Email ? ??????

Почта

- защита собственного домена от попыток отправки писем от вашего имени;
- защита собственных сотрудников от спама, вирусов и офисных файлов с опасными макросами;
- харденинг почтового сервера и сервиса обработки почты (exim, dovecot, postfix и т.п.).

Защита доменного имени (SPF, DKIM, DMARC)

SPF (Sender Policy Framework) — это TXT DNS запись, в которой следует указать IP-адреса и доверенных партнеров, которые авторизованы (имеют право) отправлять электронную почту из вашего домена, т.е. от "вашего имени". Принимающие почтовые сервера, могут сверить источник пришедшего письма (домен и IP) с соответствующей DNS записью в вашем домене, прежде чем перенаправить его в почтовый ящик получателя.

Пример SPF записи:

```
» dig txt ptsecurity.ru
...
ptsecurity.ru.        600    IN    TXT    "v=spf1 redirect=_spf.ptsecurity.com"
...
```

В записи участвует флаг redirect, который перенаправляет нас на сабдомен, давайте проверим его:

```
» dig txt _spf.ptsecurity.com
...
_spf.ptsecurity.com.    3600    IN    TXT    "v=spf1 ip4:178.238.126.136 ip4:178.238.126.137
ip4:195.133.251.200 ip4:195.133.251.201 ip4:31.44.93.58 ip4:81.27.243.31 ip4:81.27.243.54
ip4:195.133.251.208 ip4:178.238.126.134 mx -all"
...
```

v=spf1 — сообщает, что TXT запись содержит информацию, относящуюся непосредственно к SPF.

ipv4 (ipv6) — содержит список IP адресов и сетей, с которых дозволено отправлять письма.

mx — указывает на то, что серверам, указанным в MX DNS записи, так же разрешено отправлять письма. Определить MX запись можно так: dig mx ptsecurity.ru

-all сообщает серверу, что адреса, не указанные в записи SPF, не имеют права отправлять электронную почту и должны быть отклонены. Альтернативные варианты: ~all, который означает, что электронные письма, не включенные в список, будут помечены как небезопасные или спам, но все равно будут приняты, и, реже, +all, который означает, что любой сервер может отправлять электронные письма от имени вашего домена.

Еще один флаг, отсутствующий в примере include, который сообщает серверу, какие сторонние организации имеют право отправлять электронные письма от имени домена.

DKIM (DomainKeys Identified Mail) — это метод аутентификации вашего домена с помощью цифровых подписей. Соответственно, имеется две части ключа: открытая и закрытая. Закрытая часть храниться на сервере, в конфигурационных файлах вашего почтового сервиса (приложения). Открытая часть прописывается в DNS запись. Давайте попробуем ее найти. В одном из писем рекламной рассылки Positive Security можно найти вот такие строки:

```
DKIM-Signature: v=1;
 a=rsa-sha256;
 s=mindbox;
 d=email.ptsecurity.com;
 c=relaxed/relaxed;
 q=dns/txt;
 h=From:To:Subject;
 bh=x8CUMA+LBpmKFgUhm7dPgNlLQbe1ZHbqIOzByiP5Zzg=;
 b=E854evyPVTWgRO028yDTZEn2ZImhgJ6GHzsB4O25+MP7jsxqXk/9hs1vUDdEmyTBklxBgqzsi0V9EPz6vopgRQt8EU4
poAK4WBhYg8sHHna7jrPgLMMrV5q6gkIZgAVy4otrG2YHt2+vtb6R+CJrapbQvj69vj8E0J2kBA5kims=
```

v= — показывает, какая версия DKIM используется.
d= — доменное имя отправителя.
s= — это "селектор", который принимающий сервер должен использовать для поиска записи DNS.
h= — перечисляет поля заголовка, которые используются для создания цифровой подписи. В этом случае используются заголовки from, to и subject. Если бы Боб отправил электронное письмо Алисе, используя домен example.com, и в теме письма было бы «Рецепт каши», то здесь будет использовано следующее содержимое: "«bob@example.com» + «alice@example.com» + «Рецепт каши»".
bh= — это хеш тела письма.
a= — это алгоритм, используемый для вычисления цифровой подписи (b), а также для генерации хэша тела электронного письма (bh).
b= — цифровая подпись, сгенерированная из h и bh и подписанная закрытым ключом.

Для валидации сервер-получатель должен найти открытую часть ключа, для этого "селектор" и требуется:

```
» dig txt mindbox._domainkey.email.ptsecurity.com
...
mindbox._domainkey.email.ptsecurity.com. 3600 IN TXT "k=rsa;
p=MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDdyc8nf+xcoG7tOJUTXlysqykbPoshtsr1UGkG9D/yOAXlLicMP6kh
lj8B+34HDnpCMJEQgd/2PxlD47OHeo9Czx+1Lz6rX02CI4ocntmX41ysZO4Qk8FVGXZCWFwi64wtiUaw4zqYbAVb/oRxnS
fBAMatl2y+TcpKdNMl6IQeqQIDAQAB"
...
```

DMARC (Domain-based Message Authentication Reporting and Conformance) — это политика, сообщающая получающему почтовому серверу, что делать после проверки записей SPF и DKIM. Получатель на своей стороне может иметь различные спам правила, основанные на SPF и DKIM, отправитель не имеет над ними контроля. Политика DMARC же дает отправителю контролировать поведение спам фильтра на принимающей стороне.

```
» dig txt _dmarc.email.ptsecurity.com
...
_dmarc.email.ptsecurity.com. 3600 IN    TXT    "v=DMARC1; p=reject;"
...
```

v=DMARC1 — указывает, что эта запись TXT содержит политику DMARC и должна интерпретироваться как таковая серверами электронной почты.
p=reject — указывает, что почтовые серверы должны «блокировать» электронные письма, не прошедшие проверку DKIM и SPF, считая их потенциально спамом. Другие возможные настройки для этого включают p=none, который позволяет письмам, не прошедшим проверки быть доставленными, и p=quarantine, который предписывает серверам электронной почты отправлять письма в Спам.

Вы так же можете добавить параметр rua=mailto:example@third-party-example.com;, на указанный адрес будут приходить репорты о результатах работы вашей DMARC политики. Возможно, таким образом вы получите возможность вычислить, если ваше доменное имя используется в нелегетимных рассылках.

