Лекция 14: Отказоустойчивость инфраструктуры

Тема: Облачная безопасность и защита инфраструктуры

Технический университет Молдовы

Лектор: Максим Масютин

Введение

Добро пожаловать на лекцию 14. Сегодня мы рассмотрим тему, которая отличает любительскую IT-эксплуатацию от профессиональной: отказоустойчивость инфраструктуры. В течение следующих двух часов мы изучим, как проектировать системы, которые не только выдерживают отказы, но и продолжают работать во время них, как восстанавливаться при наступлении катастроф и как строить организации, способные противостоять даже самым серьёзным нарушениям.

Позвольте начать с фундаментальной истины вычислительной техники: всё выходит из строя. Жёсткие диски выходят из строя. Сети отказывают. Центры обработки данных отказывают. Целые облачные регионы отказывают. Вопрос не в том, столкнутся ли ваши системы с отказом, а в том, когда они откажут и готовы ли вы к этому.

В декабре 2021 года Amazon Web Services столкнулся со значительным сбоем в регионе US-EAST-1. Это единственное событие нарушило работу тысяч компаний, затронув всё от потоковых сервисов до устройств умного дома. Компании, проектировавшие системы с учётом отказоустойчивости, продолжили работу; те, которые этого не сделали, были полностью отключены на несколько часов.

Совсем недавно, в 2024 и 2025 годах, произошло несколько крупных облачных сбоев, затронувших Azure, Google Cloud и множество провайдеров сетей доставки контента (CDN, Content Delivery Network). Каждый инцидент подтверждает один и тот же урок: отказоустойчивость это не опция, это необходимость.

К концу этой лекции вы будете понимать концепции доступности, паттерны резервирования, планирование аварийного восстановления, непрерывность бизнеса, стратегии резервного копирования, защиту от DDoS-атак, реагирование на инциденты и формирующуюся практику хаос-инженерии.

Часть 1: Концепции доступности и SLA

Что такое доступность?

Доступность измеряет, как часто система находится в рабочем состоянии и доступна для использования. Простыми словами, это процент времени, в течение которого система работает и функционирует корректно. Высокая доступность это подход к проектированию, направленный на минимизацию простоев и обеспечение доступа пользователей к сервисам, когда это необходимо.

Формула доступности проста: доступность равна времени работоспособности, делённому на общее время, умноженному на 100. Если система не работала 8 часов в году, её доступность составляет приблизительно 99,91%.

Однако доступность, это понятие более тонкое, чем простой процент. Система может быть "работающей", но отвечать настолько медленно, что пользователи не могут выполнить свои задачи. Система может быть частично доступной, работая для одних пользователей, но не для других. Истинная доступность означает, что система выполняет свою предназначенную функцию в пределах допустимых параметров.

"Девятки" доступности

В индустрии доступность принято выражать в "девятках". Каждая дополнительная девятка значительно сокращает допустимое время простоя:

Две девятки, или доступность 99%, допускают около 87,6 часов простоя в год более 3,5 дней. Это неприемлемо для большинства бизнес-приложений.

Три девятки, или доступность 99,9%, допускают около 8,76 часов простоя в год. Это подходит для внутренних приложений, которые могут допускать периодические сбои.

Четыре девятки, или доступность 99,99%, допускают около 52,6 минут простоя в год. Это целевой показатель для большинства промышленных бизнес-приложений.

Пять девяток, или доступность 99,999%, допускают около 5,26 минут простоя в год. Это требуется для критической инфраструктуры, такой как системы финансовых торгов, службы экстренной помощи и базовые телекоммуникации.

Шесть девяток, или доступность 99,9999%, допускают около 31,5 секунд простоя в год. Немногие системы достигают этого уровня, и для этого требуется исключительная инженерия и значительные затраты.

Каждая дополнительная девятка обычно обходится значительно дороже предыдущей. Переход с 99,9% на 99,99% может удвоить стоимость

инфраструктуры. Переход с 99,99% на 99,999% может увеличить стоимость в десять раз.

Соглашения об уровне обслуживания

Соглашения об уровне обслуживания (SLA, Service Level Agreement) это договоры, определяющие ожидаемую доступность и производительность сервиса. Облачные провайдеры публикуют SLA для своих сервисов. AWS гарантирует доступность 99,99% для EC2 в пределах региона при развёртывании в нескольких зонах доступности. Azure обязуется обеспечивать 99,95% для виртуальных машин с премиальным хранилищем. Google Cloud обещает 99,99% для Compute Engine в нескольких зонах.

SLA обычно включают несколько компонентов:

Целевые уровни обслуживания (SLO, Service Level Objective) определяют целевые метрики производительности. SLO может указывать, что 99,9% API-запросов должны выполняться в течение 200 миллисекунд.

Показатели уровня обслуживания (SLI, Service Level Indicator) это фактические измерения. Вы измеряете задержку запросов и сравниваете её с SLO.

Бюджеты ошибок определяют допустимое количество сбоев. Если ваш SLO составляет 99,9% доступности, ваш бюджет ошибок равен 0,1% около 43 минут в месяц допустимого простоя.

Штрафы или компенсации определяют, что происходит при нарушении SLA. Облачные провайдеры обычно предлагают сервисные кредиты процентный возврат при нарушении SLA. Однако эти кредиты редко компенсируют фактические бизнес-потери от сбоев.

Расчёт составной доступности

Реальные системы состоят из нескольких компонентов. Если ваше приложение зависит от базы данных и системы хранения, составная доступность является произведением доступностей отдельных компонентов. Если ваш веб-сервер имеет доступность 99,9%, а ваша база данных имеет доступность 99,9%, составная доступность равна 99,9% умноженное на 99,9%, что составляет 99,8% ниже, чем у любого отдельного компонента.

