**Глава 15. Отладка**

Отладка — это ключевой процесс в решении технических проблем. В отличие от простых изменений, которые могут временно устранить проблему (например, перезагрузка), отладка включает в себя более глубокий анализ проблемы, выявление её причины и затем принятие решения, которое устраняет саму причину, а не просто маскирует симптомы. Понимание этого процесса позволяет избежать временных исправлений, которые только откладывают более сложные задачи.

Отладка начинается с точного определения проблемы, затем следует поиск её корня, и только после этого можно вносить изменения, чтобы устранить неисправность. Процесс может потребовать повторных попыток, тестирования разных гипотез, что требует терпения и опыта. Это не просто механическое исправление, а последовательный и обоснованный подход к решению задач.

**15.1. Основы отладки**

В этой части мы рассмотрим несколько советов по правильному определению и решению проблемы, а также модели поиска проблемы.

1. **Точное определение проблемы**  
   Прежде чем приступить к устранению неисправности, важно точно понять, что именно не работает. Нужно задать правильные вопросы и собрать всю необходимую информацию о симптомах. Поняв проблему, можно значительно упростить процесс её решения.
2. **Модели поиска проблемы**  
   Существует несколько моделей, которые помогают систематизировать поиск неисправностей. Например:
   * **Метод исключений**: Сначала проверяются наиболее вероятные причины проблемы, и по мере исключения вариантов сужается круг поиска.
   * **Дерево решений**: Этот подход помогает разбить проблему на подзадачи и проверить их поочередно, начиная с наиболее вероятных причин.
3. **Качества лучших средств для отладки**  
   Важно не только понять, как искать проблемы, но и иметь подходящие инструменты для этого. Хорошие средства отладки должны быть:
   * **Простыми в использовании**: Инструменты должны быть интуитивно понятными и не создавать дополнительных трудностей.
   * **Мощными**: Они должны иметь возможности для детального анализа проблемы.
   * **Интегрированными**: Инструменты должны работать в рамках вашего рабочего процесса, интегрируясь с другими системами и инструментами, которые вы используете.

Таким образом, отладка требует не только технической компетенции, но и системного подхода, терпения и подходящих инструментов.

**15.1.1. Ознакомьтесь с проблемой пользователя**

Первый шаг в решении проблемы — это понять, что именно происходит с точки зрения пользователя. Необходимо разобраться, какую задачу пользователь пытается выполнить и какие результаты он ожидает, чтобы выявить отклонения от нормы. Это может быть не так просто, поскольку пользователь может описать проблему неполно или некорректно.

Пример: если пользователь не может прочитать свою электронную почту, он может сказать, что его почтовая программа не работает, или что он не может подключиться к серверу. Однако, на самом деле, проблема может быть связана с неполадками сети, настройками DNS, проблемами с электропитанием в серверной или с другим элементом инфраструктуры.

При взаимодействии с пользователем важно проявлять терпение и пытаться понять, что именно он делает. Если пользователь использует технические термины или жаргон, не всегда стоит полагаться на эти выражения. В таких случаях можно попросить пользователя просто рассказать, что он пытался сделать, избегая сложных технических понятий.

Пример жалобы: «Мой принтер сломан» может оказаться не совсем точным, если проблема не в самом принтере, а в том, что пользователь не может напечатать документ из-за других причин (например, настройки сети или проблема с драйверами). Важно разобраться, что именно требуется пользователю, прежде чем принять решение.

Некоторые пользователи могут заранее анализировать проблему и даже предложить возможное решение, например: «Я думаю, проблема в DNS». Но важно помнить, что такие гипотезы могут быть ошибочными, и всегда нужно проверить все возможные варианты, чтобы точно понять, в чём состоит проблема.

**15.1.2. Устраняйте причину, а не симптом**

Для долгосрочной надежности систем необходимо не только устранять симптомы, но и выявлять и устранять основную причину проблемы. Временные решения могут быть полезны в краткосрочной перспективе, но для предотвращения повторных ошибок важно найти и устранить первопричину.

Пример: когда система зависает и требуется перезагрузка, это может быть временным решением. Но важно понять, почему система зависла — возможно, это связано с ошибками в драйверах, утечками памяти или другими внутренними проблемами. Если перезагрузка является единственным решением, то это всего лишь обход, а не полноценное устранение причины.

Тот же принцип касается и других проблем, например, нехватки места на диске. Удаление старых логов может освободить пространство, но проблема вернется, когда логи снова начнут накапливаться. Вместо этого можно настроить автоматическое удаление или архивирование логов, чтобы проблема не повторялась.

Даже крупные компании, такие как Microsoft, сталкиваются с ситуацией, когда их решения устраняют симптомы, но не решают основные проблемы. Например, ускоренная перезагрузка Windows 2000 была воспринята как улучшение, но многие пользователи предпочли бы, чтобы система не требовала перезагрузки так часто.

**15.1.3. Подходите системно**

Для эффективной отладки важно подходить к процессу методически и системно. Это означает, что нужно строить гипотезы, проверять их, записывать результаты и вносить изменения на основе этих данных. Случайные изменения, которые не основываются на логическом подходе, могут не решить проблему и только усложнят её.

Отладка часто включает в себя два основных метода: **исключение** и **последовательное уточнение**.

1. **Метод исключения** предполагает удаление или отключение отдельных компонентов системы, пока не исчезнет проблема. Например, заменяются платы памяти, пока не исчезает ошибка, или отключаются приложения, пока не решится ошибка.
2. **Последовательное уточнение** — это добавление компонентов системы по одному и проверка их работы на каждом этапе. Пример: при диагностике проблем с маршрутизацией вы начинаете с проверки первого сетевого перехода, затем второго, третьего и так далее, пока не найдете узкое место.

Иногда для упрощения процесса можно пропустить этапы, основываясь на предыдущем опыте, но это может привести к ошибкам, особенно если вы не проверяете каждую часть системы. Всегда помните, что последнее изменение в системе или сети может быть причиной проблемы, и стоит проанализировать все недавние изменения перед началом отладки.

Методика последовательного уточнения также применима при отладке сетевых проблем или отслеживании сообщений, проходящих через несколько серверов. Инструменты, такие как tcpdump (для UNIX) или Ethereal (для Windows), позволяют отслеживать пакеты и анализировать их на каждом этапе, что помогает выявить проблемные узлы или сегменты сети.

15.1.4. **Пользуйтесь правильными средствами**

Отладка требует использования соответствующих инструментов диагностики. Эти инструменты могут быть как физическими устройствами, так и программным обеспечением, которое можно приобрести, загрузить или разработать самостоятельно. Однако самым важным инструментом остаются знания.

Средства диагностики позволяют увидеть внутренние процессы работы системы или устройства, но без понимания того, что именно вы видите, информация будет бесполезной. Обучение не только включает использование инструментов, но и понимание их работы. Например, чтобы эффективно работать с Ethernet-анализатором (сниффером), важно не только уметь перехватывать пакеты, но и понимать, как работают различные протоколы.

