Добрый день, уважаемые студенты! Я рад вас всех видеть. Мы с вами вместе добрались до 4-го модуля и сегодня поговорим про безопасность в облаке, т.е. о AWS Cloud Security.

Сегодня мы с вами разберем ряд AWS сервисов, связанных с безопасностью. Некоторые сервисы мы разберем более подробно, о чем-то поговорим менее подробно, точнее остановимся на самых основных моментах и определениях менее популярных сервисов. Тем не менее, эти сервисы могут прийти на реальном экзамене AWS, поэтому знания и информация, которую вы получите на наших занятиях, будет достаточно для того, чтобы вы успешно ответили на вопросы в реальном экзамене AWS.

Давайте вкратце остановимся на каждой из секций. Первая – это AWS shared responsibility model. Мы поговорим, что же это такое и как оно нам помогает. Далее мы познакомимся с сервисом AWS IAM. После этого мы определим те шаги, которые нам помогают обезопасить наш новый AWS аккаунт. Далее познакомимся с некоторыми сервисами AWS, которые также связаны с безопасностью. Также поговорим о том, как мы можем защитить наши данные в облаке.

Мы с вами начинаем первую секцию, а именно AWS shared responsibility model. Когда мы говорим про безопасность в облаке, то всегда есть две стороны, первая сторона – это мы, как пользователи облака, и с другой стороны – это облачный провайдер. А так же логически подразделяем эти ответственности на две части. Это то, что облачный провайдер отвечает за security of the cloud, т.е. за безопасность самого облака. Когда мы говорим про пользователей, т.е. нас, значит, мы отвечаем за безопасность security in the cloud, т.е. безопасность в самом облаке.

Мы сказали, что AWS облачный провайдер отвечает за безопасность облака. Что же это значит? Это значит что он обеспечивает физическую безопасность дата-центров 24/7, обеспечивает охраной и всем необходимым, что не дает возможность посторонним проникнуть в эти дата-центры. Следующее – это безопасность hardware and software infrastructure, это все те железа, т.е. наши сервера, жесткие диски и все любое другое оборудование, которое принадлежит AWS. Дальше software infrastructure, имеется в виду операционные системы, все приложения, приложения виртуализации, все это также обеспечивается безопасностью, и на постоянной основе обновляется, устанавливаются различные обновления безопасности для того, чтобы никто как физически, так и виртуально не мог проникнуть в границы облачного провайдера.

Далее мы говорим о безопасности сетевой инфраструктуры и виртуализации, т.е. наших роутеров, свитча, load balancer, firewall, кабельной части и всех кабелей и т.д.

К сожалению, мы с вами не можем приехать в любой дата-центр посмотреть и убедиться, так как это запрещено, но на постоянной основе AWS приглашает специальных аудиторов для того, чтобы они проверили, убедились, выпустили соответствующие отчеты о том, что вся инфраструктура обеспечена безопасностью и соответствует стандартам, и по необходимости на бесплатной основе есть возможность скачивать все отчеты, которые периодически выходят.

Давайте теперь подробнее поговорим, за что отвечает пользователь, т.е. мы, и как обеспечить безопасность в облаке. В случае, если мы используем сервис Amazon EC2, то наша ответственность выбрать ту операционную систему, которая обновлена и не имеет проблем с безопасностью, если же есть проблемы, то нужно установить все патчи безопасности. Далее это наша ответственность выбирать те приложения, у которых нет опять же уязвимостей, либо если есть, то должны быть применены соответствующие патчи безопасности. Когда мы говорим про инфраструктуру, то при настройке сетей мы должны правильно настроить security группы, правильно должна настроена вся сеть, чтобы не было дыр безопасности. Также нужно провести некоторую беседу с командой для того, чтобы проводить Account management, как правильно сохранять, правильно хранить ключи безопасности, логины, пароли и т.д. И самое основное – это безопасность наших данных, т.е. данных in transit, когда мы передаем данные должны быть зашифрованы, и безопасность данных at rest, т.е. во время хранения они тоже должны быть в зашифрованном виде. Это для того, чтобы исключить возможные атаки, потерю данных, взлом самой инфраструктуры и поломку всего того рабочего процесса, который есть.

Теперь хотелось бы рассмотреть вопросы безопасности с точки зрения моделей предоставления сервиса. Как мы помним, у нас есть три вида, это IaaS, PaaS и SaaS. Здесь основная идея в том, что начиная от IaaS до SaaS, уровень контроля над IT-ресурсами становится меньше, и ту часть, которую мы перестаем контролировать и передаем облачному провайдеру, то обеспечение безопасности этой части также переходит облачному провайдеру.

Например, в случае с IaaS говорим про Amazon EC2, то обеспечение безопасности это на нашей стороне. В случае, когда мы говорим Amazon RDS, это PaaS решение, то мы за операционную систему, за ее безопасность уже не отвечаем. Мы это передали облачному провайдеру. В случае, когда мы говорим Software as a Service, например, та же почта, либо есть такой сервис Amazon Chime, это аналог Zoom, они предоставлены в виде готового приложения, и уже фактически у нас мало контроля или практически нет контроля ее как-то изменять. Таким образом, все то, что не относится к нам, должно быть обезопасено на стороне облачного провайдера.