Когда вами сконфигурированы все три параметра — ваш домен защищен. Шанс успешного использования вашего доброго имени для спам-рассылок становится крайне низок. Только если получатель писем готов быть обманутым и никак не проверяет входящую почту.

Защита входящей почты от спама и вирусов

OpenSource проект Rspamd. Он будет проверять SPF, DKIM и DMARC

Интересный функционал проверки на то, как хорошо настроен ваш домен и сервер предоставляет портал https://www.mail-tester.com/. Примерно таким же образом работает и Rspamd — на каждое отклонение от общепринятых правил выставляет баллы. При достижении лимита письмо блокируется.

Кроме фильтрации спам сообщений, у Rspamd есть интересные интеграции, которые так же могут влиять на рейтинг передаваемого сообщения, проставляя флаг VIRUS_FOUND=2000:

olefy — анализирует содержимое макросов;
ClamAV — сигнатурная проверка вложенных файлов на вирусы.
Базы данных ClamAV по умолчанию не имеют высокого уровня обнаружения, но их можно улучшить с помощью бесплатных или платных баз данных сигнатур SecurityInfo.
Потребуется регистрация, после которой вы сможете использовать их с одного IP адреса.

OpenSource проект MailCow. Это докеризированное приложение, которое уже включает в себя все необходимые компоненты как для работы самой почты, так и для осуществления ее защиты.

Hardening Postfix

При отправке, сообщение уходит в компонент почтового сервера, обрабатывающий именно исходящие сообщения.

Postfix — это распространенный программный компонент на серверах для отправки электронной почты. Он имеет множество опций конфигурации, в том числе для повышения собственной безопасности. Мы разберем его безопасную настройку в качестве примера одного из ключевых компонентов.

Для начала, давайте разберем, какие угрозы существуют:

Использование вашего SMTP сервера для рассылки спам сообщений. Такой мисконфиг известен как Open Relay. Если вы его допустили, то ваш IP быстро попадет в различные репутационные списки и ваши собственные пользователи будут испытывать трудности с доставкой почты.

Поиск существующих пользователей Open relay

Интересно, что "опасности" у этой уязвимости на самом деле две: внешняя и внутренняя. Внешнюю можно проверить на портале https://mxtoolbox.com/diagnostic.aspx. Но даже, если она у вас имеется, вас скорей всего сможет защитить Rspmad. Т.е. до Postfix'а присланное из вне сообщение даже не дойдет.

Особенность же внутренней уязвимости заключается в том, что спам фильтры могут игнорировать проверку сообщений, присланных из частной локальной сети (10.0.0.0/8, 192.168.0.0/16 и т.п.) и считать их доверенными. Это может игнорироваться спам фильтрами. В итоге любой пользователь в локальной сети, имеющий доступ до почтового сервера по протоколу SMTP (tcp/25), может от имени любого другого пользователя делать внутренние рассылки. Фишинг письма, разосланные таким образом имеют огромный шанс на успех.

Пример проверки уязвимости:

```
# nc -nv 10.20.30.40 25
Connection to 10.20.30.40 port 25 [tcp/*] succeeded!
220 ****************************
HELO gmail.com
250 mail.example.com
mail from:user1@example.com
250 2.1.0 Ok
rcpt to:user2@example.com
250 2.1.5 Ok
data
354 End data with <CR><LF>.<CR><LF>
123
test
.
250 2.0.0 Ok: queued as 4C069182B39
quit
221 2.0.0 Bye
```

Защититься от нее довольно просто, указав в параметре mynetworks только те сети, которые считаете доверенными. В общем случае, там должны быть указаны только loopback интерфейсы:

```
postconf -e mynetworks="127.0.0.0/8 [::ffff:127.0.0.0]/104 [::1]/128"
```

User enumiration
Команда VRFY является сокращением от «проверить» (verify). Ее можно использовать, чтобы проверить, действителен ли адрес электронной почты на почтовом сервере. Это отлично подходит для устранения неполадок, но также позволяет другим получать информацию о существовании учетной записи и использовать эту информацию для спам рассылок, брутфорса и т.п. Команда VRFY обычно не требуется для доставки между двумя почтовыми серверами и ее следует отключать.

Пример проверки уязвимости:

```
# nc -nv 10.10.99.20 25
Connection to 10.10.99.20 port 25 [tcp/*] succeeded!
220 ****************************
HELO gmail.com
250 mail.example.com
vrfy user1@example.com
502 5.5.1 VRFY command is disabled
```

Устранить такую уязвимость несложно:

```
postconf -e disable_vrfy_command=yes
```

Облачная инфраструктура

Облачные сервисы — это услуги, предоставляемые через интернет, которые позволяют пользователям получать доступ к компьютерным ресурсам по запросу.

Существует несколько уровней облачных сервисов:

- Инфраструктура как сервис (IaaS): виртуальные вычислительные ресурсы по запросу (ВМ, хранилище и сети). Пользователи могут создавать, масштабировать и управлять инфраструктурой через интерфейс управления или API.
- Платформа как сервис (PaaS): на этом уровне предоставляются инструменты разработки и выполнения приложений, такие как базы данных, веб-серверы и средства разработки. Пользователи могут создавать и развертывать приложения, не заботясь о сложностях управления инфраструктурой.
- Программное обеспечение как сервис (SaaS): здесь предоставляются готовые приложения, доступные через интернет. Это могут быть CRM-системы, управление проектами, офисные приложения и многие другие. Пользователи получают доступ к приложениям через веб-браузер или специальные клиенты, обычно без необходимости установки и обновления программного обеспечения.

Процесс работы облачных сервисов обычно выглядит следующим образом:

- Создание учетной записи для получения доступа.
- Выбор ресурсов.
- Масштабирование ресурсов

С точки зрения информационной безопасности в облачных сервисах наиболее важны такие аспекты, как обеспечение безопасной аутентификации пользователей и управление их доступом к ресурсам.