UNIX-системы славятся своей простотой в отладке, поскольку многие системные администраторы обладают глубокими знаниями об их внутренних процессах. Эти системы поставляются с документацией, а также имеют исходный код, что облегчает понимание работы системы. Программисты и администраторы могут анализировать исходный код для лучшего понимания процессов.

С другой стороны, Microsoft Windows изначально имела репутацию "черного ящика", в котором было трудно получить информацию для отладки. Однако с течением времени и с развитием образовательных программ, разработанных для Windows, стало доступно больше информации для устранения проблем.

Когда речь идет о выборе инструментов для отладки, предпочтение часто отдается более простым и компактным инструментам. Меньше инструментов, меньше сложностей. Простые инструменты часто оказываются более эффективными для диагностики проблем, чем громоздкие и сложные решения.

Пример: проблемы с подключением разделов NFS можно решить с помощью простых инструментов: ping, traceroute и rpcinfo. Каждый инструмент выполняет одну функцию, но делает это хорошо. Например, ping проверяет, проходят ли пакеты между клиентом и сервером, traceroute помогает локализовать проблемы в сети, а rpcinfo проверяет протоколы NFS.

Если все эти средства работают, можно локализовать проблему и решить её. Таким образом, часто лучшие средства — это простые утилиты, которые можно легко комбинировать для диагностики и устранения проблемы.

15.2. **Тонкости**

Тонкости отладки заключаются в реальном развитии основ: в использовании лучших средств, углубленном понимании их применения и освоении более сложных аспектов отладки системы.

**15.2.1. Лучшие средства отладки**

Лучшие средства действительно важны! Всегда появляются новые инструменты, которые могут быть лучше старых. Однако быть в курсе всех новинок — задача непростая, и легко можно пропустить возможность первыми применить новую технологию. Чтобы не упустить обновления, полезно следить за форумами, такими как конференции USENIX и SAGE, а также за специализированными веб-сайтами и рассылками.

Важно помнить, что мы поддерживаем простоту инструментов. Улучшение не должно означать усложнение: иногда лучшая новинка — это инструмент, который упрощает процесс. При выборе новых средств важно оценивать их в контексте реальных проблем, которые они решают, а не только по тому, как они выглядят на маркетинговых материалах. Стоит избегать поверхностных «модных слов» и задавать вопрос: решает ли этот инструмент реальную проблему?

Перед покупкой всегда запрашивайте пробную версию и протестируйте продукт в действии. Если инструмент не оправдает ожиданий — не бойтесь отказаться от него. Ваша обратная связь поможет улучшить продукт.

**15.2.2. Формальное обучение работе со средствами отладки**

Формальное обучение отладке имеет ряд преимуществ. Несмотря на то, что инструкции могут быть полезными, обучение может дать важные знания, которые отделяют вас от других специалистов.

**Преимущества формального обучения:**

* Обычно оно проводится вне рабочего места, что позволяет избежать постоянных отвлечений и сосредоточиться на освоении новых навыков.
* Обучение охватывает все возможности средств, а не только те, которые вы успели освоить самостоятельно.
* Преподаватели могут выявить ошибки и аспекты работы инструментов, которые разработчики могли не включить в документацию.
* Часто обучение включает практическую работу в лабораториях, где можно провести эксперименты и проверить теории, которые трудно реализовать в обычных рабочих условиях.
* По завершению обучения можно получить сертификат, который будет важным дополнением к вашему резюме и повысит интерес со стороны работодателей, особенно если у вас нет большого практического опыта.

**15.2.3. Понимание системы от начала до конца**

На высшем уровне отладки системы необходим хотя бы один человек, который понимает, как работает вся система, от начала до конца. Это может быть легко в небольших системах, но с ростом и усложнением системы люди начинают специализироваться, и каждый из них знает только свою часть. В случае крупной аварии наличие человека, который видит систему в целом, является ценным активом. В экстренной ситуации лучше всего собрать команду специалистов, каждый из которых будет отвечать за свой уровень иерархию системы.

### 15.3. ****Заключение****

Каждый системный администратор регулярно сталкивается с задачей отладки и создает для себя мысленный список стандартных решений для распространенных проблем. Однако, чтобы эффективно решать проблемы, отладка должна быть системным, методическим процессом. Это процесс, основанный на знании целей и задач пользователя, а также направленный на устранение первопричины проблемы, а не только ее симптомов. Отладка может быть как субстрактивной (процесс исключения), так и аддитивной (поэтапное уточнение).

**Почему важно устранять первопричину:** Если не устранить первопричину проблемы, она обязательно вернется. Это создаст дополнительную нагрузку на администратора и увеличит объем работы в будущем.

**Метод решения проблем:** Хотя в этой главе обсуждаются способы быстрого решения проблем, иногда необходимо принять решение об обходном пути. В некоторых случаях временное исправление, которое устраняет симптомы проблемы, может быть выполнено в рабочее время, а более глобальные и постоянные решения могут быть отложены до запланированных технических интервалов (например, в ходе техобслуживания).

**Роль лучших средств:** Хорошие средства помогают решать проблемы более эффективно, избегая при этом чрезмерной сложности. Формальное обучение использованию этих средств даст вам знания и навыки, которые нельзя получить только из руководств. В случае серьезных сбоев или уникальных проблем ничто не заменит наличия человека или нескольких людей, которые понимают систему от начала до конца.

**Простота и эффективность инструментов:** Простые инструменты способны эффективно решать серьезные проблемы. В то время как сложные инструменты могут скрывать детали принятия решений, что затрудняет диагностику.

**Коммуникация с пользователями:** Отладка часто включает активное взаимодействие с пользователями. Для успешного решения проблемы важно понять, что пользователь пытается сделать, и как проблема проявляется в его работе.

Отладка — это не только технический процесс, но и взаимодействие с пользователями для точного понимания проблемы и поиска эффективных решений.

### 16. Однократное устранение проблем

**Исправить что-то один раз лучше, чем исправлять это постоянно.** Этот принцип кажется очевидным, но на практике его реализация может столкнуться с различными препятствиями. Иногда решение проблемы на скорую руку кажется более удобным, а другие ограничения могут мешать найти окончательное решение. Тем не менее, подход "исправить один раз" помогает более эффективно планировать время и повышать качество работы системного администратора. Это также помогает лучше объяснить пользователю, почему решение проблемы заняло больше времени, чем ожидалось.

#### 16.1. **Основы**

**Принцип «исправляйте один раз»** — это наш основной подход. Если что-то нарушено, это должно быть исправлено так, чтобы проблема не возникала снова. Если ошибка может повториться на других машинах, ее следует искать и устранять и там.