Эта математика объясняет, почему резервирование необходимо. Каждая добавленная зависимость снижает теоретическую доступность, если не реализовано резервирование.

Часть 2: Паттерны резервирования

Философия резервирования

Резервирование означает наличие запасных компонентов, которые могут принять на себя нагрузку при отказе основных компонентов. Принцип прост: устранить единые точки отказа. Любой компонент, отказ которого приводит к отказу системы, является единой точкой отказа и требует внимания.

Резервирование применяется на каждом уровне: аппаратном, программном, сетевом, на уровне центра обработки данных и географического региона. По-настоящему отказоустойчивая система имеет резервирование на нескольких уровнях одновременно.

Резервирование активный-пассивный

В конфигурациях активный-пассивный один компонент обрабатывает весь трафик, в то время как резервный компонент находится в режиме ожидания. При отказе активного компонента пассивный компонент активируется и принимает на себя нагрузку.

Этот паттерн распространён для баз данных. Основная база данных обрабатывает все операции чтения и записи. Резервная база данных поддерживает синхронизированную копию данных. При отказе основной базы резервная переходит в роль основной.

Преимущества конфигурации активный-пассивный включают простоту и более низкую стоимость, поскольку пассивный компонент может использовать меньше ресурсов. Это также позволяет избежать проблем согласованности, поскольку в каждый момент времени активен только один компонент.

Недостатки включают неиспользуемую мощность пассивного компонента и время переключения на резерв. Система недоступна в период переключения, который может составлять от секунд до минут в зависимости от реализации.

Примеры для баз данных включают репликацию MySQL с переключением на резерв, потоковую репликацию PostgreSQL и группы доступности SQL Server Always On в синхронном режиме.

Резервирование активный-активный

В конфигурациях активный-активный несколько компонентов обрабатывают трафик одновременно. Все компоненты полностью работоспособны и распределяют рабочую нагрузку между собой. При отказе одного остальные продолжают обработку трафика, возможно, с повышенной нагрузкой.

Конфигурация активный-активный распространена для веб-серверов и серверов приложений за балансировщиками нагрузки. Все серверы обрабатывают запросы; при отказе одного балансировщик нагрузки перенаправляет трафик на оставшиеся серверы.

Преимущества включают отсутствие неиспользуемой мощности, поскольку все компоненты работают в штатном режиме. Также отсутствует задержка при переключении на резерв система продолжает работу без перерыва при отказе компонента.

Недостатки включают повышенную сложность, особенно для приложений с сохранением состояния. Поддержание согласованности между активными компонентами требует тщательного проектирования. Базы данных в конфигурации активный-активный, например, должны обрабатывать конфликты записи, когда несколько узлов одновременно принимают операции записи.

Примеры включают кластеры веб-серверов с балансировкой нагрузки, конфигурации баз данных с несколькими мастерами, такие как Amazon Aurora или CockroachDB, и распределённые системы кэширования, такие как Redis Cluster.

Географическое резервирование

Географическое резервирование распределяет компоненты по различным физическим расположениям центрам обработки данных, зонам доступности или регионам. Это защищает от отказов, привязанных к конкретному расположению: отключений электроэнергии, стихийных бедствий, проблем с сетевой связью или даже атак, направленных на конкретные объекты.

Облачные провайдеры предлагают несколько уровней географического распределения:

Зоны доступности это отдельные центры обработки данных в пределах региона, соединённые каналами с низкой задержкой. Зоны доступности защищают от отказов отдельного центра обработки данных. Большинство приложений должны развёртываться в нескольких зонах доступности в пределах региона.

Регионы это географически разделённые расположения, обычно на расстоянии сотен километров друг от друга. Мультирегиональное развёртывание защищает от региональных катастроф или региональных облачных сбоев. Однако оно значительно усложняет обеспечение согласованности данных.

Мультиоблачное распределение между различными провайдерами например, AWS и Azure обеспечивает защиту от сбоев или проблем, специфичных для конкретного провайдера. Это добавляет значительную операционную сложность, но обеспечивает максимальную отказоустойчивость.

Принципы проектирования резервирования

При проектировании резервирования следуйте этим принципам:

Планируйте минимум N+1. Если вам необходимо N компонентов для обработки нагрузки, разверните N+1, чтобы один мог выйти из строя без последствий.

Рассмотрите N+2 для критических систем. Это позволяет одному компоненту выйти из строя, пока другой находится на обслуживании.

Избегайте общей судьбы. Резервные компоненты не должны совместно использовать инфраструктуру, которая может вызвать коррелированные отказы. Два сервера в одной стойке разделяют общую судьбу, если стойка теряет питание.

Тестируйте своё резервирование. Резервирование, которое никогда не тестировалось, может не сработать в нужный момент. Процедуры переключения на резерв должны регулярно отрабатываться.

Часть 3: Аварийное восстановление RTO и RPO

Определение аварийного восстановления

Аварийное восстановление (DR, Disaster Recovery) это процесс восстановления систем и данных после крупного нарушения. В то время как высокая доступность автоматически обрабатывает отказы компонентов, аварийное восстановление направлено на катастрофические события: разрушение центра обработки данных, атаки программ-вымогателей, крупные облачные сбои или повреждение основных систем.

Цель DR возобновить работу в приемлемые сроки и с допустимой потерей данных. Что считается "допустимым", зависит от организации и системы.