VK Cloud

Применяются следующие меры и практики информационной безопасности:

- Мониторинг и противодействие атакам: Security Operations Center (SOC VK) осуществляет мониторинг и анализ безопасности серверов VK Cloud с использованием системы SIEM (Security Information and Event Management). Применяются механизмы, такие как антифрод и отслеживание подозрительной активности.
- Проверки безопасности: внешние проверки не реже одного раза в год, включают в себя анализ внутренних нарушителей. Также проводятся собственные аудиты информационной безопасности и участие в программах Bug Bounty для выявления и

устранения уязвимостей.
- Применение принципов безопасной разработки: разработчики проходят обучение по информационной безопасности, интегрируют и автоматизируют инструменты безопасности на всех этапах жизненного цикла разработки и эксплуатации (DevSecOps), а также проходят архитектурное ревью и аудит безопасности каждого сервиса.
- Соответствие требованиям 152-ФЗ и PCI DSS: VK Cloud обеспечивает соответствие требованиям законодательства и стандартам безопасности, таким как требования 152-ФЗ и стандарт PCI DSS.
- Применение отраслевых практик: VK Cloud применяет лучшие отраслевые практики, такие как изоляция сегментов и сервисов с помощью файервола, разграничение доступа с использованием ролевой модели IAM и ограниченный доступ к платформе только для администраторов с обязательной аутентификацией.

Yandex Cloud

В Yandex Cloud можно выделить следующие меры и инструменты обеспечения безопасности:

- Key Management Service — управление ключами шифрования;
- DDoS Protection — защита от DDoS-атак;
- Certificate Manager — управление TLS-сертификатами;
- Lockbox — создание и хранение секретов;
- Identity and Access Management — идентификация и контроль доступа к облачным ресурсам;
- Audit Trails — сервис сбора и выгрузки аудитных логов;
- Smart Web Security — защита веб-приложений;
- SmartCaptcha — инструмент для верификации запросов.
- Помимо этого, SOC Яндекса обеспечивает защиту облака, также в Yandex Cloud существует сегментирование и изоляция ресурсов.

Слои изоляции

- Изоляция серверов Yandex Cloud — серверы Yandex Cloud находятся в отдельных изолированных зонах внутри дата-центра: здесь действуют особые правила физического доступа, а физическая сеть платформы изолирована по периметру межсетевым экраном.
- Логическая изоляция на уровне гипервизора — архитектура гипервизора и средства управления виртуальной средой обеспечивают изоляцию одной виртуальной машины (ВМ) от другой. В Yandex Cloud используется аппаратная виртуализация, реализованная при помощи набора команд Intel VT-x. Взаимодействие таких ВМ можно организовать только с помощью организации доступа L3 между ними, при этом не важно на одном или разных физических хостах они размещены.
- Логическая изоляция на уровне управляемых сервисов - изоляция на этом уровне зависит от специфики и архитектуры управляемого сервиса. Если решение, предоставляемое в виде управляемого сервиса, реализует мультитенантную* среду, то есть надежную и безопасную работу различных пользователей облака (в

том числе принадлежащих разным организациям) с единой инсталляцией решения, то изоляция в основном обеспечивается на уровне логики решения.
- *Multi-tenancy (режим коллективной аренды) — архитектура программного обеспечения, при которой единый экземпляр приложения, запущенного на сервере провайдера, одновременно работает с несколькими арендаторами (компаниями-клиентами).

Так реализовано объектное хранилище (Object Storage) в Yandex Cloud.

Изоляция управляющей сети провайдера от виртуальных сетей облачных пользователей - управляющая сеть условно делится на базовую (underlay) и наложенную (overlay). Underlay — это сегмент физической сети, реализующий связность между всеми физическими компонентами облачной платформы. Overlay — это набор виртуальных сетей, которые могут использоваться как провайдером (в служебных целях для работы сервисов платформы), так и конечными пользователями. Underlay в Yandex Cloud делится на два сегмента (VLAN): технологический, который используется для работы overlay, и служебный для передачи данных между аппаратными хостами (например, в случае живой миграции ВМ). Трафик в underlay и в служебные сегменты overlay строго контролируется ACL на Tor и граничных маршрутизаторах (border routers), аппаратными файрволлами на периметре физической сети, программными файрволами на хостах и security groups (встроенных межсетевых экранах на уровне виртуальной сети).

Изоляция трафика разных виртуальных сетей — трафик виртуальных сетей маркируется с помощью MPLS-меток и может быть обработан (передан) только виртуальной сущностью, подключенной к той же виртуальной сети.
Логическая изоляция на уровне учетных записей и прав доступа - управление ресурсами облака реализуется с помощью сервиса управления ресурсами и ролевой модели. Все ресурсы сосредоточены в логическом контейнере, который в Yandex Cloud называется организацией. Ресурсная модель позволяет назначать роли контейнерам различных уровней (организации, облаку, папке) или непосредственно ресурсам (например, ключу KMS). Сервис управления ресурсами позволяет предоставлять доступ только тем пользователям, которые числятся пользователями организации.

Разделение сущностей control plane и data plane — control plane управляет ресурсами сервиса через служебные ВМ, обеспечивая изоляцию с помощью учетных записей и их прав доступа. Data plane содержит ВМ с базами данных, обеспечивая отказоустойчивость. Компоненты control plane не имеют доступа к данным пользователей, они лишь управляют рабочими компонентами, которые обрабатывают данные.

Изоляция сервисных компонент инфраструктуры провайдера от пользовательских ресурсов — фактически служебные компоненты платформы могут быть следующих видов:
компоненты, реализующие внутренние сервисы, не доступные конечным пользователям;
компоненты, реализующие доступный пользователям control plane сервисов;
компоненты, обеспечивающие работу data plane слоя (например, ВМ, на которой размещена БД, предоставляемая клиенту в рамках управляемого сервиса). В зависимости от ситуации такие сервисные компоненты изолируются как на уровне физических хостов (размещаются

на хостах, где нет пользовательской нагрузки. Пример — основные машины сервиса KMS), так и на уровне виртуальных сетей.

Identity and Access Management

Процесс управления идентичностью и доступом пользователей к ресурсам в информационной системе. Он включает в себя управление учетными записями пользователей, группами пользователей, ролями и разрешениями.

- Учетные записи пользователей. IAM управляет созданием, удалением и управлением учетными записями пользователей в системе. Это включает в себя установку и изменение паролей, настройку двухфакторной аутентификации и т. д.
- Группы пользователей Пользователей можно объединять в группы с общими характеристиками или правами доступа. IAM позволяет управлять этими группами, что делает процесс управления доступом более эффективным.
- Роли определяют набор разрешений и привязываются к пользователям или группам. Они предоставляют набор прав доступа к ресурсам, который определяет, какие действия могут выполнять пользователи.
- Разрешения определяют, какие действия могут выполнить пользователи или роли для конкретных ресурсов. Например, разрешения могут включать чтение, запись, удаление и т. д.