##### 16.1.1. **Не тратьте время зря**

Иногда, особенно с проблемами, которые кажутся тривиальными или случаются только время от времени, может быть соблазнительно временно решить проблему, не устраняя ее окончательно. Это решение может показаться легче в моменте, но вы можете не осознавать, что каждый раз решаете одну и ту же проблему. Это приводит к лишним затратам времени и усилий. Лучше потратить чуть больше времени на окончательное решение проблемы, чтобы в будущем не возвращаться к ней снова и снова.

Этот подход помогает избежать циклических исправлений и повысить эффективность работы.

### 16.1.2. ****Избегайте временных решений****

Хотя временные решения могут быть оправданы в экстренных ситуациях, их не следует рассматривать как долгосрочные. Иногда ограничения по времени или ресурсам требуют быстрого исправления проблемы, но важно помнить, что за временными мерами должны следовать постоянные решения.

Временные решения могут быть соблазнительными, так как они кажутся быстрыми и эффективными, но в конечном итоге они часто не решают основную проблему. Например, если диск заполняется логами, временное решение может заключаться в удалении старых логов вручную, но это только отодвигает необходимость установки автоматической программы для их обновления. Вместо того чтобы постоянно решать проблему вручную, стоит подумать о долгосрочном решении.

**Записывайте действия:** Чтобы избежать забывания о необходимости вернуться к решению проблемы, важно записывать сделанные временные исправления. Это можно сделать с помощью напоминаний, например, через электронную почту или специальные приложения для создания заявок. В системе UNIX можно настроить отправку напоминаний из командной строки, чтобы не полагаться на почтовый клиент, что позволяет работать с напоминаниями прямо в процессе выполнения задач.

**Психологическая ловушка временных решений:** Временные исправления часто кажутся проще и быстрее, что делает их привлекательными, особенно в условиях стресса или ограниченности ресурсов. Но такие решения могут привести к тому, что мы будем выполнять одни и те же действия снова и снова, забывая о необходимости внедрить более устойчивые решения. Мы чувствуем себя занятыми, но на самом деле не достигаем долгосрочного прогресса.

**Как разорвать замкнутый круг?** Важно осознать, что решение проблемы раз и навсегда, хотя и требует больших усилий, может сэкономить намного больше времени в будущем. Иногда это означает, что необходимо внести изменения в конфигурацию системы, например, установить новый сервис или настроить автозагрузку скриптов. Вместо того чтобы каждый раз вручную вводить команды после перезагрузки, нужно добавить их в скрипты автозагрузки, что избавит от необходимости повторять одно и то же снова и снова.

Нужно быть внимательным, чтобы избежать ошибок в таких скриптах, так как неправильные настройки могут привести к тому, что система не загрузится. Поэтому важно проверять такие изменения, перезагружая машину сразу после внесения изменений, чтобы убедиться в правильности настройки.

Таким образом, временные меры должны быть только промежуточными шагами на пути к более постоянным и долговечным решениям, и важно помнить, что не стоит застревать в цикле исправлений, который не решает основную проблему.

### 16.1.3. ****Учитесь у плотников****

Системные администраторы могут многому научиться у плотников, которые строят и ремонтируют уже много лет. Плотники придерживаются принципа: «Семь раз отмерь, один отрежь». Это означает, что важно дважды проверять свои действия, чтобы избежать ошибок. В аналогии с системным администрированием, это может быть связано с проверкой команд перед их выполнением, повторной проверкой конфигурационных файлов и запросом мнений коллег.

**Повторное измерение и копирование:** Плотники также знают, что если нужно сделать несколько одинаковых предметов, то лучший способ — это использовать первый сделанный объект как шаблон для всех последующих. Это предотвращает накопление ошибок, которые могут возникнуть, если каждый новый элемент будет измеряться относительно предыдущего. Этот подход можно применить и в администрировании, например, при настройке нескольких серверов, где лучше всего настроить один сервер правильно, а затем использовать его конфигурацию как основу для других.

**Привычка к проверке:** Регулярная проверка работы важна для предотвращения ошибок. Проверяйте свою работу дважды, перед тем как внести изменения в систему. Например, перечитайте файл конфигурации перед его сохранением, или попросите коллегу взглянуть на команду перед её выполнением.

### 16.2. ****Тонкости****

В этом разделе рассматриваются вопросы автоматизации устранения проблем. Автоматизация может помочь как в устранении симптомов, так и в реализации постоянных решений.

**Автоматизация устранения симптомов:** Важно, чтобы автоматизация не только решала проблему, но и информировала администратора о том, что это временное решение. Например, автоматическое удаление старых логов может освободить место на диске, но это не устраняет основную причину заполнения диска. Если автоматизация сообщает о своем действии, администратор может своевременно принять меры для долгосрочного решения.

**Риски автоматизации:** Когда автоматизация используется для устранения симптомов, важно избегать ситуации, когда она становится слишком привычной и перестает привлекать внимание к корневым проблемам. В таком случае важно создать механизмы, которые бы гарантировали принятие постоянных мер и не позволяли игнорировать предупреждения о временных решениях.

**Автоматизация для постоянных решений:** В некоторых случаях автоматизация может исправить проблему полностью. Например, системы виртуальных машин могут автоматически выделить дополнительные процессоры для нагрузочных вычислений, или файловая система может автоматически добавить новый диск в раздел. Это помогает решить проблему, но важно понимать, что такие меры не заменяют правильного планирования.

**Сложности и выгоды автоматизации:** Создание автоматизированных решений может занять время, но в перспективе это значительно сэкономит время и усилия, избавив от выполнения рутинных задач вручную. Например, можно создать скрипт для выполнения нескольких задач, что в долгосрочной перспективе поможет сэкономить часы работы.

### 16.3. ****Заключение****

Лучше устранить проблему раз и навсегда, чем постоянно её исправлять. Важно избегать временных решений и вместо этого сосредоточиться на поиске постоянных мер. Использование проверенных решений, правильное планирование и регулярная проверка работы — это ключевые моменты для эффективного системного администрирования. Иногда ресурсы могут быть ограничены, и в таких случаях необходимо принимать временные меры, но важно не забывать о долгосрочном решении.

Иногда автоматизация может существенно упростить процесс, но важно помнить о её недостатках. Хорошая автоматизация может существенно снизить нагрузку на администраторов, но требует времени для разработки и внедрения.

**Глава 17: Управление изменениями**

Управление изменениями — это процесс, который позволяет эффективно планировать, реализовывать и анализировать изменения в системе. Он включает в себя документацию изменений, план отмены и возможность воспроизводства. Главная цель управления изменениями — это управление рисками, возникающими из-за изменений в системе, что может привести к сбоям в обслуживании пользователей. Основной аспект управления изменениями — это оценка этих рисков и их управление через стратегии смягчения. Это процесс, который помогает обеспечить минимальное влияние изменений, снижая вероятность ошибок и ускоряя их исправление.