Для того, чтобы закрепить пройденный материал, я бы хотел с вами провести некоторую активность. Я буду задавать вопросы, а вам нужно ответить правильно: за это ответственен AWS, т.е. это относится к security of the cloud, или за это ответственны пользователи облака, т.е. мы, и относятся к security in the cloud.

Сейчас я вкратце пройдусь по инфраструктуре. Представим, что у нас есть AWS аккаунт. Мы используем сервис Amazon S3, дальше мы используем сервис VPC, в котором у нас есть инстансы Amazon EC2, а также есть инстансы базы данных Oracle. Давайте теперь пройдемся по вопросам. Вам нужно будет ставить на паузу видео, будет время подумать, и будем смотреть, сможете ли вы ответить правильно.

Первый вопрос. Обновление и установка патчей безопасности операционной системы на Amazon EC2 инстансы. Можете поставить на паузу. Правильный ответ: the customer, т.е. за это ответственны пользователи.

Следующий вопрос. Физическая безопасность дата-центра? Правильный ответ: AWS.

Третий вопрос. Кто отвечает за безопасность инфраструктуры в виртуализации? Правильный ответ: AWS.

Следующий вопрос. Кто отвечает за правильную настройку security group в Amazon EC2 инстансах? Правильный ответ: the customer, т.е. мы.

Пятый вопрос. Кто отвечает за безопасность конфигурации, т.е. за настройки приложений, которые запущены на Amazon EC2? Мы либо AWS. Правильный ответ: the customer, т.е. за это ответственны мы.

Следующий вопрос. Кто отвечает за установку обновлений патчей безопасности инстансы Oracle, который запущен как инстансы Amazon RDS? И правильный ответ – AWS. Мы не занимаемся настройкой операционной системы, обновлением безопасности, потому как она предоставлена как PaaS решение, и так как AWS отвечает за это, то и за безопасность этой части тоже лежит на нем. А мы лишь получаем готовую базу данных как сервис.

Следующий вопрос. Кто отвечает за те же патчи безопасности и обновление в базе данных Oracle, которые запущены на Amazon EC2 инстансе? Правильный ответ – the customer, т.е. за этот вид развертывания базы Oracle уже отвечаем мы, потому что все то, что запускается на Amazon EC2 инстансах, ровно так же за безопасность самого Amazon EC2 инстанса, за виртуальную безопасность отвечает пользователь.

Ну и последний вопрос. Кто отвечает за правильную настройку доступа в безопасность в Amazon S3 bucket? Правильный ответ – the customer, т.е. за это ответственны мы. AWS предоставляет возможность нам эти доступы настраивать к бакетам, а за правильную настройку уже ответственны мы.

Давайте теперь рассмотрим другую архитектуру. Логика активности остается такой же. Я задаю вопрос, вы ставите на паузу и пытаетесь ответить правильно, либо за это отвечает AWS, либо за это отвечает пользователь, т.е. мы. Итак, поехали.

В нашей инфраструктуре у нас есть настроенный VPC, внутри есть настроенный subnet, в котором есть веб-сервер на Amazon EC2 инстансе. Этот инстанс взаимодействует с Amazon S3 bucket-ом, также у нас к VPC подключен internet-gateway, через который мы подключаемся к нашему веб-серверу через AWS Management Console, либо используя SSH ключи через командную строку, т.е. AWS CLI.

Итак, первый вопрос. Обеспечение безопасности AWS Management Console так, чтобы она не была взломана хакерами. За это отвечает AWS.

Далее, правильная конфигурация subnet-ов? Правильный ответ customer, т.е. AWS предоставляет нам инструменты для конфигурации subnet-ов, а наша задача их правильно настроить и наша ответственность, чтобы они были безопасно настроены.

Следующий вопрос, правильная безопасная конфигурация VPC? За это отвечаем также мы. Идея такая же, как subnet. Идем дальше.

Защита от сбоев сети в AWS регионах? Правильный ответ: AWS, т.е. сеть между регионами, это относится к глобальной инфраструктуре AWS, поэтому за это ответственен AWS.

Защита SSH ключей? За защиту SSH ключей ответственны мы, т.е. пользователи. Идем дальше.

Шестой вопрос. Обеспечение сетевой изоляции между данными пользователей AWS? За это ответственный AWS, т.е. у нас как пользователей нет никакой возможности добраться до данных других пользователей, тем более это как-то настраивать. Поэтому за это ответственен AWS.

Седьмой вопрос. Обеспечение широкополосного быстрого доступа между веб-сервером и Amazon S3 bucket-ом? Правильный ответ – AWS, т.е. опять же, это относится к глобальной инфраструктуре AWS, так как Amazon EC2 инстансы и Amazon S3 bucket находятся там. И для того чтобы предоставить быстрый доступ между двумя этими сервисами, мы не используем интернет, точнее AWS не использует интернет, а использует свою глобальную инфраструктуру для того, чтобы это было быстрее и безопаснее. За это ответственен сам AWS.