IAM обеспечивает безопасность информационных систем, управляя доступом к ресурсам на основе принципа наименьших привилегий (Least Privilege Principle), что означает, что пользователи получают только те права, которые необходимы для выполнения их задач. Это помогает предотвратить несанкционированный доступ и минимизировать риски для безопасности.

Роли пользователей

В общем случае в облачных сервисах можно выделить следующие категории доступа:

Администраторский доступ (Admin Access)	Полный доступ ко всем аспектам и функциям облачного сервиса. Возможность управлять пользователями, ресурсами, настройками безопасности и другими административными задачами. Этот уровень доступа обычно предоставляется администраторам системы или IT-специалистам.
Пользовательский доступ (User Access)	Доступ к основным функциям и ресурсам облачного сервиса для выполнения своих задач. Обычно ограниченный доступ к административным функциям и настройкам. Этот уровень доступа предоставляется конечным пользователям для работы с приложениями или данными.

Доступ разработчика (Developer Access)	Доступ к инструментам разработки и API для создания, тестирования и развертывания приложений в облачном окружении. Возможность управлять приложениями и интеграциями с другими сервисами. Этот уровень доступа предоставляется разработчикам для создания и сопровождения приложений.
Ограниченный доступ (Restricted Access)	Ограниченный доступ к определенным ресурсам или функциям в облачном сервисе. Может включать доступ только для чтения, доступ к определенным файлам или приложениям, или другие ограничения. Этот уровень доступа часто используется для предоставления временного или контролируемого доступа.
Гостевой доступ (Guest Access)	Временный доступ для пользователей, которые не имеют постоянного аккаунта в облачном сервисе. Обычно предоставляется для совместной работы или обмена информацией с внешними сторонами. Может быть ограничен в функциональности и доступе к данным.

Примеры ролей пользователей в VK Cloud

В контексте VK Cloud важно упомянуть понятие Проект.
Проект — это структурная единица внутри облака, которая владеет ресурсами: виртуальными машинами, базами данных, кластерами Kubernetes и другими. При регистрации нового аккаунта в VK Cloud автоматически создается проект, в котором текущий пользователь зарегистрирован в роли владельца. Владелец проекта может создавать новые проекты и приглашать во все свои проекты пользователей, назначая им роли. Один и тот же пользователь может быть участником нескольких проектов и иметь в них в разные роли.

Роли для общего управления проектом

- Владелец проекта — пользователь с максимально широким набором разрешений. Владельцем становится пользователь, который создал проект, либо для которого проект был создан платформой при регистрации аккаунта. В проекте может быть только один владелец. Эту роль нельзя назначить или пригласить на нее.
- Суперадминистратор — пользователь, который имеет те же разрешения, что владелец, включая привязку карты и пополнение баланса. Суперадминистратор — единственная роль, кроме владельца, которой доступна активация сервисов в проекте.
- Администратор проекта — пользователь, который имеет полные разрешения на создание и редактирование объектов во всех сервисах. Администратор не может: активировать сервисы; пополнять баланс проекта (ему доступен только просмотр баланса); приглашать пользователей.
- Администратор пользователей (IAM) — роль, предназначенная для работы с участниками проекта на странице управления доступами. Администратор

пользователей (IAM) может приглашать участников в проект и удалять их из проекта, редактировать назначенные участникам роли. Сервисы и информация о балансе проекта этой роли недоступны.
- Администратор биллинга — роль, предназначенная для управления балансом проекта. Администратор биллинга может: привязать к проекту карту оплаты, если она еще не привязана; пополнить баланс проекта или настроить автопополнение. Сервисы и список участников проекта этой роли недоступны.
- Наблюдатель — пользователь, который имеет полные разрешения на просмотр информации в проекте, включая список участников, данные всех сервисов, баланс проекта и детализацию расходов. Наблюдатель не может создавать какие-либо объекты и не может ничего редактировать, кроме настроек своего аккаунта.

Специализированные роли

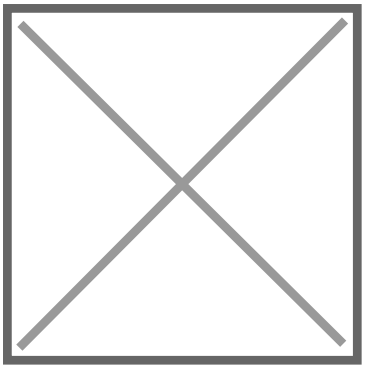
Каждая из ролей ниже предназначена для работы с одним из сервисов платформы. Этим ролям доступны: разрешения в их целевом сервисе; ряд разрешений в сопутствующих сервисах, без которых невозможна полноценная работа с целевым сервисом. У всех этих ролей отсутствует доступ к списку участников проекта и к информации о балансе. Все операции, доступные специализированным ролям, доступны также владельцу проекта, суперадминистратору и администратору проекта.

- Администратор виртуальных машин — пользователь с этой ролью может выполнять основные операции в сервисе Cloud Servers. При этом ему доступен только просмотр для: планов резервного копирования; файловых хранилищ. В сервисе виртуальных сетей он может создавать и редактировать группы правил (firewall).
- Администратор сети — пользователь с этой ролью может выполнять полный набор операций в сервисах виртуальных сетей и DNS.
- Администратор сетевой безопасности — пользователь с этой ролью может просматривать все данные в сервисах виртуальных сетей и DNS. Создавать и редактировать он может только группы правил (firewall).
- Администратор внутренних сетей — пользователь с этой ролью может: просматривать все данные в сервисах виртуальных сетей и DNS; создавать и редактировать виртуальные сети и подсети, маршрутизаторы; добавлять в проект плавающие IP.

Матрицу разрешений можно посмотреть по ссылке.

Роли в Yandex Cloud

В Yandex Cloud существует 4 примитивные роли: admin, editor, viewer и auditor. Примитивные роли в Yandex Cloud наследуют разрешения друг друга: например, в роль editor входят все разрешения роли viewer.



auditor

Роль auditor дает разрешения на чтение конфигурации и метаданных сервисов без возможности доступа к данным.

Роль auditor позволяет выполнять следующие операции:

- просмотр информации о ресурсе;
- просмотр метаданных ресурса;
- просмотр списка операций с ресурсом.

viewer

Роль viewer дает разрешения на чтение к ресурсам.

Роль viewer включает все разрешения, которые дает роль auditor. В отличие от роли auditor, роль viewer предоставляет возможность доступа к данным сервиса в режиме чтения.

