаимодействия.

Управление идентификаторами машин часто сложнее, чем идентификаторами людей, потому что их так много, они эфемерны (создаются и уничтожаются за секунды) и работают со скоростью машин.

Часть 9: Принцип минимальных привилегий

Теперь давайте обсудим один из важнейших принципов в области безопасности: принцип минимальных привилегий. Этот принцип гласит, что каждый субъект должен обладать только минимальными привилегиями, необходимыми для выполнения его законных функций.

Это кажется очевидным, но на практике реализовать это поразительно сложно. Существует естественная тенденция предоставлять больше доступа, чем необходимо — это проще, это позволяет избежать жалоб от пользователей, которые не могут выполнять свою работу, и это предусматривает будущие потребности. Но каждая ненужная привилегия — это ненужный риск.

Приведу статистические данные. Согласно отчёту Identity Defined Security Alliance за 2025 год, 80% организаций столкнулись с нарушением безопасности, связанным с идентификацией, за последние два года. В большинстве из них были задействованы избыточные привилегии — учётные записи, имевшие доступ к ресурсам, которые им были не нужны, и этот доступ был использован злоумышленниками.

Принцип минимальных привилегий применяется на каждом уровне системы. Пользователи должны иметь доступ только к тем файлам и приложениям, которые им необходимы. Приложения должны иметь доступ только к тем

системным ресурсам, которые им необходимы. Процессы должны выполняться с минимальными привилегиями, необходимыми для их функционирования.

Реализация минимальных привилегий требует понимания того, что действительно необходимо каждому субъекту. Это сложнее, чем кажется. Пользователи часто не знают точно, что им нужно, и склонны запрашивать дополнительный доступ "на всякий случай". Определение реальных потребностей в доступе часто требует тщательного анализа должностных функций и рабочих процессов.

Доступ, ограниченный по времени, — важный аспект минимальных привилегий. Доступ, предоставленный для проекта, должен быть отозван по завершении проекта. Административные привилегии, которые нужны время от времени, должны предоставляться временно, а не постоянно. Это иногда называют доступом "just-in-time" (точно в срок).

Принцип минимальных привилегий также распространяется на то, что я называю мышлением о "радиусе поражения". Когда что-то идёт не так — когда учётная запись скомпрометирована или пользователь совершает ошибку — какой ущерб может быть нанесён? Минимальные привилегии уменьшают этот радиус поражения, гарантируя, что ни одна учётная запись не обладает возможностью нанести катастрофический ущерб.

Реализация минимальных привилегий — это непрерывный процесс, а не разовое мероприятие. Потребности в доступе меняются со временем. Сотрудники берут на себя новые обязанности, проекты начинаются и завершаются, организационные структуры эволюционируют. Регулярные проверки доступа необходимы для обеспечения того, чтобы привилегии оставались надлежащими.

Часть 10: Разделение обязанностей

Разделение обязанностей, иногда называемое разграничением обязанностей, — ещё один фундаментальный принцип безопасности. Идея заключается в том, что определённые критические операции должны требовать участия нескольких лиц, чтобы ни один человек не мог выполнить операцию в одиночку.

Классический пример — финансовый контроль. Лицо, которое может авторизовать платёж, не должно быть тем же лицом, которое может его выполнить. Таким образом, для хищения средств растратчику пришлось бы вовлечь нескольких человек. Каждый участник обеспечивает контроль над действиями других.

В информационной безопасности разделение обязанностей реализуется через политики управления доступом, которые предотвращают опасные комбинации привилегий. Человек, который пишет код, не должен быть тем же человеком, который развёртывает код в продуктивной среде. Человек, который администрирует учётные записи пользователей, не должен быть тем, кто проверяет журналы доступа.

Существует два типа разделения обязанностей: статическое и динамическое. Статическое разделение означает, что конфликтующие роли никогда не назначаются одному и тому же лицу. Если вам назначена роль "разработчик", вам никогда не может быть назначена роль "ответственный за развёртывание в продуктивную среду". Ограничение является постоянным.

Динамическое разделение более гибко. Вам могут быть назначены обе роли, но вы не можете выполнять обе в рамках одной транзакции или временного периода. Например, вы можете разрабатывать код и развёртывать код, но не можете развёртывать код, который вы сами разработали. Кто-то другой должен развернуть ваш код, а вы должны развёртывать код другого человека.