И последний вопрос. Продвижение использования MFA для всех авторизаций пользователей? За это ответственен customer, т.е. мы. Идея в чем? Идея в том, что мы внутри IAM создаем пользователей, таким образом обычные физические пользователи, наши сотрудники могут, используя выданный логин-пароль, авторизоваться в нашем AWS аккаунте. Они работают все в одном AWS аккаунте и помимо логина, пароля можно настроить так называемый MFA. Обычно ннастраивается виртуальный MFA, устанавливается приложение и каждые 30 секунд генерируется новый шестизначный код, и во время авторизации пользователь не только вводит логин и пароль, он также подтверждает, что он имеет доступ к авторизованному устройству. Таким образом вводит сгенерированный новый ключ, и после этого дальше заходит. И вот продвижение и включение этой опции MFA, оно полностью лежит на customer, т.е. на нас. Она изначально отключена, но есть рекомендация от AWS ее включить и в некоторых случаях бывает ее не включают, это все зависит от зрелости компании. Чем она больше, тем больше она обращает внимание на возможные уязвимости, на возможные места атаки, таким образом через MFA мы еще больше гарантируем себе, что аккаунт не будет взломан и злоумышленник не подключится к облаку AWS для того, чтобы что-то украсть, либо сломать инфраструктуру. На этом мы заканчиваем первую секцию.

Давайте остановимся на основных моментах. Мы разобрали, что такое shared responsibility model и поговорили о том, что для полной безопасности облака, обе стороны, что облачный провайдер, что пользователь, т.е. мы, должны обеспечить безопасность в определенных местах. Эти определенные места можно разделить на два направления, т.е. AWS обеспечивает безопасность of the cloud, т.е. самого облака, а customer, уже работая внутри облака, обеспечивает security in the cloud, т.е. безопасность в облаке. В случае AWS подразумевается физическая безопасность дата-центров, железа, всего того ПО, которое используется, чтобы у него не было ошибок, дыр безопасности и это все преждевременно обновлялось. Когда мы говорим про customer, то подразумевается правильная настройка тех сервисов для того, чтобы не открыть некоторые места для возможной атаки, правильный management, правильное управление ключами безопасности, логинами, паролями, т.е. скажем так credentials и своими пользователями. Также в зависимости от сервиса, который мы используем, они у нас подразделяются на три вида по предоставлению этого сервиса. Это модель IaaS, модель PaaS и модель SaaS. В зависимости от уровня у нас меняется контроль над IT-ресурсом, ну и, соответственно, определяется тем самым, отвечаем ли мы за безопасность той или иной части конкретного ресурса.

В случае, когда мы говорим про IaaS, так как у нас больше контроля над IT-ресурсами, значит у нас больше мест, где мы должны обеспечить безопасность. Когда мы говорим про PaaS и SaaS, контроль у нас уменьшается. Соответственно, за ту часть, за которую уже ответственен облачный провайдер, он и там же обеспечивает безопасность. Это уже не наша задача.

Мы переходим ко второй части, а это – сервис AWS Identity and Access Management, т.е. AWS IAM. Это тот сервис, который обеспечивает любые доступы, так или иначе связанные с текущим AWS аккаунтом и всеми теми ресурсами, которые в нем могут быть. Таким образом, мы говорим через этот сервис, кто может получить доступ, на какие ресурсы может получить доступ, как он и при каких условиях получает этот доступ, а также следует отметить, что AWS IAM – это бесплатный сервис, за который мы ничего не оплачиваем.

Давайте теперь остановимся на основных компонентах сервиса AWS IAM.Это IAM user, IAM group, IAM policy и IAM role. У каждого есть своя цель. Когда мы говорим про IAM user, это некая сущность, у которой есть свой идентификатор, у нее также есть свой ARN, есть ключи доступа, логин, пароль, и мы можем эту информацию передать нашему сотруднику в компании для того, чтобы он мог под этим user-ом подключаться к нашему AWS аккаунту в облаке.

Далее мы говорим, что у нас есть IAM groups и мы наших пользователей можем объединять в этих группах. Это используется когда у нас есть несколько пользователей, которые схожи по должностным обязанностям, либо как минимум у них должен быть равный доступ внутри облака, то мы можем этих пользователей добавить к специально созданной группе с соответствующим именем и уже группе выдать необходимые права. Таким образом, всем тем пользователям, которые там сидят, у них будет доступ к соответствующим сервисам и действиям. Здесь следует отметить, что в одной группе может быть несколько пользователей, но и каждый пользователь может быть в нескольких группах. Это тоже верно. И пользователь, например, может для определенного набора сервисов получить доступ через одну группу и через другую группу получить доступ к другим сервисам. Таким образом, в совокупности у него больше будет доступа, чем у другого пользователя, который сидит только в одной группе. Также можно более тонко настроить доступы каждого пользователя внутри AWS.

Когда мы говорим IAM policy, это тот документ, который описывается в виде JSON, и он описывает информацию до каких ресурсов, на какие операции есть доступ, но у какой сущности не прописывается, мы привязываем policy к user напрямую, либо к group. В целом, best practice это policy привязывать к группам. Даже если у нас есть один пользователь, нам все равно рекомендуется создать группу этого пользователя, подключить к этой группе и уже после все policy, которые мы создаем, привязывать не к пользователю, а к созданным нами группам.

Ну и следующая последняя сущность – это IAM role. Идея в том, что мы говорили, что мы строим IT-инфраструктуру в облаке, она состоит из разных компонентов, каждый компонент – это какой-то определенный сервис, ее ресурс. И они взаимодействуют между собой для того, чтобы выполнять какую-то бизнес-задачу. Так вот, изначально у любого сервиса нет никаких доступов, нет никаких прав, он только может работать внутри своего сервиса. Когда мы создаем, например, AWS Lambda, т.е. функцию, то мы прописываем там код, внутри может быть какая-то логика, но у этого кода нет доступа к другим AWS ресурсам, т.е. мы не можем обратиться к какому-то определенному Amazon S3 bucket-у для того, чтобы оттуда извлечь какой-то файл. Этого нет. Поэтому в AWS Lambda есть возможность привязать role к AWS Lambda, а к role уже привязать policy, который дает доступ на соответствующий бакет. После чего уже можно внутри кода обращаться к этому бакету, и мы не получим ошибку, что там access denied и доступ запрещен.