Целевое время восстановления (RTO)

Целевое время восстановления (RTO, Recovery Time Objective) определяет максимально допустимое время восстановления работоспособности после катастрофы. RTO отвечает на вопрос: как долго мы можем не работать?

RTO значительно варьируется в зависимости от критичности системы:

Для корпоративного сайта RTO может составлять 4-8 часов. Бизнес способен пережить день без маркетингового сайта.

Для платформы электронной коммерции RTO может составлять 1-2 часа. Каждый час простоя означает потерю дохода.

Для торговой системы RTO может составлять минуты. Длительные сбои влекут финансовые и регуляторные последствия.

Для систем обеспечения безопасности жизни RTO стремится к нулю. Системы экстренного реагирования не могут допустить никакого простоя.

RTO определяет выбор архитектуры DR. Достижение RTO в 4 часа несложно при наличии качественных резервных копий. Достижение RTO в 15 минут требует сложных конфигураций активный-активный или горячего резерва.

Целевая точка восстановления (RPO)

Целевая точка восстановления (RPO, Recovery Point Objective) определяет максимально допустимую потерю данных, измеряемую во времени. RPO отвечает на вопрос: сколько данных мы можем позволить себе потерять?

Если ваш RPO составляет 24 часа, вы можете восстановиться из ежедневных резервных копий, допуская потерю данных за один день. Если ваш RPO составляет 1 час, вам необходимы ежечасные резервные копии или непрерывная репликация. Если ваш RPO стремится к нулю, вам необходима синхронная репликация, при которой каждая запись подтверждается и на основной, и на DR-системе до получения подтверждения.

Как и RTO, RPO варьируется в зависимости от системы:

Для архивной системы RPO в 24 часа может быть приемлемым. Исторические данные изменяются нечасто.

Для транзакционной системы максимальный RPO может составлять 1 час. Потеря часа заказов или взаимодействий с клиентами имеет значительные последствия.

Для финансовой системы часто требуется нулевой RPO. Ни одна транзакция не может быть потеряна.

Стратегии DR

Различные стратегии обеспечивают различные комбинации RTO/RPO:

Резервное копирование и восстановление простейший и наиболее экономичный подход. Регулярное создание резервных копий, их хранение за пределами площадки и восстановление при необходимости. Это обеспечивает RTO от нескольких часов до нескольких дней и RPO, зависящий от частоты резервного копирования. Стоимость минимальна, но восстановление медленное.

Пилотный режим поддерживает минимальную DR-среду с работающими, но не обрабатывающими трафик основными системами. При наступлении катастрофы среда пилотного режима масштабируется до полной мощности. Это обеспечивает RTO от 1 до 4 часов и умеренный RPO. Стоимость умеренная.

Горячий резерв поддерживает уменьшенную, но полностью функциональную копию производственной среды. Трафик может быть перенаправлен на резерв с минимальными изменениями. Это обеспечивает RTO от минут до часа и RPO, стремящийся к нулю при наличии репликации. Стоимость значительная, но не чрезмерная.

Горячее дублирование или конфигурация активный-активный обеспечивает полную производственную мощность в нескольких расположениях одновременно. Восстановление не требуется трафик перенаправляется автоматически. Это обеспечивает RTO и RPO, стремящиеся к нулю. Стоимость наибольшая, поскольку поддерживается двойная мощность.

Соображения при планировании DR

При планировании DR учитывайте следующие факторы:

Документируйте зависимости. Ваше приложение может восстановиться быстро, но как насчёт системы аутентификации, от которой оно зависит? Составьте карту всех зависимостей и убедитесь, что планы DR охватывают каждую из них.

Планируйте частичные отказы. Что если откажут только некоторые системы? Можете ли вы работать в деградированном режиме, пока восстанавливаете отдельные компоненты?

Учитывайте человеческий фактор. Кто инициирует переключение на резерв? Как они узнают, когда действовать? Документированы ли процедуры и проведены ли учения?

Регулярно тестируйте. Нетестированный план DR это не план, это надежда. Назначайте регулярные тестирования DR и извлекайте уроки из каждого.

Часть 4: Планирование непрерывности бизнеса

За пределами технического восстановления

Планирование непрерывности бизнеса (BCP, Business Continuity Planning) выходит за рамки IT-систем и охватывает способность всей организации продолжать работу во время кризиса и после него. В то время как DR сосредоточено на технологиях, BCP охватывает людей, процессы, помещения и коммуникации.

Полноценная программа непрерывности бизнеса отвечает на вопрос: если мы не можем работать в штатном режиме, как мы продолжим обслуживать клиентов и выполнять обязательства?

Анализ воздействия на бизнес

Анализ воздействия на бизнес (BIA, Business Impact Analysis) является основой BCP. BIA определяет критические бизнес-функции и оценивает последствия их нарушения. Для каждой функции оцениваются:

Финансовое воздействие: какой доход теряется за каждый час нарушения? Какие дополнительные расходы возникают?

Операционное воздействие: какие процессы перестают работать? Какие последующие эффекты возникают?

Юридическое и регуляторное воздействие: какие требования соответствия нарушаются? Какие штрафы могут быть наложены?

Репутационное воздействие: как клиенты и партнёры воспринимают длительные сбои?

Результаты BIA определяют распределение ресурсов для планирования непрерывности. Критические функции с высоким воздействием за час получают наибольшие инвестиции в отказоустойчивость и возможности восстановления.

Стратегии непрерывности

BCP разрабатывает стратегии поддержания критических функций:

Альтернативные рабочие площадки позволяют продолжить работу при недоступности основных помещений. Это могут быть резервные офисы, возможность удалённой работы или соглашения с другими организациями о совместном использовании пространства.

