МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение
высшего образования
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

Факультет среднего профессионального образования

Методические рекомендации

для выполнения лабораторных работ

по междисциплинарному курсу

МДК 03.01 «ВНЕДРЕНИЕ И ПОДДЕРЖКА КОМПЬЮТЕРНЫХ СИСТЕМ»

образовательной программы среднего профессионального образования

(ОП СПО)

по специальности

09.02.07 ««Информационные системы и программирование»»

Санкт-Петербург
2023 г.

СОДЕРЖАНИЕ

Лабораторная работа№1. «Установка серверного программного обеспечения»

---

1. Цель работы

- Освоить процесс установки серверной операционной системы (ОС).
- Научиться устанавливать и настраивать серверное программное обеспечение (ПО).
- Проверить работоспособность серверных служб.

🚀 **Рекомендуемое время выполнения: 6 часов**

---

2. Оборудование и программное обеспечение

2.1. Аппаратное обеспечение

- Компьютер с поддержкой виртуализации (Intel VT-x / AMD-V).
- Минимальные требования для виртуальной машины (ВМ):
  - **ОЗУ:** 4 ГБ (рекомендуется 4 ГБ для Windows Server).
  - **Жесткий диск:** 40–60 ГБ (в зависимости от ОС).
  - **Сетевой адаптер:** NAT (для выхода в интернет) или мостовое подключение (для локальной сети).

2.2. Программное обеспечение

- **Гипервизор** Windows Hyper-V Server 2019.
- **Серверная ОС** (на выбор):
  - **Windows Server 2019/2022** (официальный ISO от Microsoft).
  - **Linux-сервер (Ubuntu Server, CentOS, Debian)**.
- **Серверные приложения** (для установки):
  - Веб-сервер (**IIS, Apache, Nginx**).
  - СУБД (**MySQL, PostgreSQL, MS SQL Server**).
  - Дополнительные сервисы (**FTP, почтовый сервер, файловый сервер**).

---

3. Ход работы

3.1. Подготовка виртуальной машины

1. **Создание ВМ**:
   - Запустить гипервизор.
   - Нажать **"Создать"** → указать имя (например, **"WinServer2022"**).
   - Выбрать тип ОС (**Windows / Linux**).
   - Выделить **ОЗУ (2048–4096 МБ)**.
   - Создать виртуальный жесткий диск (**VDI, 40–60 ГБ, динамический**).
2. **Настройка сети**:
   - В свойствах ВМ → **Сеть** → выбрать **"Сетевой мост"** или **"NAT"**.
3. **Подключение ISO-образа**:
   - В настройках ВМ → **Носители** → выбрать ISO-файл серверной ОС.

3.2. Установка серверной ОС

Для Windows Server

1. Запустить ВМ → начать установку.
2. Выбрать **язык, раскладку клавиатуры, время**.
3. Нажать **"Установить"** → выбрать версию (**Standard/Datacenter**).
4. Принять лицензию → выбрать **"Выборочная установка"**.
5. Разметить диск (**создать раздел 40+ ГБ**).
6. После установки задать **пароль администратора** (сложный!).
7. Войти в систему → запустить **sconfig** для базовой настройки.

Для Linux (Ubuntu Server)

1. Запустить ВМ → выбрать язык → **"Установить Ubuntu Server"**.
2. Настроить раскладку клавиатуры.
3. Выбрать **"Guided - use entire disk"** (автоматическая разметка).
4. Указать имя сервера, пользователя и пароль.

5. Установить **OpenSSH** (если нужен удаленный доступ).
6. Завершить установку → перезагрузить.

3.3. Первоначальная настройка сервера

Для Windows Server

1. **Обновление системы**:
   - Открыть **sconfig** → **6) Загрузка обновлений** → установить.
2. **Настройка сети**:
   - `sconfig` → **8) Настройка сети** → задать статический IP (если нужно).
3. **Включение RDP (удаленный доступ)**:
   - `sconfig` → **7) Настройка удаленного управления** → включить.