Еще один use case когда используются role, это когда у нас есть несколько AWS аккаунтов, и представим, что мы всех пользователей создали в одном аккаунте, но наша инфраструктура работает как в первом, так и во втором AWS аккаунте. Нам нужно не создать пользователей во втором AWS аккаунте, а нужна возможность как-то обратиться к этим пользователям в первом AWS аккаунте к ресурсам во втором. Это делается следующим образом: на втором AWS аккаунте создается соответствующая role, прописывается trust policy, т.е. прописываются те AWS аккаунты и те пользователи, которые могут использовать эту role. Таким образом, на стороне первого аккаунта мы выдаем для соответствующих пользователей либо групп возможность assume, т.е. возможность использовать role, которая находится в другом аккаунте, т.е. с двух сторон должны разрешить доступ. В этом случае есть возможность этому пользователю, зная название аккаунта, куда он подключается, зная название role, которую он будет применять, он меняет свою role, переподключается. Таким образом, он скажем так, переселяется или телепортируется с одного AWS аккаунта в другой AWS аккаунт. Все те доступы, которые есть у этой role во втором AWS аккаунте, у него появляются. И в рамках этой role он может работать уже во втором AWS аккаунте.

Давайте теперь поговорим, как пользователь может подключаться к AWS аккаунту. Есть два варианта. Первый – это программное, когда мы используем AWS CLI, либо AWS SDK. В этом случае ему нужно использовать свой access key ID либо secret access key, который генерируется при создании пользователя. У нас будет отдельное Демо, и все те моменты, которые мы проговорим на этой лекции, мы проделаем в этом Демо.

Второй вариант – это когда пользователь подключается к AWS Management Console. В этом случае ему нужно знать 12-значный account ID или alias этого аккаунта, если он настроен. Также он должен знать свой логин, т.е. username, также пароль от этого пользователя.

Ну и третье – если подключен MFA, и он требуется, то он должен через приложение, которое установлено на устройстве virtual MFA ввести генерируемый код для того, чтобы успешно подключиться.

Теперь давайте поговорим про авторизацию. Есть автентификация – это когда пользователь подключается к облаку и говорит, что пользователь – это он, и в этом случае он заходит в систему. Когда мы говорим про авторизацию, тут уже вопрос в том, что этому пользователю, которому поняли, мы его идентифицировали, что он может делать. Изначально, когда создается только пользователь, у него нет доступа никуда, т.е. есть такой термин как implicit deny, это неявный запрет на все ресурсы, если оно явно не предоставлено, оно обычно включается по умолчанию, и чтобы предоставить доступ, мы к этому пользователю либо к группе, которая подключена к пользователю, подключаем какой-то AWS policy, в котором как раз таки прописывается куда, при каких условиях, на какие конкретные операции есть доступ. После этого важно еще упомянуть, есть такое понятие как principle of least privilege. Это говорит о том, что нужно максимально стараться предоставлять только нужные доступы.

Например, если пользователю нужен доступ в Amazon S3 bucket к режиму чтения, и возможность скачивать файлы для того, чтобы выполнять свою работу, то этому пользователю не надо предоставлять полный доступ, т.е. не нужно давать ему возможность создавать новые бакеты, обращаться к другим файлам и бакетам, к которым он не должен заходить, а обращаться только к тем, с которыми он работает. Также в рамках того же бакета не нужно ему выдавать доступ на удаление либо на загрузку новых файлов, если того не требует его должностные обязанности. Нужно просто следовать принципу выдавать только доступ, который необходим.

Ну и последнее, что стоит отметить, сервис IAM – глобальный. Когда вы подключаетесь и открываете этот сервис, то сверху справа вы не увидите, к какому региону он относится, а увидите надпись global. Это говорит о том, что созданные пользователи автоматически доступны на всех регионах. Если вам необходимо ограничить доступ к определенным регионам, то это можно сделать внутри IAM policy, который вы привяжете к этому пользователю, либо к группе.

Давайте теперь подробнее поговорим, что такое IAM policy, каких видов они могут быть. IAM policy подразделяется на identity-based policy и resource-based policy. Identity-based policy – это policy, которая привязывается к некой сущности, т.е. identity, а в нашем случае это может быть роль, пользователь или группа. Когда мы говорим resource-based policy, то на некий ресурс определенного сервиса есть возможность привязать policy, которые относятся только к нему, и она описывает какие сущности могут обращаться либо что-то делать конкретно с этим ресурсом.

Примером может быть Amazon S3 bucket. Вы можете к нему привязать некоторую resource-based policy, которая говорит, что у пользователей, которые находятся в группе Administrators, есть возможность изменять свойства этого бакета. А пользователям, которые сидят в, например, Simple Users группе есть возможность скачивать файлы внутри этого бакета, но нет возможности удалять либо изменять эти файлы.