Ручные обходные процедуры определяют, как выполнять критические функции без технологий. Если система заказов выходит из строя, могут ли сотрудники принимать заказы по телефону и вводить их позже? Ручные процедуры должны быть задокументированы и отработаны.

Соглашения с третьими сторонами могут позволить партнёрам или поставщикам выполнять функции во время нарушений. Конкурент может временно обрабатывать определённые услуги в рамках соглашений о взаимной помощи.

Планы коммуникации обеспечивают информирование заинтересованных сторон о происходящем. Сотрудникам нужны инструкции, клиентам обновления, а регуляторы могут потребовать уведомления.

Управление кризисами

Управление кризисами обеспечивает руководство и координацию во время крупных инцидентов. Группа управления кризисами, обычно включающая высшее руководство, принимает решения об объявлении катастроф, начале восстановления и взаимодействии с заинтересованными сторонами.

Процедуры управления кризисами определяют:

Критерии эскалации: какие условия инициируют кризисное реагирование?

Активация команды: как собираются члены команды, особенно в нерабочее время?

Полномочия на принятие решений: кто может утверждать расходы, публично общаться или задействовать контракты?

Протоколы коммуникации: как команда координируется и обменивается информацией?

Тестирование непрерывности бизнеса

Как и DR, BCP требует тестирования. Типы тестов включают:

Кабинетные учения это проработка сценариев в переговорной комнате. Команда обсуждает, что бы она делала в различных ситуациях. Это малозатратно и выявляет пробелы в планах и понимании.

Функциональные учения фактически выполняют конкретные процедуры без полного нарушения работы. Сотрудники могут работать из альтернативных расположений в течение дня, имитируя потерю помещения.

Полномасштабные учения имитируют реальные катастрофы максимально реалистично. Они дорогостоящи и нарушают работу, но выявляют слабые места, которые другие тесты пропускают.

Стандарты и фреймворки

ISO 22301 это международный стандарт систем менеджмента непрерывности бизнеса. Он предоставляет фреймворк для создания, внедрения, поддержания и совершенствования BCP. Сертификация демонстрирует приверженность организации непрерывности.

Business Continuity Institute и Disaster Recovery Institute International предлагают профессиональные сертификации и лучшие практики для специалистов.

Часть 5: Стратегии резервного копирования

Основа восстановления

Резервные копии это копии данных, хранящиеся отдельно от производственных систем. Когда основные данные теряются, повреждаются или шифруются

программами-вымогателями, резервные копии позволяют восстановление. Без надёжных резервных копий аварийное восстановление невозможно.

Тем не менее сбои резервного копирования остаются распространёнными. Согласно отчёту Veeam, разработчика решений резервного копирования, за 2024 год, 58% организаций сталкивались со сбоями резервного копирования, препятствовавшими полному восстановлению. Резервные копии, которые не проверяются, это не резервные копии, а предположения.

Правило резервного копирования 3-2-1

Правило 3-2-1 это базовая стратегия резервного копирования:

Три копии данных: оригинальные производственные данные плюс две резервные копии.

Два различных носителя хранения: различные типы хранилищ для предотвращения сбоев, специфичных для носителя.

Одна копия за пределами площадки: хранится в другом расположении для защиты от катастроф на уровне площадки.

Это правило эволюционировало для облачных сред. Расширение 3-2-1-1-0 добавляет:

Одна копия офлайн или изолирована воздушным зазором: отключена от сетей для предотвращения распространения программ-вымогателей.

Ноль ошибок: проверенные резервные копии с регулярным тестированием восстановления.

Типы резервного копирования

Полное резервное копирование копирует все данные. Его просто восстановить, но оно потребляет значительное время и место хранения. Частое выполнение полных резервных копий часто непрактично для больших наборов данных.

Инкрементное резервное копирование копирует только данные, изменившиеся с момента последнего резервного копирования любого типа. Оно быстрое и экономичное по хранению, но для восстановления требуется последняя полная резервная копия плюс все последующие инкрементные копии в последовательности.

Дифференциальное резервное копирование копирует данные, изменившиеся с момента последней полной резервной копии. Оно занимает больше времени и места, чем инкрементное, но упрощает восстановление: нужны только последняя полная резервная копия и последняя дифференциальная.

Синтетическое полное резервное копирование объединяет предыдущую полную резервную копию с последующими инкрементными для создания новой полной

копии без чтения всех производственных данных. Это сокращает окно резервного копирования, сохраняя простоту восстановления.

Неизменяемые резервные копии

Неизменяемые резервные копии не могут быть модифицированы или удалены в течение указанного периода хранения. Это защищает от программ-вымогателей, нацеленных на системы резервного копирования, а также от внутренних угроз или административных ошибок.

Варианты реализации включают:

Хранилище WORM (Write Once Read Many однократная запись, многократное чтение) физически предотвращает модификацию.

Блокировка объектов в облачном хранилище (S3 Object Lock, Azure Immutable Blob Storage) предотвращает удаление через API.

Хранилище с воздушным зазором физическое отключение от сетей исключает векторы удалённых атак.

Функции приложений резервного копирования, обеспечивающие неизменяемость через программные средства управления.

Неизменяемость стала обязательной. Современные программы-вымогатели целенаправленно атакуют системы резервного копирования для предотвращения восстановления. Ландшафт программ-вымогателей 2024-2025 годов показывает, что злоумышленники систематически уничтожают или шифруют резервные копии перед развёртыванием программ-вымогателей на производственных системах.