Современные системы RBAC реализуют разделение обязанностей через ограничения. Взаимоисключающие роли — это роли, которые не могут быть назначены одному и тому же пользователю. Ограничения по мощности лимитируют количество пользователей, которые могут быть назначены на конфиденциальную роль. Ограничения по предпосылкам требуют, чтобы определённые роли были назначены прежде других.

Разделение обязанностей действительно добавляет сложность и может снизить эффективность. Если каждое действие требует нескольких утверждающих лиц, работа замедляется. Ключ в том, чтобы применять разделение обязанностей к действительно чувствительным операциям, позволяя рутинным задачам выполняться без лишних препятствий.

Часть 11: Управление доступом на практике

Позвольте обсудить, как эти концепции объединяются в современных системах. Сегодняшние корпоративные среды обычно реализуют несколько моделей управления доступом, работающих совместно.

На уровне инфраструктуры вы часто найдёте контроли в стиле MAC. Сегментация сети предотвращает взаимодействие серверов в одной зоне с серверами в другой зоне. Среды исполнения контейнеров обеспечивают изоляцию между рабочими нагрузками. Эти мандатные контроли обеспечивают основу безопасности независимо от того, как настроены приложения.

На уровне приложений доминирует RBAC. Корпоративные приложения, такие как SAP, Salesforce и Microsoft 365, используют управление доступом на основе ролей. Пользователям назначаются роли, а ролям предоставляются разрешения на функции и данные. Это обеспечивает управляемый способ контроля доступа для больших групп пользователей.

На уровне данных всё чаще применяется ABAC. Когда необходим детальный контроль над конкретными записями — этот пользователь может видеть эти записи, но не те, на основании различных атрибутов — ABAC обеспечивает необходимую выразительность.

Облачные среды имеют собственные механизмы управления доступом. AWS использует политики IAM, которые по сути являются ABAC с языком политик на основе JSON. Azure использует RBAC с определениями пользовательских ролей. Google Cloud использует IAM с условиями. Понимание этих платформо-специфичных реализаций является необходимым для обеспечения облачной безопасности.

Тенденция 2025-2026 годов — движение к унифицированному управлению политиками. Организации внедряют подходы "политика как код", при которых политики управления доступом для всех систем — облачной инфраструктуры, приложений, баз данных, API — управляются через единый фреймворк. Такие инструменты, как Open Policy Agent, HashiCorp Sentinel и облачные сервисы политик, способствуют этой конвергенции.

Часть 12: Проблемы управления доступом и лучшие практики

Позвольте завершить лекцию практическими рекомендациями по эффективному внедрению управления доступом.

Во-первых, знайте свои данные. Невозможно защитить данные, если вы не знаете, где они находятся и насколько они конфиденциальны. Классификация данных — это предпосылка эффективного управления доступом. Работайте с заинтересованными сторонами бизнеса, чтобы понять, какие данные наиболее критичны и наиболее конфиденциальны.

Во-вторых, тщательно проектируйте роли. В системах RBAC хорошо продуманные роли имеют решающее значение. Роли должны соответствовать должностным функциям, а не отдельным лицам. Избегайте создания слишком детальных ролей, что ведёт к взрыву ролей, или слишком широких, что нарушает принцип минимальных привилегий.

В-третьих, автоматизируйте предоставление и отзыв доступа. Ручное управление доступом не масштабируется и подвержено ошибкам. Внедряйте автоматизированные процессы для предоставления доступа при найме сотрудников или смене ролей и, что не менее важно, для отзыва доступа при увольнении или когда он более не нужен.

В-четвёртых, проводите проверки доступа. Как минимум ежегодно, а для конфиденциальных ресурсов чаще, проверяйте, кто имеет доступ к чему. Выявляйте и удаляйте ненужный доступ. Это часто называют "сертификацией доступа" или "аттестацией доступа".

В-пятых, отслеживайте паттерны доступа. Управление доступом — это не только предотвращение, но и обнаружение. Отслеживайте необычные паттерны доступа, которые могут указывать на скомпрометированные учётные записи или

внутренние угрозы. Современные решения управления идентификацией включают аналитические возможности для этих целей.