Когда мы говорим также про identity-based policy, подразделяются три вида. Первое – это custom policy, которую мы прописываем. Она существует отдельно от чего-либо и вы это policy уже привязываете к role, пользователю либо группе. Может быть managed policy, это значит, она управляется и обновляется, поддерживается самим Amazon-ом. На каждый сервис может быть несколько managed policy разного уровня. Например, мы говорим Amazon S3, есть AmazonS3 full access policy, это когда в этом policy прописывается так, что любой identity, у которого есть этот policy, имеет полный контроль к сервису Amazon S3.

Если же мы говорим про policy попроще, это power user policy, в этом случае это продвинутый пользователь, у которого нет административных прав, он не может создавать либо удалять бакеты, но он может работать со всем тем, удалять, создавать файлы внутри бакетов.

Ну и последнее может быть readonly policy, которая предоставляет только возможность считывать некоторые файлы, при этом создавать, удалять, а тем более создавать новые бакеты или удалять эти бакеты, он не может.

Третий вариант это inline policies, это когда мы не создаем отдельно стоящий policy с отдельным АRN, с отдельным ID, а создаем внутри либо определенные группы или определенные role, так называемые inline policy. Эта policy привязана к группе или role, и она не может существовать отдельно. Если вы создаете какое-то уникальное policy, то нет необходимости ее создавать отдельно, а можно создать конкретно к role либо к группе, и она там будет привязана и также работать. Если же у вас какая-то policy, которую вы будете привязывать к нескольким разным role, либо к нескольким разным группам, то стоит рассмотреть создание отдельно стоящей policy, для того чтобы ее уже привязывать к разным сущностям.

Другой важный момент, который стоит упомянуть – это в каком виде прописываются наши policy. Это JSON документ, который идет в определенной структуре, у него есть определенные ключи, которые мы разберем на следующих слайдах, и соответствующие ожидаемые значения. Когда мы также говорим про policy, то вне зависимости от того, какой это ресурс, либо какая это сущность, рассматриваются все имеющиеся policy на конкретную сущность или ресурс. В тот момент, когда доступ есть, он выдается, в тот момент, когда доступа нет, она не выдается и пишется сообщение implicit deny, т.е. неявный запрет. Также когда мы явно прописываем в policy explicit deny, т.е. доступ запрещен, то любой доступ, который вы в других местах в любом количестве указываете как разрешен, но хотя бы в одном другом месте прописано, что он запрещен, то оно преобладает над любым разрешением и доступ вы не получаете.

Давайте рассмотрим пример IAM policy. Вы здесь видите пример IAM policy, это JSON документ, у него есть определенная структура, ожидаются определенные ключи и значения. Давайте подробнее рассмотрим. Мы видим, что есть version, есть statement. Version – это стандартное значение, 2012-10-17, ее вы не меняете, она остается, если что-то поменяется, то эту информацию вы всегда можете найти в AWS документации.

Идем дальше. Statement, т.е. это те наши policy, которые передаются в массиве. В нашем случае мы видим, что у нас два policy, первый – это разрешает, т.е. в пункте effect вы видите allow, второй – deny. Два возможных варианта, либо allow, либо deny. Если мы говорим про action, то здесь перечисляется список операций, которым мы либо предоставляем доступ, или запрещаем доступ.

Что такое операция? Это название сервиса, двоеточие и название операции. Например, S3 : get object. Это мы выдаем, либо запрещаем доступ к сервису Amazon S3, а именно к операции получить объект, т.е. после того, как у нас есть это разрешение, мы можем скачать файл. Если мы видим место операции звездочку, это значит все операции в рамках этого сервиса предоставляются.

Давайте рассмотрим следующий – это resource, здесь мы указываем список ресурсов, для которых мы хотим создать наш policy. Здесь важный момент, который стоит упомянуть, это то, что, еще раз повторюсь, вне зависимости от того, какое количество allow policy вы создаете, если у вас в каком-то из любых существующих policy на этот ресурс, либо на эту сущность, т.е. identity, есть deny policy, то оно имеет приоритет перед всеми allow. Когда мы говорим, что нет ни allow, ни deny policy для конкретного ресурса, для конкретной операции конкретного сервиса, то в этом случае нет явного предоставления доступа, т.е. allow, и в этом случае мы ничего не предоставляем, по дефолту у нас implicit deny, т.е. неявный запрет.

Пару дополнительных моментов, касательно resource-based policy. Мы с вами проговорили, что это те policy, которые создаются для определенного ресурса, и в этом документе указывается, так как она привязана уже к ресурсу, то мы в policy документе указываем, кто и на какую операцию получает или наоборот запрещает доступ.

Давайте все то, что мы проговаривали ранее, рассмотрим как блок-схему. Когда мы говорим, что у нас есть любой policy, не важно, это resource-based policy, либо это identity-based policy, то мы проверяем, есть ли на конкретную операцию для этого пользователя, на конкретный ресурс explicit deny, если есть, мы говорим что запрещаем доступ. Если explicit deny, т.е. явного запрета нет, то мы двигаемся дальше, теперь ищем, есть ли explicit allow, значит, есть ли явное разрешение на конкретную операцию, если есть, то доступ выдается. В том случае, если нет ни explicit deny, ни explicit allow, то в целях безопасности мы запрещаем это действие, что называется implicit deny. Это важный момент, поэтому я постарался в нескольких видах это объяснить.