Проверка резервных копий

Резервные копии должны регулярно проверяться. Проверка включает:

Проверку целостности, подтверждающую, что данные резервной копии не повреждены. Большинство систем резервного копирования включают контрольные суммы или хеш-верификацию.

Тестирование восстановления фактическое восстановление данных из резервных копий. Необходимо регулярно восстанавливать файлы, базы данных и целые системы из резервных копий для проверки работоспособности процесса.

Тестирование времени восстановления измеряет, сколько времени фактически занимает восстановление. Ваши предположения о RTO должны подтверждаться тестированием.

Тестирование приложений подтверждает корректную работу восстановленных систем. База данных, восстановленная без ошибок, всё же может иметь проблемы на уровне приложения.

Особенности облачного резервного копирования

Облачные среды требуют адаптированных стратегий резервного копирования:

Нативные облачные сервисы резервного копирования (AWS Backup, Azure Backup) интегрируются с облачными ресурсами, но могут не обеспечивать полную защиту.

Кросс-региональная репликация защищает от региональных сбоев, но не защищает от логического повреждения, которое реплицируется.

Кросс-аккаунтное резервное копирование защищает от компрометации учётной записи путём хранения резервных копий в отдельных AWS-аккаунтах или подписках Azure.

Затраты на исходящий трафик могут быть значительными при восстановлении больших объёмов данных из облачного хранилища. Включите эти затраты в планирование DR.

Проверяйте SLA облачного провайдера по резервному копированию. Облачные провайдеры часто не выполняют резервное копирование ваших данных автоматически это ваша ответственность.

Часть 6: Защита от DDoS-атак

В лекции 7 мы подробно рассмотрели техники DDoS-атак, включая принципы работы SYN-флуда, атак с усилением и Slowloris на уровне протоколов. Здесь мы сосредоточимся на том, как защищаться от них с точки зрения отказоустойчивости инфраструктуры.

Кратко напомним: DDoS-атаки делятся на три категории:

- **Объёмные атаки** насыщают пропускную способность массивным трафиком (UDP-флуд, атаки с усилением). Измеряются в битах в секунду.
- **Протокольные атаки** исчерпывают ресурсы сервера или сетевого оборудования, эксплуатируя механизмы протоколов (SYN-флуд, атаки фрагментированными пакетами). Измеряются в пакетах в секунду.
- **Атаки на уровне приложений** отправляют запросы, выглядящие легитимными, но потребляющие непропорционально много ресурсов сервера (HTTP-флуд, Slowloris). Измеряются в запросах в секунду.

DDoS-атаки выросли в масштабах: крупнейшие теперь превышают 3 Тбит/с. Ни одна организация не может самостоятельно поглотить такие атаки без специализированной защиты.

Стратегии защиты от DDoS

Защита требует многоуровневого подхода:

Избыточное выделение ресурсов поддерживает запас мощности для поглощения атак. Это дорого и ограничено невозможно обеспечить ресурсы для атак в несколько терабит.

Ограничение частоты запросов ограничивает количество запросов от отдельных источников. Это помогает при небольших атаках, но не решает проблему распределённых атак с множеством источников, отправляющих небольшое количество запросов каждый.

Сети доставки контента (CDN) распределяют трафик по глобальным сетям. CDN поглощают объёмные атаки, распределяя нагрузку по своей масштабной инфраструктуре.

Сервисы защиты от DDoS от таких провайдеров, как Cloudflare, Akamai, AWS Shield и Azure DDoS Protection, анализируют трафик и фильтруют атаки до того, как они достигнут вашей инфраструктуры. Эти сервисы используют машинное обучение для различения атакующего трафика и легитимных пользователей.

Anycast-маршрутизация распределяет трафик по нескольким центрам обработки данных. Атаки рассредоточиваются по расположениям, а не концентрируются на одной цели.

Вышестоящая фильтрация работает с интернет-провайдерами и хостинг-провайдерами для фильтрации атакующего трафика до его попадания в вашу сеть.

Облачная защита от DDoS

Облачные провайдеры предлагают интегрированную защиту от DDoS:

AWS Shield Standard обеспечивает автоматическую защиту от распространённых объёмных и протокольных атак без дополнительной платы. Shield Advanced добавляет защиту на уровне приложений, выделенную поддержку и защиту от затрат.

Azure DDoS Protection обеспечивает защиту от объёмных и протокольных атак, интегрированную с сетевой инфраструктурой Azure.

Google Cloud Armor обеспечивает защиту от DDoS и возможности межсетевого экрана уровня приложений.

Эти сервисы необходимы для облачных развёртываний. Включите их и настройте соответствующим образом для ваших приложений.

Часть 7: Реагирование на инциденты в облачных средах

Особенности реагирования на инциденты в облаке

Реагирование на инциденты в облачных средах отличается от традиционного реагирования на собственной инфраструктуре. У вас нет физического доступа к серверам. Доказательства могут быть эфемерными завершённые экземпляры не оставляют образов дисков. Несколько сторон разделяют ответственность за обнаружение и реагирование.

Тем не менее основы остаются прежними: подготовка, обнаружение, анализ, локализация, устранение, восстановление и пост-инцидентная деятельность. Реализация меняется, но процесс остаётся неизменным.

Подготовка к облачным инцидентам

Подготовка ещё более критична в облачных средах:

Включите всестороннее журналирование. Облачные сервисы генерируют журналы, но вы должны их включить и настроить. AWS CloudTrail регистрирует вызовы API. Azure Activity Logs фиксируют операции с ресурсами. Google Cloud Audit Logs отслеживают административные действия.