Роль viewer позволяет выполнять следующие операции:

- просмотр информации о ресурсе;
- получение списка вложенных ресурсов, например списка виртуальных машин в каталоге;
- просмотр списка операций с ресурсом.

editor

Роль editor дает разрешения на все операции для управления ресурсом, кроме назначения ролей другим пользователям.

Роль editor включает все разрешения, которые дает роль viewer.

Помимо них, роль editor позволяет выполнять следующие операции:

- создание ресурса;
- обновление ресурса;
- удаление ресурса.

admin

Роль admin дает все разрешения для управления ресурсом, включая назначение ролей другим пользователям. Можно назначать любые роли за исключением resource-manager.clouds.owner (владелец ресурса)

Помимо ранее перечисленных операций, роль admin позволяет выполнять следующие операции:

- установить права доступа к ресурсу;
- изменить права доступа к ресурсу.

В зависимости от сервисов, роли могут немного варьироваться, но в целом они сводятся к ранее перечисленным примитивным ролям. Подробное описание ролей относительно различных сервисов можно посмотреть по ссылке.

Логирование в облачных сервисах

События в аудитных логах Yandex Cloud относятся к различным уровням:

- уровень Yandex Cloud — события, происходящие с ресурсами Yandex Cloud;
- уровень ОС;
- уровень приложений;
- уровень сети (Flow Logs).
- Уровень Yandex Cloud

Основным инструментом сбора логов уровня Yandex Cloud является сервис Yandex Audit Trails. Сервис позволяет собирать аудитные логи о происходящих с ресурсами Yandex Cloud событиях и загружать эти логи в бакет Yandex Object Storage или лог-группу Cloud Logging для дальнейшего анализа или экспорта.

Для сбора метрик, анализа некоторых событий уровня Yandex Cloud и настройки оповещений рекомендуется использовать сервис Yandex Monitoring. С его помощью возможно отслеживать, например, резкое возрастание нагрузки на Compute Cloud, RPS сервиса Application Load Balancer, значительные изменения в статистике событий сервиса Identity and Access Management. Кроме того, Monitoring можно применять для мониторинга работоспособности самого сервиса Audit Trails и мониторинга событий безопасности.

Формат записей аудитного лога универсален для всех событий, события представляют собой JSON-объекты. Значения некоторых полей определяются ресурсом-источником и типом события.

Яндекс предоставляет справочник событий. Аудитные логи возможно экспортировать в лог-группу Cloud Logging и в SIEM-систему клиента для анализа информации о событиях и инцидентах.

Экспорт событий в SIEM

Решения для экспорта аудитных логов Yandex Cloud подготовлены для следующих SIEM-систем:

- Yandex Managed Service for Elasticsearch (ELK);
- ArcSight;
- Splunk.

Для настройки экспорта в любые SIEM подходят утилиты GeeseFS или s3fs. Она позволяет смонтировать бакет Yandex Object Storage как локальный диск виртуальной машины. Далее на ВМ необходимо установить коннектор для SIEM и настроить вычитывание JSON-файлов из бакета.

Реагирование на события

С помощью Yandex Cloud Functions можно настроить оповещения о событиях Audit Trails, а так же автоматическое реагирование на вредоносные действия, например удаление опасных правил или прав доступа.

Уровень ОС
При использовании облачных сервисов по модели IaaS и использовании групп узлов Kubernetes клиент отвечает за безопасность ОС и выполняет сбор событий уровня ОС самостоятельно. Для сбора стандартных событий, которые генерирует ОС, и их экспорта в SIEM-систему клиента существуют бесплатные инструменты, такие как:

Osquery;
Filebeat (ELK);
Wazuh.
Дополнительные опции генерации событий возможно реализовать с помощью утилиты Auditd для Linux, Sysmon для Windows.

Системные метрики Linux (процессор, память, диск) можно собирать с помощью компонента Unified Agent сервиса Monitoring.

Также события ОС возможно экспортировать в Cloud Logging с помощью плагина Fluent bit

Для описания событий, которые нужно искать в логах, Яндекс рекомендует использовать формат Sigma, поддерживаемый популярными SIEM-системами. Репозиторий Sigma содержит библиотеку событий, описанных в этом формате.

Уровень приложений
Сбор событий уровня приложений, развернутых на ресурсах Compute Cloud, клиент может выполнять самостоятельно. Например, записывать логи приложения в файл и передавать их в SIEM-систему с помощью инструментов, перечисленных в подразделе Уровень ОС выше.

Уровень сети
Запись событий о сетевом трафике VPC (Flow Logs) на текущий момент может выполняться только средствами клиента. Для сбора и передачи событий могут использоваться решения

из Yandex Cloud Marketplace (например, NGFW, IDS/IPS, сетевые продукты) либо бесплатное ПО.

Управление доступом в Yandex Cloud

Все операции в Yandex Cloud предварительно отправляются на проверку в IAM:

- Пользователь просит сервис Compute Cloud создать новый диск в каталоге default.
- Сервис спрашивает IAM, можно ли этому пользователю создать диск в этом каталоге.
- IAM проверяет, что пользователь — участник облака с каталогом default и имеет необходимые разрешения для создания диска в этом каталоге.

Если какого-то из разрешений у пользователя нет, операция не будет выполнена, и Yandex Cloud сообщит об ошибке.
Если все разрешения имеются, то IAM сообщает об этом сервису.
Сервис создает новый диск.
Управление доступом в Yandex Cloud построено на политике Role Based Access Control (RBAC). Чтобы предоставить доступ к ресурсу, вы указываете, кому и какие роли назначены на ресурс.

Чтобы назначить роль, вы выбираете ресурс, выбираете роль и описываете субъект, которому назначается роль. Таким образом вы привязываете права доступа к ресурсу. Вы также можете назначить роль на родительский ресурс, от которого наследуются права доступа, например назначить роль на каталог или облако.

Ресурсы, на которые можно назначать роли
Назначать роли можно на облако, каталог и другие ресурсы из списка. Если нужно предоставить доступ к ресурсу, которого нет в списке, например к кластеру Yandex Managed Service for PostgreSQL, назначьте роль на родительский ресурс, от которого наследуются права доступа. У кластеров Managed Service for PostgreSQL права доступа наследуются от каталога.

Роль
Назначать роли на ресурс могут пользователи с ролью администратора на этот ресурс, а также владельцы облака, которому принадлежит ресурс.