Двигаемся к следующему основному понятию, это IAM groups. В целом мы про него поговорили, давайте теперь закрепим более подробные детали. Первое – в группе могут быть несколько пользователей и соответственно каждый пользователь может быть в нескольких группах. Второй момент – группа не может быть вложной, таким образом вы не можете добавить какую-то группу в другую группу, в группе могут быть только пользователи. Третье – не существует какой-то группы по умолчанию, в которой у вас сидят все ваши пользователи. Если вам нужно всех пользователей добавить в какую-то группу, вам нужно будет это сделать либо вручную, либо написать простой скрипт, но это все делается на вашей стороне, создается специальная группа, как хотите ее называете и добавляете всех ваших пользователей.

Следующий важный компонент IAM – IAM roles. IAM roles в большинстве случаев используется в двух случаях. Первый случай – это для того, чтобы сервисам предоставить доступ к другим сервисам. Один из примеров мы в прошлый раз рассматривали. Представим, у нас есть лямбда-функция и она в своей работе должна подключаться к серверу Amazon S3, с него загружать по определенному пути некоторый файл, изменять его и перезаливать обратно. В этом случае, для того чтобы лямбда смогла это сделать, нужно к role, которая привязана к этой функции, выдать соответствующие права. Вы прописываете policy, привязываете к role, но мы помним с вами, что мы можем создать inline policy, которые есть только в этой role, а также можем воспользоваться managed policy, которая управляется, поддерживается самим Amazon. Второй вариант использования role – это кросс-аккаунтный доступ пользователей с одного AWS аккаунта к ресурсам другого AWS аккаунта. Идея в том, что на втором аккаунте создается и прописывается role, в котором указывается account ID основного account ID, а здесь в основном account ID, где сидят ваши пользователи, выдается доступ к assume role операции, указывается конкретный ресурс, ARN этого ресурса на втором AWS аккаунте, и только после этого, когда с двух сторон выдали разрешение, то у этого пользователя есть возможность переключиться на ту role и со всем тем доступом, который есть в этой role, работать в другом AWS аккаунте.

Мы с вами подошли к концу второй части нашей сегодняшней лекции, познакомились с сервисом IAM. Это очень важный сервис, который во всех проектах, во всех возможных IT-инфраструктурах в облаке, она используется по той простой причине, что мы работаем удаленно, подключаемся к облаку удаленно, и для того, чтобы это делать, нам нужно выдать определенные права. Более того, когда мы говорим IT-инфраструктура, это такая система, где внутри системы сидят наши сервисы, и они работают между собой. Чтобы эти сервисы могли работать между собой, нам также нужно выдать соответствующие права.

Основное правило, когда мы выдаем права, нужно помнить то, что мы выдаем по принципу least privilege, т.е. выдаем только те права, которые нужны в данный момент для этого identity, либо для этого ресурса. Если остановимся на основных понятиях, то это IAM policy, это то, что выдает права, и у нас есть три сущности, это IAM user, IAM group, и IAM role.

Мы с вами добрались до третьей секции, и здесь мы поговорим, какие шаги нужно предпринять, какие действия нужно сделать для того, чтобы максимально обезопасить ваш новый созданный AWS аккаунт. Когда вы создаете новый AWS аккаунт, то в этом аккаунте ничего нет. Единственный вариант подключиться к облаку, это под root user, это так называемый супер пользователь, у которого есть полный доступ ко всему. В best practices не рекомендуется использовать root user для ежедневных задач, создать специальный пароль, создать для него MFA и спрятать подальше, использовать только в тех случаях, когда есть необходимость. Есть определенные действия внутри AWS облака, когда вам нужно использовать root пользователя, вы с этими действиями будете встречаться редко, потому как обычно это разовые операции, а не какие-то повторяющиеся действия.

Теперь, если говорим про bestpractices, что рекомендует AWS, это для вас самого, как владельца этого AWS аккаунта, создать специального пользователя admin и уже под admin заходить и производить все ежедневные операции, которые вам нужны. Когда мы говорим про ваших сотрудников, то для них создаются отдельные пользователи, желательно рекомендуется создать специальную группу в зависимости от должностных обязанностей ваших сотрудников и далее уже все role, все policy привязывать к группе.

Один из других настроек, которые вам также нужно сделать в root аккаунте, это для root аккаунта, если они есть, удалить access ключи. Как мы помним, они используется для программного доступа через CLI либо SDK к облаку. Так как через root пользователя ничего происходить не будет, он обычно используется только тогда, когда вы сами лично заходите, во всех других случаях нужно использовать специально созданных пользователей, либо role, если это сервис. Поэтому эти access ключи вы удаляете. Другой момент – это то, что вам нужно создать так называемый password policy и сделать его по возможности сложным, т.е. это тот документ, который описывает насколько сложный или из каких символов должен состоять пароль для всех IAM пользователей.