Централизуйте журналы. Отправляйте журналы в единое расположение SIEM, выделенную учётную запись для журналирования или платформу управления журналами. Злоумышленники могут удалить журналы в скомпрометированных учётных записях; централизованное журналирование сохраняет доказательства.

Установите базовые показатели поведения. Знайте, как выглядит нормальная работа, чтобы можно было выявить аномалии. Используйте облачные инструменты, такие как AWS GuardDuty, Azure Defender или Google Cloud Security Command Center.

Определите процедуры реагирования на сценарии, специфичные для облака: скомпрометированные учётные данные IAM, открытые хранилища данных, криптомайнинг на скомпрометированных экземплярах.

Установите отношения с командами безопасности облачного провайдера. Крупные провайдеры имеют выделенные контакты по реагированию на инциденты для серьёзных случаев.

Обнаружение в облачных средах

Обнаружение опирается на несколько источников:

Облачные сервисы безопасности обеспечивают автоматическое обнаружение угроз. GuardDuty обнаруживает разведку, компрометацию экземпляров и компрометацию учётных записей. Azure Defender выявляет угрозы в сервисах Azure.

Корреляция в SIEM объединяет облачные журналы с другими источниками данных для выявления паттернов, указывающих на компрометацию.

Пользовательские правила обнаружения направлены на угрозы, специфичные для организации. Отслеживайте необычные вызовы API, неожиданный географический доступ или создание ресурсов в неиспользуемых регионах.

Оповещения о выставлении счетов могут выявить криптомайнинг неожиданные вычислительные расходы часто указывают на скомпрометированные экземпляры, используемые для майнинга криптовалюты.

Локализация и устранение

Облачные среды позволяют быструю локализацию:

Немедленно отзовите скомпрометированные учётные данные. IAM-пользователи, ключи доступа и ключи сервисных учётных записей могут быть отключены мгновенно.

Изолируйте скомпрометированные ресурсы с помощью групп безопасности или сетевых списков контроля доступа (ACL, Access Control List). Не завершайте работу экземпляров немедленно, если они нужны для криминалистического анализа.

Создайте моментальные снимки скомпрометированных экземпляров для криминалистического расследования перед завершением их работы. Вы можете анализировать моментальные снимки в изолированных средах.

Проверьте и отзовите доступ третьих сторон. Скомпрометированные учётные записи могли предоставить постоянный доступ через федерацию или межаккаунтные роли.

Восстановление в облаке

Восстановление использует возможности облака:

Переразверните из известных корректных шаблонов. Инфраструктура как код позволяет быстро воссоздать чистые среды.

Восстановите из проверенных резервных копий. Убедитесь, что резервные копии не затронуты той же компрометацией.

Внедрите дополнительные меры контроля для предотвращения повторения перед восстановлением сервиса.

Внимательно отслеживайте систему во время восстановления. Злоумышленники могли установить механизмы закрепления, которые переживают начальную очистку.

Пост-инцидентная деятельность

Облачные инциденты требуют тщательного пост-инцидентного анализа:

Сохраните все доказательства для возможных юридических или регуляторных разбирательств.

Проведите анализ первопричин. Как злоумышленники получили доступ? Какие меры контроля не сработали? Какие пробелы в обнаружении существовали?

Обновите процедуры и меры контроля на основе извлечённых уроков.

Поделитесь соответствующей информацией с облачными провайдерами. Они могут иметь дополнительные сведения или должны устранить проблемы, затрагивающие других клиентов.

Определите, требуется ли уведомление регуляторов. GDPR, NIS2 и другие нормативные акты требуют уведомления о нарушениях в определённые сроки.

Часть 8: Хаос-инженерия

Намеренные отказы

Хаос-инженерия это практика намеренного внесения отказов для проверки отказоустойчивости системы. Вместо того чтобы ждать, когда отказы произойдут, мы вызываем их преднамеренно в контролируемых условиях, чтобы узнать, как ведут себя наши системы и как их улучшить.

Netflix стал пионером хаос-инженерии с помощью Chaos Monkey инструмента, который случайным образом завершает работу производственных экземпляров. Если системы отказоустойчивы, пользователи ничего не замечают. Если системы выходят из строя, инженеры узнают, где необходимы улучшения.

Принципы хаос-инженерии

Принципы хаос-инженерии, опубликованные Netflix и сообществом хаос-инженерии, определяют эту практику:

Сформируйте гипотезу о нормальном состоянии системы. Определите, как выглядит нормальная работа успешные транзакции, время отклика, частота ошибок.

Варьируйте реальные события. Вносите отказы, которые действительно могут произойти: сбои серверов, сетевые разделения, внесение задержек, отказы зависимостей.

Проводите эксперименты в производственной среде. Имитируемые среды могут не выявить производственных проблем. Тщательно контролируемые производственные эксперименты дают наиболее реалистичные результаты.

Автоматизируйте эксперименты для непрерывного проведения. Отказоустойчивость должна проверяться регулярно, а не однократно.

Минимизируйте радиус поражения. Начинайте с небольших экспериментов и ограниченного масштаба. Расширяйте по мере роста уверенности.

Инструменты хаос-инженерии

Экосистема хаос-инженерии значительно развилась:

Chaos Monkey завершает работу случайных экземпляров в AWS. Часть Simian Army от Netflix.

Gremlin, разработчик платформы хаос-инженерии, предоставляет коммерческую платформу для хаос-экспериментов с различными типами инфраструктуры.

Litmus Chaos платформа хаос-инженерии с открытым исходным кодом для сред Kubernetes.

AWS Fault Injection Simulator предоставляет управляемую хаос-инженерию для ресурсов AWS.