Каждая роль состоит из набора разрешений, описывающих допустимые операции с ресурсом. Пользователь может назначить роли только с теми разрешениями, которые имеются у него самого. Например, чтобы назначить роль владельца облака, пользователь должен сам обладать этой ролью, а роли администратора для этого недостаточно.

Субъект, которому назначается роль
Роли назначаются субъектам. Существуют следующие типы субъектов:

- userAccount — аккаунт на Яндексе, добавленный в Yandex Cloud.
- serviceAccount — сервисный аккаунт, созданный в Yandex Cloud.

- Сервисному аккаунту можно назначать роли на любые ресурсы в любом облаке, если эти ресурсы относятся к той же организации, что и сервисный аккаунт. Также сервисному аккаунту можно назначать роли на саму организацию.
- federatedUser — аккаунт пользователя федерации удостоверений, например из Active Directory.
- group — группа пользователей, созданная в Yandex Cloud Organization.
- system — системная группа.

Наследование прав доступа
Если у ресурса есть дочерние ресурсы, то все разрешения от родительского ресурса будут унаследованы дочерними ресурсами. Например, если вы назначите пользователю роль на каталог, в котором лежит виртуальная машина, то все разрешения этой роли будут действовать и для виртуальной машины.

Если на дочерний ресурс тоже назначены роли, то список разрешений на этот ресурс будет объединен со списком разрешений на родительский ресурс. Нельзя ограничить список разрешений, унаследованных от родительского ресурса.

Имперсонация
Имперсонацией называется выполнение пользователем действий с ресурсами облака от имени сервисного аккаунта, которому назначены необходимые права. Имперсонация чаще всего применяется, чтобы временно расширить права пользователя, не прибегая к генерации статических учетных данных.

Например, когда у пользователя нет прав на просмотр каталога, но на какое-то время они ему оказались нужны. Для этого администратор может назначить сервисному аккаунту роль на просмотр каталога, а пользователю назначить специальную роль iam.serviceAccounts.tokenCreator. В результате пользователь сможет от имени сервисного аккаунта просматривать ресурсы в каталоге или получить IAM-токен сервисного аккаунта. Пользователь не сможет изменить права доступа или удалить сервисный аккаунт. В нужный момент администратор может отозвать роль.

Управление доступом в VK Cloud

С помощью ACL.

ACL (Access Control List) позволяет контролировать, какие операции разрешены каким пользователям. ACL может стоять как и на уровне всего бакета, так и на уровне конкретного объекта. Установить и прочесть ACL можно через приведенные методы ниже. По умолчанию, создаваемый бакет или объект максимально ограничен в доступе — только владелец бакета/объекта может работать с ним. У остальных пользователей — запрет доступа. ACL указывается в XML формате, где в поле Owner ID нужно указать свой канонический идентификатор в системе VK Cloud. Получить его можно разными способами. Один из способов:

aws s3api list-buckets --query Owner.ID --output text --endpoint-url https://hb.vkcs.cloud