Еще одним функционалом, который помогает еще больше обезопасить ваше облако, является MFA. MFA – это опциональный функционал, поэтому вам нужно ее специально включить и активировать. Что значит MFA? MFA – это Multi-Factor Authentication. Идея в том, что пользователи при логине в AWS Management Account предоставляли несколько факторов. Что такое фактор? Фактор – это, например, что пользователь знает. А пользователь знает логин и пароль. Другой фактор – это то, что у пользователя есть. Это что-то такое физическое, которое не может быть у злоумышленника. Например, это телефон и как это реализовано, есть так называемый виртуал MFA. Устанавливается приложение в ваш смартфон и делается специальная привязка вашего аккаунта внутри этого приложения с AWS Management Account пользователем. После того как привязка произошла в приложении вам каждые 30 секунд генерируется шестизначное число. Таким образом, в момент, когда вы проходите аутентификацию, вы вводите ваш логин и пароль. После этого вы вводите ваш MFA ключ и уже на стороне AWS Management Account происходит проверка: действительно ли в это время, этому пользователю было отображено это шестизначное число. Если да, оно совпадает, это говорит о том, что вы как пользователь с двух факторов себя идентифицировали. Таким образом вы исключаете еще больше вероятность того, что это не вы, а какой-то злоумышленник.

Следующим шагом, который помогает увеличить безопасность вашего облака, это так называемый AWS CloudTrail. Это сервис, который является аналогом черного ящика в самолете. Он является максимально защищенным и записывает все те вызовы к AWS API, т.е. все те изменения вашей инфраструктуры в облаке. По умолчанию включено и бесплатно хранится 90 дней всех ваших действий в вашем аккаунте. У вас так же есть такая возможность, если же вам необходимо настроить какие-то дополнительные алерты или же хранить не 90 дней, а больше для каких-то определенных положений, если вы какая-то финансовая организация либо государственная, вам необходимо эту информацию хранить какое-то количество лет. В этом случае вы создаете так называемый Trail, это ресурс сервиса AWS CloudTrail, который как раз таки накапливает все действия, происходящие в вашем аккаунте на какой-то больший период. В этом случае может взыматься определенная плата. Trail хранит все свои данные внутри Amazon S3, а так как логи это текстовая информация, то обычно расходы на AWS CloudTrail достаточно низкие. Но в случае, когда у вас возникает какая-то проблема внутри облака, для целей расследования это просто незаменимая информация, поэтому рекомендуется ее включать и держать всегда активной.

Мы с вами проходили на предыдущих занятиях о сервисах связанных с billing и затратами на облако, т.е. Cost. Вам также как best practice рекомендуется настроить алерты на бюджет, также рекомендуется настроить так называемые отчеты, так чтобы вы могли быстро и оперативно видеть все ваши расходы и предпринимать какие-то определенные действия.

Это вы можете сделать внутри AWS Cost Explorer, если же вам нужна более детальная информация, то вы можете воспользоваться сервисом AWS Cost and Usage Reports, который все данные и метаданные, связанные с вашими расходами, со всеми деталями выгружает в Amazon S3 и вы эти данные можете использовать для построения уникальных кастомных отчетов.

Зачастую использование сервиса AWS Cost and Usage Reports является достаточно продвинутым вариантом, практикуется в крупных компаниях, поэтому в вашем случае нет смысла в самом начале терять время на изучение, на настройку этих отчетов, а можно положиться на сервис AWS Cost Explorer, который все необходимое вам покажет и в удобном виде сохранит ваши отчеты.

На этом мы подошли к концу третьей части нашей лекции, и мы поговорили про все те основные шаги, действия, те сервисы и тот функционал, который доступен в AWS облаке для максимального обеспечения безопасности вашей инфраструктуры в облаке.

Мы переходим к четвертой части нашей лекции и поговорим про дополнительный набор сервисов, которые помогают нам обезопасить наши аккаунты.

Самый первый сервис – это AWS Organizations. AWS Organizations – это сервис, который позволяет центрально управлять несколькими AWS аккаунтами. Также он позволяет нам настроить consolidated billing. Идея в том, что мы все расходы на каждом из наших AWS аккаунтов собираем в одно место и отображаем в удобном виде в одном основном аккаунте.

Для того, чтобы обеспечить безопасность, на стороне AWS Organizations тоже есть некоторый функционал. Он называется Service control policies. Идея в том, что это такой же JSON документ, это такая же policy, но оно привязывается уже к AWS аккаунту. Более того, мы с вами помним, что есть функционал OU, т.е. organizational units внутри AWS Organizations, который позволяет ваши аккаунты между собой группировать. Таким образом вы можете гибко и максимально удобно группировать ваши аккаунты, а Service Control Policies есть возможность привязывать не к отдельному аккаунту, а к целой группе, т.е. к organizational units. Таким образом максимально правильно настроить доступы, например, в тех аккаунтах, где у вас происходит разработка, дать больше доступа и больше контроля над ресурсами, а в ваших продакшн аккаунтах дать максимально меньше доступа, но так, чтобы все работало корректно и исключить возможные поломки ввиду того, что у кого-то где-то больше есть доступа и если в продакшн среду попадает какой-то злоумышленник, мог нанести минимальный вред вашей IT-инфраструктуре.

Следующий сервис – AWS KMS, т.е. Key Management Service. Этот сервис помогает нам управлять нашими ключами шифрования в облаке. На текущий момент вам достаточно знать определение, более подробно об этом сервисе мы будем говорить на следующих занятиях.

Следующий сервис – Amazon Cognito. Мы этот сервис не будем подробно рассматривать, так как для цели реального экзамена достаточно знать определение и определение следующее. Это сервис, который позволяет вам управлять пользователей ваших приложений, т.е. дает возможность авторизовываться, регистрироваться и менеджить, хранить информацию о всех ваших пользователях.

Следующий managed сервис, связанный с безопасностью – AWS Shield. Вам также достаточно знать определение, оно следующее – этот сервис позволяет вам защищаться от DDoS-атак.