Azure Chaos Studio предлагает аналогичные возможности для сред Azure.

Chaos Mesh облачно-нативная платформа хаос-инженерии для Kubernetes.

Типы хаос-экспериментов

Распространённые хаос-эксперименты включают:

Завершение работы экземпляра проверяет, как приложения обрабатывают потерю узлов.

Внесение сетевой задержки тестирует обработку таймаутов и пользовательский опыт в условиях медленного соединения.

Сетевое разделение имитирует сценарии расщепления мозга, когда компоненты не могут взаимодействовать.

Нагрузка на процессор и память тестирует поведение при исчерпании ресурсов.

Отказ зависимости имитирует недоступность баз данных, API или других зависимостей.

Отказ DNS тестирует поведение при сбое разрешения имён.

Смещение часов вносит временной дрейф для выявления ошибок в логике, зависящей от времени.

Внедрение хаос-инженерии

Начинайте хаос-инженерию постепенно:

Начните в предпроизводственной среде. Проводите хаос-эксперименты в промежуточных средах, пока не будете уверены в своих системах и процессах.

Начните с известной отказоустойчивости. Ваши первые эксперименты должны тестировать механизмы, которые, по вашему мнению, работают. Проверьте свои предположения, прежде чем исследовать неизвестные области.

Установите чёткие условия прекращения. Определите, что представляет собой неприемлемое воздействие, и обеспечьте возможность немедленной остановки экспериментов.

Широко информируйте. Команды должны знать, когда проводятся хаос-эксперименты. Неожиданности ведут к недоверию.

Документируйте и распространяйте результаты. Хаос-эксперименты генерируют ценные знания, которые должны приносить пользу всей организации.

Хаос-инженерия и культура отказоустойчивости

Хаос-инженерия это в такой же степени культура, как и технология. Она нормализует обсуждение отказов. Она смещает мышление от "выйдет ли это из строя?" к "когда это выйдет из строя, что произойдёт?". Она формирует уверенность на основе доказательств, а не надежд.

Организации, практикующие хаос-инженерию, сообщают о меньшем количестве производственных инцидентов, более быстром времени восстановления и повышенной уверенности в отказоустойчивости своих систем.

Часть 9: Практические упражнения и разбор случаев

Упражнение 1: Расчёт доступности

Рассчитайте составную доступность системы со следующими компонентами:

- Кластер веб-серверов: доступность 99,95%
- API-шлюз: доступность 99,99%
- База данных: доступность 99,9%

- Кэш: доступность 99,95%

Решение: 0,9995 x 0,9999 x 0,999 x 0,9995 = 0,9979 или доступность 99,79%

Эта система имеет менее трёх девяток доступности. Для улучшения необходимо добавить резервирование к компоненту с наименьшей доступностью (базе данных).

Упражнение 2: Анализ RTO/RPO

Для каждой системы ниже предложите подходящие значения RTO и RPO и обоснуйте их:

1. Публичный маркетинговый сайт
2. Внутренняя система управления персоналом
3. Система обработки платежей в реальном времени
4. Система тикетов службы поддержки клиентов

Учитывайте бизнес-воздействие, регуляторные требования и техническую осуществимость.

Упражнение 3: Проектирование стратегии резервного копирования

Спроектируйте стратегию резервного копирования для системы электронных медицинских карт медицинской организации. Учтите:

- Регуляторные требования (HIPAA)
- Чувствительность данных
- Требования к восстановлению
- Защита от программ-вымогателей

Разбор случая: Инцидент с базой данных GitLab (2017)

В январе 2017 года GitLab столкнулся с инцидентом с базой данных, который привёл к потере производственных данных. Инженер случайно удалил производственную базу данных во время устранения неполадок. Когда попытались восстановить данные, обнаружилось, что несколько систем резервного копирования не работали:

- Регулярные резервные копии не завершались успешно
- Реплики значительно отставали от основной базы
- Восстановление на момент времени не было настроено
- Проверка резервных копий не проводилась

GitLab потерял примерно 6 часов данных и испытал 18 часов простоя.

Извлечённые уроки:

- Проверяйте резервные копии через регулярное тестирование восстановления
- Активно мониторьте системы резервного копирования
- Не полагайтесь на единственный метод резервного копирования
- Документируйте и отрабатывайте процедуры восстановления

GitLab достойно отреагировал, публично задокументировав инцидент и реализовав комплексные улучшения.

Разбор случая: Сбой AWS US-EAST-1 (2017)

В феврале 2017 года опечатка в скрипте обслуживания вывела из строя значительную часть S3 в регионе US-EAST-1. Каскадные эффекты были огромными, поскольку множество сервисов зависели от S3, включая собственную панель статуса AWS.

Извлечённые уроки:

- Зависимости от общих сервисов могут вызывать коррелированные отказы
- Панели мониторинга статуса не должны зависеть от систем, которые они мониторят
- Мультирегиональные архитектуры обеспечивают защиту от региональных проблем
- Даже облачные гиганты могут столкнуться с крупными сбоями

Заключение

Позвольте подвести итоги ключевых концепций сегодняшней лекции об отказоустойчивости инфраструктуры.

Мы начали с концепций доступности и SLA. Доступность измеряется в девятках, и каждая дополнительная девятка значительно сокращает допустимое время простоя и увеличивает стоимость. SLA определяют ожидания через SLO, SLI и бюджеты ошибок. Составная доступность является произведением доступностей компонентов.