Пример ACL, который дает те же права, что и по умолчанию (только владелец имеет полный доступ, никто больше):

```
<?xml version="1.0" encoding="UTF-8"?>
<AccessControlPolicy xmlns="http://<имя_бакета>.hb.vkcs.cloud/images/01.jpg/">
  <Owner>
    <ID>fcd68908-6c76-42d1-968b-82ae2a5a251d</ID>
    <DisplayName>owner-display-name</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xsi:type="Canonical User">
        <ID>fcd68908-6c76-42d1-968b-82ae2a5a251d</ID>
        <DisplayName>display-name</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

Элемент <Owner> производит идентификацию владельца по каноническому идентификатору пользователя учетной записи VK Cloud.
Элемент <Grant> определяет идентификатор получателя для предоставляет разрешение.
Элемент <Permission> внутри <Grant> определяет тип доступа для получателя.
Базовый ACL, показанный выше в качестве примера, имеет один элемент <Grant>. Чтобы описать несколько получателей, то для каждого надо добавить свой элемент <Grant>.

При установке через HTTP заголовки, нужно использовать заголовки для выдачи специфичных для заголовка прав:

x-amz-grant-read — список получателей прав для READ.
x-amz-grant-write — список получателей прав для WRITE.
x-amz-grant-read-acp — список получателей прав для READ_ACP.
x-amz-grant-write-acp — список получателей прав для WRITE_ACP.
x-amz-grant-full-control — список получателей прав для FULL_CONTROL.
x-amz-acl — использование шаблонного ACL.
Выдача прав
Идентификаторов для выдачи прав может быть несколько типов:

Конкретный пользователь в VK Cloud. Для этого нужно знать его уникальный идентификатор.

Проект в VK Cloud. Для этого нужно знать уникальный идентификатор проекта.
Предварительно определенные группы. Список указан ниже.
Весь мир. Включая анонимный доступ. Любой, знающий полный адрес до объекта, имеет доступ.
Выданное право для идентификатора выше может быть одним или составлено из нескольких типов:

READ — только чтение, какие-либо изменения не разрешены.
WRITE — только запись, какие-либо чтения не разрешены. Включая удаление.
READ_ACP — чтение ACL, какие-либо изменения не разрешены.
WRITE_ACP — запись ACL, какие-либо чтения не разрешены. Включая удаление.
FULL_CONTROL — все, что перечислено выше, сразу.
Выдача прав по идентификатору пользователя
Идентификатор пользователя (он же известен как канонический ID/Canonical ID) — это уникальный идентификатор, состоящий из набора символов. Пример: fcd68908-6c76-42d1-968b-82ae2a5a251d. Формат не фиксирован, при работе с идентификатором пользователя какие-либо преобразования и привязки к формату идентификатора не рекомендуются.

Если используется установка через XML, то внутри <Grantee> надо указать тег <ID>.

Пример: <ID>fcd68908-6c76-42d1-968b-82ae2a5a251d</ID>

Если используется установка через HTTP заголовок, то в значении заголовка надо использовать ключ id.

Пример: id="fcd68908-6c76-42d1-968b-82ae2a5a251d".

Канонический ID пользователя учетной записи VK Cloud можно получить, прочитав ACL бакета или объекта, к которому учетная запись VK Cloud имеет права доступа. Когда отдельная учетная запись VK Cloud получает разрешения по запросу на Grant, в ACL добавляется запись гранта с каноническим ID пользователя учетной записи VK Cloud.

Выдача прав по идентификатору проекта
Если идентификатор пользователя неизвестен, но известен идентификатор проекта, то можно указать проект. При обработке запроса на установку ACL, система ищет идентификатор пользователя и сохраняет его в ACL. В результате ACL всегда будут содержать канонический ID пользователя для проекта, а не идентификатор проекта.

Если используется установка через XML, то внутри <Grantee> надо указать тег <EmailAddress>.

Пример: <EmailAddress>mcs2400549523</EmailAddress>

Если используется установка через HTTP заголовок, то в значении заголовка надо использовать ключ id.

Пример: emailAddress="mcs2400549523".

Получить свой идентификатор проекта можно в личном кабинете в области информации об учетной записи. Кнопка, расположенная рядом с идентификатором проекта, позволяет скопировать параметр для удобства.

Виды разрешений
В таблице перечислены наборы разрешений, которые Cloud Storage поддерживает в ACL. Набор разрешений ACL одинаков для ACL объекта и ACL бакета. Эти ACL предоставляют разрешения для определенных бакетов или операций с объектами. В таблице перечислены разрешения и описано, что они означают в контексте объектов и бакетов.

Разрешение

Применение к бакету

Применение к объекту

READ

HeadBucketGetBucketLifecycleGetBucketNotificationListObjectsListPartsListMultiparts

Позволяет получить содержимое объекта и его метаданные

WRITE

Позволяет создавать, удалять, перезаписывать любые объекты в бакете

Не применимо

READ_ACP

Позволяет читать ACL бакета: GetBucketAclGetBucketCors

Позволяет читать ACL объекта: GetObjectAcl

WRITE_ACP

Позволяет изменять ACL бакета

Позволяет изменять ACL объекта PutObjectAcl

FULL_CONTROL

Комбинирует права READ, WRITE, READ_ACP, WRITE_ACP для бакета

Комбинирует права READ, WRITE, READ_ACP, WRITE_ACP для объекта

Сопоставление разрешений ACL и разрешений политики доступа
ACL допускает только конечный набор разрешений по сравнению с количеством

разрешений, которые можно установить в политике доступа. Каждое из этих разрешений позволяет выполнять одну или несколько операций Cloud Storage.

В следующей таблице показано, как каждое разрешение ACL сопоставляется с соответствующими разрешениями политики доступа. Как видно, политика доступа допускает больше разрешений, чем ACL. ACL используется в основном для предоставления базовых разрешений на чтение и запись, аналогично разрешениям файловой системы.

Разрешение ACL

Политика доступа для бакета

Политика доступа для объекта

READ

ListBucket,ListBucketMultipartUploads

GetObject

WRITE

PutObject,DeleteObject

Не применимо

READ_ACP

GetBucketAcl

GetObjectAcl

WRITE_ACP

PutBucketAcl

PutObjectAcl

FULL_CONTROL

Эквивалент предоставлению READ, WRITE,READ_ACP, и WRITE_ACP ACL разрешений

Эквивалент предоставлению READ, READ_ACP и WRITE_ACP ACL разрешений

Ключи состояния
При предоставлении разрешения политики доступа, можно использовать условные ключи, чтобы ограничить значение ACL для объекта с помощью политики бакета. Приведенные ниже контекстные ключи соответствуют спискам ACL. Эти контекстные ключи

предназначены для указания использования определенного ACL в запросе:

s3 — доступ на чтение.
s3 — права записи.
s3 — доступ на чтение ACL бакета.
s3 — права на запись ACL бакета.
s3 — полный контроль.
s3 — использование шаблонного ACL.

Фиксированный ACL
Cloud Storage поддерживает набор предопределенных разрешений, известных как стандартные списки ACL. Каждый фиксированный ACL имеет предопределенный набор получателей и разрешений. В следующей таблице перечислены стандартные списки ACL и связанные с ними предопределенные разрешения.

Фиксированный ACL

Относится к

Разрешения добавлены в ACL

private

Бакет и объект

Владелец получает FULL_CONTROL. Больше ни у кого нет прав доступа (по умолчанию).

public-read

Бакет и объект

Владелец получает FULL_CONTROL. Группа AllUsers получает READ доступ.

public-read-write

Бакет и объект

Владелец получает FULL_CONTROL. Группа AllUsers получает READ и WRITE доступ.

aws-exec-read

Бакет и объект

Владелец получает FULL_CONTROL.

authenticated-read

Бакет и объект

Владелец получает FULL_CONTROL. Группа AuthenticatedUsers получает READ доступ.

bucket-owner-read

Объект

Владелец объекта получает FULL_CONTROL. Владелец бакета получает READ доступ. Если указать этот шаблонный ACL при создании бакета, Cloud Storage проигнорирует его.

bucket-owner-full-control

Объект

И владелец объекта, и владелец бакета получают FULL_CONTROL над объектом. Если указать этот фиксированный ACL при создании бакета, Cloud Storage проигнорирует его.

В запросе можно указать только один из этих фиксированных списков ACL.

В запросе указывается фиксированный ACL, используя заголовок запроса x-amz-acl. Когда Cloud Storage получает запрос со стандартным списком управления доступом в запросе, он добавляет предопределенные разрешения в список управления доступом ресурса.

Списки управления доступом (ACL)
Hotbox предоставляет возможность управлять доступом к контейнерам и объектам с помощью списка управления доступа - ACL. У каждого контейнера и объекта есть свой список доступа. Этот список определяет каким проектам или глобальным группам предоставляются права доступа и соответствующие права доступа. При получении запроса на ресурс сервис проверяет соответствующий ACL на наличие прав доступа у запрашивающего.

При создании контейнера или объекта сервис создает стандартный ACL, который предоставляет владельцу ресурса право полного контроля над этим ресурсом, и запрещает доступ остальным проектам и глобальным группам. Это показано в следующем примере ACL бакета (стандартный ACL объекта имеет ту же структуру).