**Основные компоненты управления изменениями:**

1. **Общение и составление графика:** Важно обеспечить хорошую коммуникацию с пользователями и другими системными администраторами, чтобы все знали о предстоящих изменениях, а также составить график изменений, минимизируя влияние на систему.
2. **Планирование и проверка:** Нужно планировать, как и когда вносить изменения, а также разработать планы для проверки работоспособности системы после изменения и план отмены, если возникнут проблемы.
3. **Процессы и документация:** Все изменения должны соответствовать установленным процессам, быть хорошо спланированными и задокументированными, а также одобренными перед их реализацией.
4. **Контроль версий и автоматизация:** Для отслеживания изменений и упрощения отката проблемных обновлений рекомендуется использовать систему контроля версий. Автоматизация изменений помогает обеспечить точное и воспроизводимое выполнение всех процессов.

Управление изменениями также учитывает различные категории систем, типы изменений и специальные процедуры для каждой комбинации. Например, обновление директорий, установка нового ПО, модернизация существующего ПО, изменения в политике безопасности и конфигурации могут требовать различных уровней контроля.

**Риски и процессы управления изменениями:** Малые изменения могут быть исключены из строгих процессов управления изменениями, чтобы не замедлять работу системных администраторов. Однако значительные изменения, которые могут повлиять на критические для бизнеса системы, должны проходить полный процесс управления изменениями, что включает обязательное общение с заинтересованными сторонами и планирование на приемлемое время.

**ITIL и стандарты управления изменениями:** Для более глубокой проработки управления изменениями можно обратиться к ITIL (Information Technology Infrastructure Library), который является ценным ресурсом для изучения области управления изменениями и процессов системного администрирования. Стандарты ITIL становятся общепринятыми и широко применяемыми в разных организациях.

### 17.1.1. ****Управление риском****

Управление риском — важный аспект деятельности системного администратора. Основной риски, с которыми сталкиваются администраторы, связаны с невыполненным обслуживанием и потерей данных. Для управления риском важнейшим инструментом являются резервные копии, которые помогают предотвратить потерю данных и обеспечить непрерывность обслуживания.

#### Этапы управления риском:

1. **Выявление и оценка рисков:** Оцените, какие системы и службы могут быть затронуты изменениями. Какие возможны наихудшие сценарии, и как эти изменения повлияют на пользователей?
2. **Смягчение риска:** После оценки риска необходимо разработать план смягчения, который включает:
   * **Консультации по изменению:** Как изменение влияет на требования бизнеса и другие события?
   * **План тестирования:** Как проверить, было ли изменение успешным?
   * **План отмены:** Как вернуть систему в рабочее состояние, если изменение не удастся?
   * **Момент принятия решения:** Когда и как принять решение о необходимости отмены изменения?
   * **Подготовка:** Что можно сделать заранее, чтобы изменение прошло быстро и эффективно?

Важно заранее определить, при каких условиях обновление системы будет остановлено, чтобы обеспечить время для реализации плана отмены и восстановления системы.

#### Проблема принятия решения:

Момент принятия решения об откате изменений может быть сложным для администратора. Часто бывает искушение продолжить работу "еще немного". В этом случае полезно, чтобы решение было поддержано другим системным администратором или руководителем, чтобы гарантировать своевременное исполнение плана отмены.

В идеале изменения должны быть протестированы заранее в лаборатории или на дополнительной машине. Однако не все компании могут позволить себе такие ресурсы, и некоторые изменения невозможно протестировать в тестовой среде.

### 17.1.2. ****Структура распространения информации****

Эффективное распространение информации о предстоящих изменениях важно для системных администраторов и пользователей.

#### Два аспекта распространения информации:

1. **Информирование среди администраторов:** Все члены команды должны быть осведомлены о предстоящих изменениях, чтобы они могли быть готовы к возможным проблемам. Это позволит быстрее выявлять и устранять проблемы.
2. **Информирование пользователей:** Пользователи должны быть заранее уведомлены о крупных изменениях, чтобы подготовиться к переходу на новые версии сервисов или программ. Важно четко объяснить, когда старая версия будет недоступна, как пользоваться новой версией и что делать, если возникнут проблемы.

#### Важность правильного объема сообщений:

Информировать пользователей необходимо, но при этом важно не перегрузить их слишком большим количеством информации. Избыточные сообщения могут быть проигнорированы, что приведет к непониманию и недовольству пользователей.

#### Выбор метода распространения информации:

Метод распространения зависит от культуры компании. В некоторых случаях удобнее использовать новостные группы для уведомлений, а в других — электронную почту. Важно, чтобы уведомления о значительных изменениях были отправлены "толканием", а не требованием проверять страницу или группу на регулярной основе.

Таким образом, правильная структура распространения информации помогает снизить количество возникающих проблем и повысить удовлетворенность пользователей.

### 17.1.3. ****Составление графика****

Распределение времени — это важная часть управления изменениями. Время внесения изменения может значительно повлиять на пользователей, поэтому необходимо тщательно планировать, когда и как вносить обновления. Рассмотрим три основных типа изменений: штатные, критические и крупномасштабные.

#### 1. **Штатные обновления**

Штатные обновления обычно происходят в любое время и не заметны для большинства пользователей. Эти изменения включают:

* Обновление серверов аутентификации или баз данных.
* Настройка системы для отдельных пользователей.
* Устранение проблем с рабочими станциями или принтерами.

Эти обновления не требуют специального графика, так как их влияние ограничено, и ошибка не затронет большую часть пользователей.

#### 2. **Крупномасштабные обновления**

Крупномасштабные обновления затрагивают большое количество систем или требуют серьезных нарушений в работе системы, сети или службы. Эти изменения включают:

* Модернизация системы аутентификации.
* Изменения в инфраструктуре электронной почты или печати.
* Обновление сетевой инфраструктуры.

График крупных изменений должен быть согласован с пользователями через механизмы уведомлений, например, электронную почту. Эти изменения не должны происходить слишком часто, а если это происходит, стоит пересмотреть их классификацию. Обычно крупномасштабные обновления происходят вне пиковой нагрузки. Однако в некоторых случаях они могут проводиться в рамках одного технического перерыва, чтобы минимизировать влияние на пользователей.

#### 3. **Критические обновления**

Критические обновления могут быть незначительными по масштабу, но они могут вызвать серьезные проблемы, если не будут выполнены правильно. Эти обновления могут включать:

* Изменения конфигурации маршрутизаторов или межсетевых экранов.
* Внесение изменений в критический сервер или систему.

Критические обновления требуют особого внимания и заблаговременного уведомления пользователей. Эти обновления должны быть проведены вне пиковых часов, чтобы обеспечить время для обнаружения и устранения проблем до того, как они затронут пользователей. Также рекомендуется, чтобы администратор оставался на месте некоторое время после внесения критического изменения для быстрого реагирования на возможные проблемы.