Паттерны резервирования устраняют единые точки отказа. Конфигурация активный-пассивный обеспечивает простое переключение на резерв с некоторым простоем. Конфигурация активный-активный поддерживает непрерывную работу при отказах. Географическое резервирование защищает от событий, привязанных к конкретному расположению.

Планирование аварийного восстановления определяет, как восстановить работу после катастрофических событий. RTO определяет максимально допустимое время простоя. RPO определяет максимально допустимую потерю данных. Различные стратегии DR обеспечивают различные комбинации RTO/RPO при различных затратах.

Планирование непрерывности бизнеса выходит за рамки IT и охватывает всю организацию. Анализ воздействия на бизнес определяет критические функции. Стратегии непрерывности поддерживают работу альтернативными средствами. Управление кризисами обеспечивает руководство во время инцидентов.

Стратегии резервного копирования следуют правилу 3-2-1-1-0. Неизменяемые резервные копии защищают от программ-вымогателей. Проверка через регулярное тестирование восстановления обязательна.

DDoS-атаки бывают объёмными, протокольными и на уровне приложений. Защита требует многоуровневых мер, включая CDN, облачную защиту и специализированные сервисы.

Реагирование на инциденты в облачных средах требует подготовки через журналирование, обнаружения через облачные сервисы безопасности и использования облачных возможностей для быстрой локализации и восстановления.

Хаос-инженерия намеренно вносит отказы для формирования уверенности в отказоустойчивости. Она нормализует обсуждение отказов и создаёт культуру непрерывного совершенствования.

Для подготовки изучите план DR вашей организации или знакомой организации. Рассчитайте составную доступность системы, которой вы пользуетесь. Подумайте, как принципы хаос-инженерии могут быть применены к системам, с которыми вы работаете.

На следующей лекции мы рассмотрим управление рисками безопасности как систематически выявлять, оценивать и обрабатывать риски безопасности.

Вопросы для обсуждения

1. Как организациям следует определять соответствующий уровень инвестиций в доступность для различных систем?
2. Какие уроки можно извлечь из крупных сбоев облачных провайдеров о планировании отказоустойчивости инфраструктуры?
3. Как можно безопасно внедрить практики хаос-инженерии в организациях, избегающих рисков?
4. Система резервного копирования компании никогда не тестировалась. Во время атаки программы-вымогателя обнаруживается, что резервные копии

повреждены. Как можно было это предотвратить?

5. Всегда ли необходима доступность 99,99%? Приведите примеры систем, для которых допустима более низкая доступность, и объясните почему.

Благодарю за внимание. Есть ли вопросы?

Контрольные вопросы

1. Объясните концепцию "девяток доступности" и рассчитайте годовое время простоя для доступности 99,99%.
2. Сравните паттерны резервирования активный-пассивный и активный-активный. Каковы компромиссы каждого?
3. Определите RTO (Recovery Time Objective, целевое время восстановления) и RPO (Recovery Point Objective, целевая точка восстановления). Как они влияют на выбор стратегии аварийного восстановления?
4. Что такое правило резервного копирования 3-2-1-1-0, и почему оно было расширено по сравнению с первоначальным правилом 3-2-1?
5. Что такое объёмные DDoS (Distributed Denial of Service, распределённый отказ в обслуживании)-атаки и как CDN (Content Delivery Network, сеть доставки контента) и центры очистки трафика защищают от них?
6. Что такое протокольные DDoS-атаки, такие как SYN-флуд, и как от них защищаются?
7. Что такое DDoS-атаки на уровне приложений и как межсетевые экраны уровня приложений WAF (Web Application Firewall, межсетевой экран веб-приложений) и ограничение частоты запросов помогают защищаться от них?
8. Что такое анализ воздействия на бизнес, и как он определяет планирование непрерывности бизнеса?
9. Объясните принципы хаос-инженерии и опишите три типа хаос-экспериментов.
10. Какие уроки преподносит инцидент с базой данных GitLab 2017 года о проверке резервных копий?

Ключевые термины

- **Активный-активный**: паттерн резервирования, при котором все узлы одновременно обрабатывают трафик
- **Активный-пассивный**: паттерн резервирования, при котором резервные узлы принимают на себя нагрузку при отказе основных узлов

- **BCP**: план обеспечения непрерывности бизнеса, обеспечивающий продолжение критических функций во время и после катастрофы
- **BIA**: анализ воздействия на бизнес, определяющий критические бизнес-функции и их зависимости
- **CDN**: сеть доставки контента, распределённая система доставки контента с улучшенной производительностью и защитой от DDoS
- **Хаос-инженерия**: практика намеренного внесения отказов для проверки отказоустойчивости системы
- **DDoS**: распределённая атака типа "отказ в обслуживании", атака, перегружающая системы трафиком из множества источников
- **DR**: аварийное восстановление, восстановление IT-систем и данных после катастрофического события
- **Бюджет ошибок**: допустимый объём простоя или ошибок в рамках периода SLA
- **Высокая доступность**: проектирование системы, обеспечивающее заданный уровень непрерывности работы с минимальным простоем
- **Неизменяемая резервная копия**: резервная копия, которую нельзя изменить или удалить после создания
- **Балансировка нагрузки**: распределение сетевого трафика между несколькими серверами для обеспечения надёжности и производительности
- **RPO**: целевая точка восстановления, максимально допустимый объём потери данных, измеряемый во времени
- **RTO**: целевое время восстановления, максимально допустимая длительность простоя системы
- **SLA**: соглашение об уровне обслуживания, формальное обязательство по стандартам производительности
- **SLI**: показатель уровня обслуживания, количественная мера производительности сервиса
- **SLO**: целевой уровень обслуживания, целевое значение показателя уровня обслуживания