В-шестых, планируйте привилегированный доступ. Административные учётные записи и служебные учётные записи с повышенными привилегиями требуют особого внимания. Внедряйте решения для управления привилегированным доступом, обеспечивающие доступ "точно в срок", запись сессий и безопасное хранение учётных данных.

В-седьмых, тестируйте свои средства контроля. Политики управления доступом должны тестироваться как любой другой код. Проверяйте, что разрешения работают, как ожидается, и что запреты надлежащим образом применяются. Включайте тестирование управления доступом в программу оценки безопасности.

Заключение

Управление доступом — это основа информационной безопасности. Без эффективного управления доступом все прочие меры безопасности существенно ослабляются. Злоумышленник, получивший несанкционированный доступ, может обойти межсетевые экраны, преодолеть шифрование и избежать обнаружения.

Сегодня мы рассмотрели обширный материал — от базовых концепций идентификации, аутентификации и авторизации, через основные модели управления доступом DAC, MAC, RBAC, ABAC и PBAC, до принципов минимальных привилегий и разделения обязанностей. Эти концепции будут повторяться на протяжении всего нашего изучения безопасности, и вам потребуется применять их практически в любой роли, связанной с безопасностью.

В следующей лекции мы более глубоко погрузимся в механизмы аутентификации, рассмотрев многофакторную аутентификацию, современные беспарольные подходы и федеративную идентификацию. А пока я призываю вас изучить механизмы управления доступом в системах, которыми вы пользуетесь ежедневно. Посмотрите, как они реализуют роли, как обрабатывают разрешения и где могут иметь слабые места.

Вопросы для обсуждения

- Как организациям следует определять, какая модель управления доступом наиболее подходит для их среды?
- Какие проблемы возникают при реализации принципа минимальных привилегий в крупных организациях, и как их можно преодолеть?
- Как модели управления доступом могут потребовать эволюции для решения задач безопасности, связанных с удалённой работой и облачными

вычислениями?

- Чем идентификация машин отличается от идентификации людей с точки зрения жизненного цикла и управления?

Спасибо за внимание, увидимся в следующий раз.

Контрольные вопросы

- Объясните разницу между идентификацией, аутентификацией и авторизацией.
- Сравните и сопоставьте DAC и MAC. В каких ситуациях каждая из моделей была бы уместна?
- Как RBAC упрощает администрирование управления доступом по сравнению с DAC?
- Какие преимущества ABAC даёт по сравнению с RBAC? Какие у него проблемы?
- Что такое ReBAC, и какую проблему он решает, которую не может решить RBAC?
- Объясните принцип минимальных привилегий и приведите три примера его реализации.
- Что такое разделение обязанностей и почему оно важно для безопасности?
- Опишите разницу между статическим и динамическим разделением обязанностей.
- Как современные облачные платформы реализуют управление доступом? Приведите примеры из AWS, Azure или GCP.

Ключевые термины

- **Список управления доступом (ACL)**: список, указывающий, какие субъекты могут выполнять какие операции над ресурсом
- **ABAC**: управление доступом на основе атрибутов
- **Аутентификация**: процесс проверки заявленной личности
- **Авторизация**: процесс определения того, какие действия субъекту разрешено выполнять
- **Модель Белла-Лападулы**: модель MAC, ориентированная на конфиденциальность
- **Модель Биба**: модель MAC, ориентированная на целостность

- **DAC**: дискреционное управление доступом
- **Идентификация**: процесс заявления о своей личности
- **Идентификация машин**: идентификация, присвоенная нечеловеческой рабочей нагрузке (например, контейнеру, сервису)
- **MAC**: мандатное управление доступом
- **Необходимость знать**: принцип безопасности, ограничивающий доступ к информации на основании потребности в этой информации для выполнения конкретной задачи
- **PBAC**: управление доступом на основе политик
- **Политика как код**: управление политиками доступа как версионируемым кодом (например, OPA/Rego)
- **Принцип минимальных привилегий**: предоставление только минимально необходимого доступа
- **RBAC**: управление доступом на основе ролей
- **ReBAC**: управление доступом на основе отношений (например, Google Zanzibar)
- **Разделение обязанностей**: требование участия нескольких лиц для выполнения чувствительных операций
- **XACML**: eXtensible Access Control Markup Language (расширяемый язык разметки управления доступом)