#### 4. **Выбор времени для критических обновлений**

Разные компании могут иметь различные подходы к времени внесения критических изменений. Например, в компании электронной коммерции критические изменения могут быть выполнены утром, когда сайт не так активно используется. В других компаниях предпочтительно проводить изменения в рабочие часы, чтобы можно было оперативно исправить возникшие проблемы.

Важно избегать ситуации, когда внешние специалисты или неавторизованные пользователи могут вносить изменения, которые могут негативно повлиять на важные системы.

#### 5. **Классификация изменений**

Некоторые изменения могут считаться штатными в одной компании и критическими в другой. Например, подключение нового узла в корпоративной сети может быть обычным обновлением в одной компании, но в другой — это может быть критическое изменение. Определите, какие изменения следует классифицировать как штатные, критические или крупномасштабные в зависимости от контекста компании и подразделения.

#### 6. **Запрет на изменения**

Важным элементом является определение времени, когда изменения можно проводить только в минимальном объеме. Это время запрета на изменения часто устанавливается в конце квартала или финансового года, когда критические процессы, такие как финансовые отчеты или аудит, требуют стабильности системы.

В графиках обновлений важно учитывать различные факторы, включая тип изменения, время воздействия и потребности бизнеса, чтобы минимизировать влияние на пользователей и обеспечить бесперебойную работу всех систем.

### 17.1.4. ****Процессы и документация****

Процессы и документация играют важную роль в управлении изменениями, обеспечивая системных администраторов необходимыми инструкциями для безопасного и правильного внесения изменений. Правильное соблюдение процессов и подготовка соответствующей документации позволяет избежать ошибок и минимизировать риски. Важно, чтобы системные администраторы заполняли формы контроля изменений, которые детально описывают следующее:

* Внесенные изменения и их причины.
* Затронутые системы и службы.
* Потенциальные риски и план отмены изменений.
* Время реализации изменений и план отмены.

Требования к деталям документации зависят от важности системы или службы. Для критически важных машин или служб процесс управления изменениями должен быть более строгим, чем для менее значимых систем. Например, в компаниях электронной коммерции важнейшие базы данных и серверы обработки кредитных карт требуют жесткой документации и контроля, тогда как менее важные службы, такие как электронная почта или печать, могут требовать менее строгих процедур.

#### 17.1.5. **Технические аспекты**

Процесс обновления конфигурационных файлов должен быть стандартизирован и документирован. Он должен включать пошаговые инструкции, которые применяются во всех случаях обновления системных файлов. Ключевые моменты включают:

* Создание истории изменений, чтобы можно было отслеживать и анализировать изменения в конфигурации.
* Блокировка конфигурационных файлов, чтобы предотвратить одновременное редактирование несколькими пользователями.
* Автоматизированные проверки формата информации в файлах.
* Уведомления других систем и приложений о произошедших обновлениях, если это необходимо.

#### 17.1.5.1. **История изменений и блокировка**

Система истории изменений позволяет отслеживать, кто и когда вносил изменения в файл конфигурации. Это очень важно для восстановления работы системы в случае повреждения текущей версии. История изменений также может включать комментарии к изменениям, чтобы другие администраторы могли понять, что именно было сделано и почему.

Кроме того, для предотвращения одновременного редактирования файла несколькими людьми используется система блокировки. Эти механизмы важны для поддержания целостности и безопасности данных. Системы контроля версий, такие как **SubVersion**, **Revision Control System** и **Concurrent Versions System**, обеспечивают такие функции, как блокировка файлов, идентификация пользователя и добавление комментариев к изменениям.

В системах под Windows аналогичные функции могут предоставлять инструменты, такие как **SourceSafe**. Системы контроля версий помогают следить за версиями файлов, что делает их полезными для работы с конфигурационными файлами и предотвращения ошибок при изменениях.

### 17.1.5.2. ****Автоматизированные проверки****

На заключительных этапах обновления конфигурационных файлов необходимо провести два важнейших действия: проверку синтаксических ошибок и гарантировать, что все приложения, использующие данный файл, начали применять новые настройки. Эти этапы должны выполняться с использованием автоматизированного программного обеспечения, которое может также уведомить серверы о том, что их конфигурационные файлы были изменены, или при необходимости распространить обновления на другие машины.

Иногда может потребоваться разделить эти два этапа. Если для применения обновленных конфигураций требуется временное прерывание обслуживания, можно провести синтаксическую проверку немедленно, а само обновление отложить до более удобного времени. Важно учитывать, что проверка синтаксиса и обновление системы не всегда происходят одновременно, чтобы минимизировать влияние на производительность.

Конфигурации некоторых систем трудно проверить автоматически. В таких случаях важно иметь альтернативный способ тестирования, чтобы гарантировать, что изменения будут корректными. Например, в UNIX-системах скрипты загрузки часто изменяются вручную, чтобы добавить или изменить службы, которые должны запускаться при старте системы, или изменить параметры сетевых интерфейсов. Такие скрипты требуют тщательной проверки, так как ошибка может помешать завершению перезагрузки системы. В некоторых UNIX-системах скрипты загрузки разделяются на отдельные компоненты, что позволяет проверять каждый из них по отдельности, повышая уверенность в правильности изменений.

Если скрипты не были проверены заранее, проблемы с ними могут быть обнаружены только при следующей перезагрузке системы. Это создает риск возникновения проблем в самый неудобный момент, например, в ночное время или в отпуске. Хорошие системы могут работать без перезагрузки несколько месяцев, что затрудняет отслеживание изменений, внесенных до последней перезагрузки. Ведение журналов изменений или система заявок может помочь, но при длительном времени между перезагрузками найти нужную информацию может быть сложно.

### 17.2. ****Тонкости****

После того как базовая структура управления изменениями, включающая процесс обновления конфигурации, распространение информации и составление графика изменений, будет установлена, можно применять дополнительные методы для повышения стабильности компании. Например, можно создать автоматизированные интерфейсы для распространенных изменений конфигурации, которые будут выполнять блокировку, ведение истории изменений, проверку и обновление без участия системных администраторов. Также полезно организовывать формальные собрания для обсуждения предложений об изменениях с участием нескольких сторон.

#### 17.2.1. **Автоматизированные интерфейсы**

Создание автоматических проверок для системных файлов, чтобы гарантировать их корректность перед утверждением изменений, обеспечивает большую стабильность системы. Следующий шаг в этом направлении – разработка оконечного интерфейса, который будет задавать необходимые вопросы, проверять ответы на ошибки, искать пропуски и обновлять файл, используя предоставленные данные. Это позволит централизованно проверять ошибки и обеспечит точность обновлений, устраняя необходимость многократной проверки отдельных системных файлов.