```
1<?xml version="1.0" encoding="UTF-8"?>
2<AccessControlPolicy xmlns="http://BucketName.hb.vkcs.cloud/doc/2006-03-01/">
3  <Owner>
4    <ID>***Owner-Canonical-User-ID***</ID>
5    <DisplayName>owner-display-name</DisplayName>
6  </Owner>
7  <AccessControlList>
8    <Grant>
9      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
10xsi:type="Canonical User">
11       <ID>***Owner-Canonical-User-ID***</ID>
12       <DisplayName>owner-display-name</DisplayName>
```

```
13      </Grantee>
14      <Permission>FULL_CONTROL</Permission>
15    </Grant>
16  </AccessControlList>
17</AccessControlPolicy>
```

Блок Owner определяет владельца по каноническому идентификатору пользователя проекта и по домену.
Блок Grant определяет получателя прав (проект сервиса или глобальную группу) и предоставляемое право доступа.
Базовый ACL содержит один элемент Grant для владельца. Вы можете предоставлять права с помощью добавления элементов Grant, где каждый из них определяет получателя прав и соответствующее право доступа.

Получатель прав
Получателем прав может являться проект сервиса или одна из глобальных групп сервиса. Вы можете предоставлять права проекту сервиса при помощи адреса электронной почты (домена) или канонического идентификатора проекта. При этом если вы указываете адрес электронной почты (домен) в вашем запросе на права доступа, то сервис определяет канонический идентификатор пользователя для соответствующего проекта и добавляет его в ACL. В результате ACL всегда будут содержать канонический идентификатор пользователя для проекта сервиса, а не адрес электронной почты проекта (домен).

Глобальные группы сервиса
У сервиса существует набор предопределенных групп. При предоставлении группе прав доступа к проекту вы указываете один из наших URI вместо канонического идентификатора пользователя. Сервисом предоставляются нижеуказанные глобальные группы.

Группа Authenticated Users — группа авторизованных пользователей.
Данная группа представляет собой все проекты сервиса. Наличие права доступа к этой группе позволяет любому проекту сервиса получать доступ к ресурсу. Но в то же время все запросы должны быть подписаны (авторизованы).

Группа All Users — группа всех пользователей.
Наличие права доступа к этой группе позволяет всем получать доступ к ресурсу. Запросы могут быть подписанными (авторизованными) или неподписанными (анонимными). В неподписанных запросах отсутствует заголовок аутентификации Authentication в запросе.

Предоставляемые разрешения
Следующая таблица содержит набор разрешений, поддерживаемых сервисом в ACL. Необходимо отметить, что набор разрешений ACL один и тот же для объекта и контейнера (за исключением запрета права WRITE на объекте).Нижеследующие таблицы содержат разрешения и описывают их в контексте разрешений объекта и контейнера.

Разрешение

При предоставлении на контейнере

При предоставлении на объекте

READ

Позволяет получателю прав получить список объектов в контейнере.

Позволяет получателю прав прочитать данные объекта и его метаданные.

WRITE

Позволяет получателю прав создавать, переписывать и удалять любой объект в контейнере.

Неприменимо.

READ_ACP

Позволяет получателю прав прочитать ACL контейнера.

Позволяет получателю прав прочитать ACL объекта.

WRITE_ACP

Позволяет получателю прав записывать ACL для соответствующего контейнера.

Позволяет получателю прав записывать ACL для соответствующего объекта.

FULL_CONTROL

Дает получателю прав следующие разрешения на контейнер: READ, WRITE, READ_ACP и WRITE_ACP.

Дает получателю прав следующие разрешения на объект: READ, READ_ACP и WRITE_ACP.

Соответствие разрешений ACL и разрешений политики доступа

Каждое из прав доступа позволяет провести в сервисе одну или несколько операций. Следующая таблица показывает соответствие прав доступа и операций.

Разрешение ACL

Соответствующее разрешение политики доступа при предоставлении разрешения ACL на бакете

Соответствующее разрешение политики доступа при предоставлении разрешения ACL на объекте

READ

HeadBucketListMultipartsListObjectsListParts

GetObjectHeadObject

WRITE

AbortMultipartUploadCompliteMultipartUploadInitiateMultipartUploadUploadPartPutObjectDeleteObject

Неприменимо

READ_ACP

GetBucketAcl

GetObjectAcl

WRITE_ACP

PutBucketAcl

PutObjectAcl

FULL_CONTROL

Эквивалентно предоставлению следующих разрешений ACL: READ, WRITE, READ_ACP и WRITE_ACP.

Эквивалентно предоставлению следующих разрешений ACL: READ, READ_ACP и WRITE_ACP.

Связанный ACL
Сервис поддерживает набор предопределенных предоставлений разрешений — так называемых «готовых ACL». Каждый готовый ACL содержит предопределенный набор получателей прав и разрешений. Следующая таблица содержит набор готовых ACL и связанных с ними предопределенных предоставлений разрешений.

Связанный ACL

Область применения

Добавленные в ACL разрешения

private

Контейнер и объект

Владелец получает полные права (FULL_CONTROL). Остальные пользователи не получают прав доступа (по умолчанию).

public-read

Контейнер и объект

Владелец получает полные права (FULL_CONTROL). Группа всех пользователей Allusers получает право доступа на чтение (READ).

public-read-write

Контейнер и объект

Владелец получает полные права (FULL_CONTROL). Группа всех пользователей AllUsers получает права доступа на чтение (READ) и запись (WRITE). Как правило, не рекомендуется предоставлять данные разрешений на контейнер.

authenticated-read

Контейнер и объект

Владелец получает полные права (FULL_CONTROL). Группа авторизованных пользователей (AuthenticatedUsers) получает права доступа на чтение (READ).

bucket-owner-read

Объект

Владелец объекта получает полные права (FULL_CONTROL). Владелец бакета получает права на чтение (READ).

bucket-owner-full-control

Объект

Владелец объекта и владелец бакета получают полные права (FULL_CONTROL) на объект.

Можно указывать только один из этих связанных ACL в своем запросе — только при установлении ACL через заголовки.

Установка ACL
API-сервис позволяет вам установить ACL при создании бакета или объекта. Сервис также предоставляет API для возможности установки ACL на существующем бакете или объекте. Эти API дают возможность устанавливать ACL с помощью нижеуказанных способов.

Установка ACL с помощью заголовков запроса — при отправке запроса по созданию ресурса (бакета или объекта) вы устанавливаете ACL при помощи заголовков запроса. Данные

заголовки позволяют вам указать или готовый ACL, или установить предоставления разрешений явным образом (однозначно определить получателя прав и разрешения). Установка ACL с помощью тела запроса — при отправке запроса по установке ACL на существующем ресурсе вы можете установить ACL или в заголовке запроса, или в теле.

Revision #3
Created 17 October 2025 11:00:50 by Admin
Updated 17 October 2025 19:09:09 by Admin