Мы с вами переходим к пятой части нашей лекции и поговорим про то, как мы можем защищать наши данные в облаке. Первое состояние, в котором данные могут пребывать, это так называемый at rest, т.е. это тот случай, когда данные хранятся на жестких дисках. В этом случае у нас есть возможность использовать так называемый AES-256 шифрование, либо воспользоваться сервисом AWS KMS, про который мы ранее говорили, для шифрования данных в состоянии покоя, в состоянии хранения.

Другое состояние данных, в котором их также нужно защищать, это в состоянии передачи, т.е. когда мы передаем наши данные по сети. В этом случае есть сервис AWS Certificate Manager, который позволяет нам генерировать SSL ключи, либо TLS сертификаты, для того чтобы защитить поток данных. Это также относится к веб-трафику, т.е. используя этот сервис AWS Certificate Manager мы можем настроить Secure HTTP, т.е. HTTPS соединение. Увидев этот список, вы можете определить, действительно ли для этих бакетов нужен доступ с интернета, если же нет, то вы сможете точечно пройтись по ним и заблокировать общий доступ с интернета.

В зависимости от того, в какой сфере работает ваша компания, в какой стране вы находитесь, либо в каком регионе земного шара, вы должны соответствовать некоторым стандартам качества, либо регуляторным требованиям. AWS со своей стороны на периодической основе проходит разные проверки, получает сертификаты качества и заключительные документы, что он соответствует некоторым регуляторным требованиям.

Из тех compliance программ, которые стоит упомянуть и которые могут прийти на AWS экзамене, это что AWS получил сертификат качества ISO/IEC 27001:2013. Также есть подтверждение, что соответствует требованиям акта HIPAA. Это для тех компаний, которые работают в сфере медицины. Третье – это GDPR, это для компаний, которые в большинстве случаев находятся в Европе, и в этом документе описываются требования, как хранить и как работать с персональными данными.

Все эти документы, сертификаты качества, заключительные документы вы можете скачать в так называемом сервисе AWS Artifact. Но это не совсем сервис, а больше набор страничек, в котором вы можете по требованиям, по необходимости для вашей компании скачать эти документы.

Если мы говорим compliance в рамках AWS аккаунта для себя, для того чтобы понимать, что происходит, какие ресурсы есть, какие изменения в этих ресурсах произошли, то мы можем воспользоваться сервисом AWS Config, который фиксирует все изменения и может дополнительные действия исполнить по определенным условиям. Этого определения достаточно для целей экзамена, и мы идем дальше.

У нас есть самый последний бонусный раздел нашей лекции, мы максимально быстро по нему пройдемся. Есть такой сервис AWS Service Catalog. Идея этого сервиса в том, чтобы выбрать те сервисы, которые одобрены вашей компанией, и уже предоставить вашим сотрудникам доступ к именно этим сервисам через AWS Service Catalog.

Другие три сервиса – это Amazon Macie, это тот сервис, который использует машинное обучение для того, чтобы сканировать все данные, которые есть в вашей инфраструктуре, чтобы выявить так называемые персональные данные. Дальше уже с выявленной информацией вы можете соответственно действовать, либо сохранять эти данные в специальное хранилище, либо удалить, если они случайно туда попали.

Другие два сервиса они похожи, это – Amazon Inspector и Amazon GuardDuty. Идея этих сервисов в том, что это managed-сервисы Amazon, и как только вы ее запускаете, происходит автоматическое сканирование вашего облака, вашей IT-инфраструктуры для выявления некоторых проблем безопасности, и в последующем составляется некоторый список рекомендаций, чтобы улучшить состояние безопасности вашей IT-инфраструктуры.

На этом мы добрались до последних слайдов нашей лекции. Мы завершаем нашу лекцию. Я бы хотел вкратце дать некоторый summary.

Первое, мы разобрали, что такое shared responsibility model.

Далее мы познакомились подробно с сервисом AWS IAM.

После этого мы рассмотрели те шаги и действия, которые рекомендуется предпринять при создании нового AWS аккаунта. А далее рассмотрели ряд сервисов AWS, которые так или иначе связаны с обеспечением безопасности в облаке.

Здесь хотел бы отметить, что вы большие молодцы. Я понимаю, что объем информации большой, много новых терминов, много новых названий сервисов. Самое главное, вы не переживайте. Рекомендация от меня – делать некоторые конспекты, как минимум записывать названия сервисов. Когда вы это записываете, либо пересматриваете лекции, названия сервисов уже будут запоминаться и казаться знакомыми.

Мы более подробно будем проговаривать про некоторые сервисы, которые встречаются чаще на реальном экзамене AWS. Также у нас будут активности, будет демо. Я вам покажу, как выглядит AWS Management Console на примере разных сервисов для того, чтобы лучше закрепить, визуально лучше запомнилось. У нас также будут несколько лабораторных активностей, в которых вы сами сможете попробовать и пощупать облако. Для некоторых из вас такая активность поможет лучше запомнить, понять, как выглядит облако, как с ней работать, какие преимущества вы получаете и так далее.

На этом слайде предоставлены ссылки, которые вам могут быть полезны, если вы хотите о какой-то из этих тем узнать более подробно.

На этом мы завершаем наше лекционное занятие. Спасибо вам большое. Удачи! Увидимся на следующих занятиях.