### 17.2.2. ****Собрания по вопросам управления изменениями****

Официально организованные собрания по вопросам управления изменениями являются важным инструментом для повышения стабильности систем. Эти собрания помогают обсуждать предлагаемые изменения, их возможные последствия, а также составить график внесения изменений. В рамках этого процесса системные администраторы обязаны сообщать, что и когда они планируют сделать, сколько времени это займет, как будет проверяться изменения, как будет осуществляться отмена и сколько времени это займет. Этот процесс заставляет администраторов тщательно продумывать последствия изменений и подготовиться к возможным проблемам.

Такие собрания также служат для предупреждения других сотрудников компании о предстоящих изменениях, что помогает избежать возникновения неожиданных проблем. Все ключевые сотрудники компании, чьи области могут быть затронуты изменениями, должны участвовать в этих заседаниях. Они могут предупредить свои группы и подготовить их к будущим изменениям. Участники собраний, которые активно участвуют в процессе одобрения, отклонения или переноса изменений, называются заинтересованными лицами.

Собрания по управлению изменениями обеспечивают общий обзор происходящих в компании изменений и дают возможность старшим системным администраторам и руководителям заметить потенциальные проблемы до внесения изменений. Это помогает избежать неупорядоченности и повысить стабильность работы системы. Обычно такие собрания проводятся раз в неделю или месяц в зависимости от частоты изменений.

Одобрение всех изменений заинтересованными лицами помогает повысить стабильность системы, а также предоставляет системным администраторам дополнительную защиту. Если сотрудники компании жалуются на сбои и изменения, происходящие в неудобное время, консультации по пересмотру изменений могут помочь вовлечь пользователей в процесс и получить их одобрение.

### 17.2.3. ****Упрощение процесса****

Когда основная структура управления изменениями будет установлена, следует оценить процесс с точки зрения его возможного упрощения. Стоит задуматься, есть ли избыточные или неиспользуемые вопросы в формах предложений изменений. Можно ли оптимизировать процесс, связанный с бумажной работой? Например, если формы являются сетевыми, можно ли добавить возможность сохранения значений по умолчанию для таких полей, как имя и контактная информация?

Также важно учитывать проблемы пользователей, использующих систему в ее текущем виде, чтобы понять, какие улучшения могут быть внесены в процесс управления изменениями, чтобы сделать его более эффективным.

### 17.3. ****Заключение****

Управление изменениями – это важное средство для повышения надежности системы. Оно помогает не только ограничивать внесение определенных изменений, но и обеспечивает наличие процесса предварительного рассмотрения изменений, который позволяет выявить потенциальные проблемы, которые могли быть упущены системным администратором. Управление изменениями также помогает в устранении проблем, поскольку все изменения отслеживаются, и в случае возникновения сложностей можно вернуться к предыдущим изменениям для их анализа.

Частота проведения собраний по управлению изменениями зависит от объема и частоты изменений в системе. Создание механизмов для предварительной проверки нормальной работы системы перед внесением изменений позволяет снизить риск того, что изменение, призванное устранить одну проблему, приведет к созданию новой или ухудшению стабильности системы.

### 18. Обновления серверов

Обновление операционной системы сервера — это критическая задача, требующая серьезной подготовки и тщательного тестирования. Хотя сама операция может показаться простой, она включает несколько этапов, которые обеспечат успешный результат и минимизируют риски. Этот процесс начинается с подготовки и использования контрольного списка. Если процесс обновления ОС не будет строго структурирован, могут возникнуть проблемы, даже если они кажутся незначительными на первый взгляд.

### 18.1. ****Основы****

Основная цель обновления ОС заключается в том, чтобы после обновления все сервисы, которые работали до этого, продолжали функционировать без сбоев. Кроме того, обновление должно либо улучшать функциональность, либо повышать надежность системы, но не снижать их. Чтобы достичь этого, процесс обновления должен быть организован четко и следовать определенной последовательности.

#### Шаги процесса обновления ОС:

1. **Составьте контрольный список служб:**
   * Какие службы предоставляет сервер?
   * Кто является пользователем каждой из этих служб?
   * Какие программы предоставляют эти службы?
2. **Проверьте совместимость программ с новой ОС:**
   * Убедитесь, что программы, которые обеспечивают работу служб, совместимы с новой версией операционной системы, или запланируйте их обновление.
3. **Разработайте тесты для проверки работоспособности служб:**
   * Для каждой службы необходимо подготовить тесты, которые подтвердят, что она работает правильно после обновления.
4. **Напишите план отмены:**
   * Определите условия, при которых обновление можно будет отменить, и разработайте процедуру отмены.
5. **Выберите технический перерыв:**
   * Запланируйте время, когда обновление будет проводиться, и уведомите об этом всех заинтересованных сторон.
6. **Объявите об обновлении:**
   * Уведомите пользователей и другие заинтересованные стороны о времени и процессе обновления.
7. **Проверьте тесты:**
   * Выполните тесты, чтобы удостовериться, что они действительны и работают как ожидается.
8. **Заблокируйте пользователей:**
   * Перед выполнением обновления блокируйте доступ пользователей, чтобы избежать вмешательства во время процесса.
9. **Проведите обновление с наблюдением:**
   * Обновление следует проводить под наблюдением, возможно, с помощью другого человека, чтобы он мог оперативно реагировать на проблемы.
10. **Повторите все тесты:**
    * После выполнения обновления повторите все тесты, чтобы удостовериться в правильности работы системы.
11. **Выполнение плана отмены (при необходимости):**
    * Если тесты не проходят или происходят другие сбои, которые требуют отмены обновления, выполните план отмены.
12. **Разблокируйте пользователей:**
    * После успешного обновления и тестирования разблокируйте пользователей, позволив им снова использовать сервер.
13. **Сообщите пользователям о завершении или отмене обновления:**
    * Уведомите всех пользователей о том, что обновление завершено, или, если оно было отменено, объясните причины.
14. **Анализируйте процесс:**
    * Проанализируйте, что прошло успешно, а что не удалось, и обновите контрольный список с учетом полученного опыта.

Каждый из этих шагов обеспечивает минимизацию рисков и помогает гарантировать успешное обновление, минимизируя влияние на работу сервисов и пользователей.

### 18.1.1. Этап 1: Составьте контрольный список служб

Контрольный список служб — это основа для выполнения всего процесса обновления. В нем нужно отразить все службы, которые предоставляет сервер, а также указать, кто использует эти службы и какие программы их обеспечивают. Этот список будет использоваться на всех этапах обновления, обеспечивая прозрачность и контроль за каждой службой.

#### Рекомендации по составлению контрольного списка:

* **Использование электронных таблиц**: Электронные таблицы являются отличным инструментом для организации и совместного использования информации. Преимущество электронных таблиц заключается в их легкости для обновления и доступности для всех участников процесса. Лучше всего разместить таблицу в Интернете, чтобы обеспечить доступность последней версии документа для всех участников.
* **Обновления через сеть**: Вместо того чтобы отправлять файл каждому участнику, предоставьте доступ к нему через интернет, что позволит всем увидеть актуальную информацию. Важно, чтобы доступ был активным, и все обновления своевременно доходили до всех участников.
* **Совместная проверка**: Рекомендуется провести встречу с ключевыми участниками, которые будут задействованы в процессе, и представить им план. Они смогут проверить, все ли учтено, и предложить корректировки. Это обеспечит наибольшую эффективность процесса.

### 18.1.2. Этап 2: Проверьте совместимость программ

После составления контрольного списка важно проверить, какие программы будут работать с новой операционной системой. Для этого нужно связаться с разработчиками или провести тестирование, чтобы удостовериться в совместимости программ с обновленной ОС.

#### Что делать, если программа не совместима:

* **Обновление до версии, поддерживаемой обеими ОС**: Если возможно, обновите программу до версии, которая поддерживается и текущей, и новой операционной системой. Это можно сделать заранее, еще до обновления ОС.
* **Обновление после обновления ОС**: Если программа совместима только с новой ОС, запланируйте ее обновление сразу после обновления ОС. Это особенно актуально для редко используемых приложений, где можно скоординировать перерывы в работе.
* **Программа больше не поддерживается**: В случае, если программа больше не поддерживается разработчиком, могут возникнуть проблемы с обновлением. В таких случаях возможно потребуется поиск альтернатив или полное отказ от использования данного продукта.

### 18.1.3. Этап 3: Тесты для проверки

Когда службы и программы проверены на совместимость, нужно подготовить тесты для каждой службы, чтобы убедиться, что она работает правильно после обновления.

#### Как организовать тестирование:

* **Автоматические скрипты**: Наилучший вариант — это создать автоматизированные скрипты для тестирования, которые будут запускаться и проверять работоспособность каждой службы. Скрипты могут выводить сообщения о результатах тестирования (например, «OK» или «FAIL»).
* **Использование инструментов тестирования**: Некоторые программы могут иметь встроенные средства для тестирования установки, которые помогут подтвердить, что все работает правильно. Если такие средства недоступны, их можно запросить у разработчиков.
* **Регрессивное тестирование**: Важно, чтобы результаты работы службы на новой ОС совпадали с результатами на старой. Для этого можно использовать такие инструменты, как команду diff в UNIX, которая сравнивает два текстовых файла и указывает различия между ними. Это может помочь удостовериться, что обновление не привело к нежелательным изменениям.

#### Примечание:

Тесты могут быть простыми, как например, компиляция программы "Hello, World!", или более сложными, с проверкой конкретных результатов работы программ. Важно избегать поверхностных тестов, которые могут не выявить возможных проблем.

### 18.1.4. Этап 4: Напишите план отмены

План отмены — это ключевой элемент, который обеспечит возможность возврата к предыдущему состоянию в случае проблем с обновлением. Без такого плана процесс обновления становится рискованным, так как в случае неполадок, которые могут возникнуть в процессе, вы должны быть готовы к быстрому реагированию.

#### Ключевые моменты при разработке плана отмены:

1. **Решение о времени отмены**: План отмены должен быть активирован по истечении установленного времени для обновления, которое должно включать как время на само обновление, так и дополнительные временные интервалы для устранения неполадок. Если по завершении этого времени обновление не удалось или появились критические ошибки, необходимо начать процесс отката.
2. **Мониторинг времени**: Важно, чтобы за временем следил кто-то, не входящий в процесс обновления, например, руководитель или старший администратор.
3. **Резервные копии**: Для малых и средних систем можно создать полные резервные копии до начала обновления. Для крупных систем достаточно регулярно делать резервные копии, чтобы иметь возможность быстро восстановить систему в случае неудачи.

### 18.1.5. Этап 5: Выберите технический перерыв

В этом этапе необходимо согласовать с пользователями время, когда обновление будет проходить. Этот процесс включает в себя организационные и технические моменты:

1. **Когда?**: Определите, в какое время будет удобно провести обновление. Обычно это время, когда системы не критичны для бизнеса, например, в ночное время или в выходные дни. Учитывайте возможные ограничения, такие как доступность поддержки разработчиков.
2. **Сколько времени?**: Продолжительность технического перерыва зависит от времени, необходимого на обновление, устранение возможных проблем и выполнения плана отмены. На начальных этапах лучше оценивать время с запасом, чтобы избежать недооценки.
3. **Риски и задержки**: Сообщите пользователям, что перерыв может быть дольше, чем планировалось, из-за непредвиденных факторов. Однако, если процесс завершится раньше, обязательно проинформируйте пользователей.

### 18.1.6. Этап 6: Сообщите об обновлении в соответствии с установленным порядком

На этом этапе важно уведомить пользователей о предстоящем обновлении, чтобы они могли подготовиться. Используйте одинаковый формат для всех уведомлений, чтобы пользователи знали, чего ожидать.

#### Рекомендации:

1. **Использование единого формата**: Независимо от того, как именно будет передано сообщение (по электронной почте, голосовой почте, через новостную группу, веб-страницу или даже бумажное сообщение), важно поддерживать единый формат. Это поможет избежать путаницы и обеспечить согласованность.
2. **Четкость и лаконичность**: Сообщение должно быть кратким и по теме. Большинство пользователей будут читать только строку "Тема", поэтому важно, чтобы заголовок был понятным и информативным.

### 18.1.7. Этап 7: Выполните тесты

Прежде чем начать обновление, важно провести тесты, чтобы убедиться, что текущие проблемы не связаны с системой до обновления. Это предотвратит ситуацию, когда план отмены выполняется только для того, чтобы обнаружить, что проблемы существовали и до обновления.

### 18.1.8. Этап 8: Заблокируйте пользователей

Перед началом обновления следует заблокировать пользователей, чтобы избежать их попыток входа в систему во время технического перерыва. Лучше позволить пользователям выйти из системы самостоятельно, чем принудительно их отключать. Однако важно использовать средства ОС для предотвращения новых входов, чтобы избежать недоразумений, когда пользователи могут подумать, что система доступна, даже если обновление еще продолжается.

### 18.1.9. Этап 9: Выполните обновление под наблюдением

Когда приходит время обновления, не выполняйте его в одиночку. На этом этапе важно работать в команде, поскольку два человека могут быть полезны для быстрого выявления ошибок и улучшения качества работы. Также может быть полезным привлечение более опытных коллег, если обновление не прошло успешно.

Совместное выполнение обновлений способствует лучшему усвоению знаний и повышению качества работы. Важно всегда помнить, что наличие второго человека может помочь избежать многих ошибок и ускорить решение проблем.

### 18.1.10. Этап 10: Проверьте свою работу

После выполнения обновления повторно запустите все тесты, чтобы убедиться в корректности системы. Если тесты не проходят, необходимо действовать по стандартному процессу отладки. Очень важно пройти все тесты, даже если некоторые из них прошли успешно, так как изменение одного компонента может повлиять на другие системы.

Привлекайте пользователей для проверки работы системы. Их обратная связь важна для окончательной проверки успешности обновления.

### 18.1.11. Этап 11: Если ничего не получилось, выполните план отмены

Если по прошествии времени обновление не завершено, или если тесты не выполняются, следует немедленно начать выполнение плана отмены. Это решается временем, которое вы отведете на обновление. После того как план отмены активирован, следует сразу приступить к восстановлению системы до предыдущего состояния.

Записывайте все детали, например, если изменения были частично выполнены, если обновление не удалось или если служба была недоступна. Это поможет в будущем при анализе произошедшего и улучшении процессов.

### 18.1.12. Этап 12: Восстановите доступ пользователей

Когда обновление завершено, необходимо восстановить доступ пользователям. Это может включать проверку работы службы и подтверждение, что все системы работают как ожидалось. Важно выполнить эту задачу аккуратно, чтобы не возникло новых проблем с системой.

### 18.1.13. Этап 13: Сообщите о завершении/отмене

После того как обновление завершено, сообщите пользователям о завершении обновления или о том, что был задействован план отмены. В сообщении укажите, какие службы снова работают, и как пользователи могут сообщить о возможных проблемах.

Сообщения должны быть краткими и ясными, чтобы пользователи могли быстро понять, что произошло, и как действовать в случае возникновения ошибок.

### 18.2.1. Добавляйте и удаляйте службы одновременно

Во время обновления иногда нужно добавлять и удалять службы одновременно. Это может усложнить процесс, так как требует особых тестов и внимания. Когда вы добавляете службы, важно помнить, что тесты для новых служб можно запустить на другом узле, если они уже доступны там. Удаление службы может быть как простым, так и сложным. Простой процесс удаления предполагает, что можно будет быстро восстановить службу, если обнаружатся забытые зависимости.

Для упрощения работы с удаленными службами можно оставить возможность их активации в будущем, например, путем добавления комментариев в конфигурационные файлы или изменения их состояния на «OFF» или «DISABLED», чтобы другие администраторы не включили службу по ошибке.

### 18.2.2. Полная установка

Иногда вместо того чтобы обновлять систему, гораздо проще выполнить полную установку. Это позволяет избежать накопления устаревших файлов, фрагментации системы и других проблем. Полная установка на временной машине, которая идентична основной, дает возможность проводить обновление спокойно, не прерывая работу старой системы. После того как новая система настроена и проверена, можно подключить необходимые диски и настроить конфигурацию сети.

### 18.2.3. Повторное использование тестов

Если тесты были хорошо написаны, их можно интегрировать в систему мониторинга в реальном времени, что упростит процесс обновлений. Например, при использовании системы мониторинга для проверки состояния служб обновление не потребует дополнительных тестов. Однако некоторые тесты, такие как тесты нагрузки, сложно выполнять на работающей системе. В таких случаях их можно запускать в периоды низкой загрузки или по запросу во время отладки.

### 18.2.4. Запись изменений системы

Записывание изменений в систему помогает отслеживать, что было добавлено или изменено на сервере. Например, в UNIX-системах можно использовать файл /var/adm/CHANGES для записи изменений. Регулярное обновление этого файла помогает поддерживать его актуальность. Чтобы упростить процесс, можно создать псевдоним оболочки или скрипт, который автоматически открывает этот файл для редактирования.

Хранить логи изменений следует в общедоступных местах, таких как википедии или на сервере, чтобы они были доступны в случае сбоя машины. Важно установить и придерживаться правил хранения этих логов, чтобы избежать путаницы.

### 18.2.5. Генеральная репетиция

Перед выполнением обновлений на основной машине полезно провести генеральную репетицию на другой машине. Это поможет выявить возможные проблемы и оценить время, необходимое для обновления. Генеральная репетиция также позволяет подготовиться к непредвиденным ситуациям и усовершенствовать процесс.

Кроме того, такие репетиции помогают лучше понять, как взаимодействуют все заинтересованные стороны, и как синхронизировать работу различных команд. В театре аналогичные репетиции помогают организовать взаимодействие между актерами и техническими специалистами, что также важно для системных администраторов.

### 18.2.6. Установка старых и новых версий на одной машине

Иногда обновление касается только одной службы, а не всей операционной системы. В таких случаях полезно, если старые версии остаются на машине в отключенном состоянии, пока новые версии не установлены и не сертифицированы. Например, веб-сервер Apache можно установить в директорию /opt/apache-x.y.x, где x.y.z — это номер версии. Для удобства используется символическая ссылка в /opt/apache, которая указывает на текущую активную версию. Если новая версия вызывает проблемы, достаточно изменить символическую ссылку, чтобы вернуться к предыдущей версии, что является простым способом отката.

В некоторых случаях старая и новая версия программы могут работать одновременно. Например, можно запустить новую версию Apache на другом порте, не затрагивая старую версию, что удобно для отладки.

### 18.2.7. Минимальные изменения первоначальной версии

Обновления проще, если минимизировать изменения в системе. В UNIX-системах можно загружать обновления в отдельный раздел, что позволяет оставлять системные разделы в неизменном виде. Изменения обычно касаются конфигурационных файлов в директории /etc, которые можно заранее скопировать для справки, чтобы не восстанавливать их с резервных копий. В системах, где данные загружаются с сервера, достаточно сохранить только несколько директорий, таких как /var, а также задачи из crontab и at, данные электронной почты и, в некоторых системах, файлы менеджера календаря. Для отслеживания изменений в конфигурации удобно использовать системы контроля версий, такие как RCS.

### 18.3. Заключение

Мы рассмотрели процесс обновления операционной системы, акцентируя внимание не на конкретных командах, а на общих принципах. Ключевыми моментами процесса являются внимание к деталям, тестирование и обмен информацией. Контрольный список является основным инструментом для управления обновлениями: он помогает отслеживать, какие службы нужно обновить, сколько времени это займет и когда будет выполнено обновление.

Кроме того, контрольный список помогает определить необходимые тесты, которые можно повторно использовать и которые обеспечат качество обновления. Если что-то идет не так, план отмены, включенный в список, поможет вернуться к предыдущему состоянию. Важно, чтобы информация о процессе обновления была доступна всем заинтересованным сторонам, что помогает улучшить взаимодействие между пользователями и системными администраторами.

Автоматизация тестов позволяет повторно проверять систему и интегрировать их в системы мониторинга в реальном времени, что значительно упрощает выполнение обновлений в будущем. Этот процесс является основой для более сложных обновлений, и его освоение является важным шагом для системного администратора.

Хотя обновления не всегда гарантируют 100% успех, способность быстро вернуться к предыдущему состоянию при проблемах минимизирует риски. Меньшее количество команд и действий снижает вероятность ошибок, и возможность повторного применения метода обновления на нескольких машинах помогает поддерживать целостность системы.