Для Linux (Ubuntu Server)

1. **Обновление пакетов**:

```bash
sudo apt update && sudo apt upgrade -y
```

2. **Настройка статического IP (если нужно)**:

```bash
sudo nano /etc/netplan/00-installer-config.yaml
```

Пример конфигурационного файла:

```yaml
network:
  ethernets:
    ens33:
      dhcp4: no
      addresses: [192.168.1.100/24]
      gateway4: 192.168.1.1
      nameservers:
        addresses: [8.8.8.8, 1.1.1.1]
```

Применить:

```bash
sudo netplan apply
```

3. **Проверка SSH**:

```bash
sudo systemctl status ssh
```

3.4. Установка серверного ПО

Веб-сервер

- **Windows (IIS)**:
  - `sconfig` → **1) Добавить роли и компоненты** → **"Веб-сервер (IIS)"**.
  - Проверить: открыть браузер → `http://localhost`.
- **Linux (Apache/Nginx)**:

```bash
# Apache
sudo apt install apache2 -y
sudo systemctl start apache2

# Nginx
sudo apt install nginx -y
sudo systemctl start nginx
```

Проверить:

```bash
curl http://localhost
```

СУБД (MySQL/PostgreSQL)

- **Windows (MS SQL)**:
  - Установить через **"Добавить роли"** → **"Сервер базы данных"**.
- **Linux (MySQL)**:

```bash
sudo apt install mysql-server -y
sudo mysql_secure_installation
```

Дополнительные сервисы (FTP, файловый сервер)

- **FTP-сервер (Linux: vsftpd)**:

```bash
sudo apt install vsftpd -y
sudo systemctl start vsftpd
```

---

4. Проверка работоспособности

1. **Веб-сервер**:
   - Открыть браузер → `http://<IP_сервера>` → должна быть стартовая страница.
2. **СУБД**:
   - Подключиться через **MySQL Workbench** или `mysql -u root -p`.
3. **FTP**:
   - Проверить подключение через **FileZilla**.

---

5. Контрольные вопросы

1. Какие этапы включает установка серверной ОС?
2. Чем отличается серверная ОС от десктопной?
3. Как настроить статический IP в Linux?
4. Какие команды используются для управления сервисами в Linux?
5. Как проверить, что веб-сервер работает?

---

6. Вывод

В ходе работы были установлены и настроены:

✅ Серверная ОС (Windows/Linux).

✅ Веб-сервер (IIS/Apache/Nginx).

✅ СУБД (MySQL/PostgreSQL).

✅ Дополнительные сервисы (FTP).

Приобретены навыки развертывания серверного ПО и проверки его работоспособности.

Приложение (скриншоты)

(В отчет добавить скриншоты ключевых этапов: установка ОС, настройка сети, работа серверов.)

📌 Готовый отчет должен содержать:

- Титульный лист.
- Цель работы.
- Пошаговое описание с скринами.
- Ответы на вопросы.
- Выводы.

Лабораторная работа№2. «Настройка серверного программного обеспечения»

---

1. Цель работы

1. Освоить настройку основных серверных служб
2. Научиться конфигурировать веб-сервер, СУБД и файловые службы
3. Обеспечить безопасность серверных приложений

🚀 Рекомендуемое время выполнения: 4 часа

---

2. Оборудование и ПО

- Серверная ОС (Windows Server 2022 или Ubuntu Server 22.04)
- Веб-сервер (IIS/Apache/Nginx)
- СУБД (MySQL/PostgreSQL/MS SQL)
- Дополнительное ПО:
  - FTP-сервер (FileZilla Server/vsftpd)
  - Почтовый сервер (hMailServer/Postfix)
  - Файловый сервер (Samba)

---

3. Настройка серверных компонентов

3.1. Настройка веб-сервера

Для Windows (IIS)

1. Установка роли:

```powershell
Install-WindowsFeature Web-Server, Web-Mgmt-Console -IncludeAllSubFeature
```

2. Создание сайта:

```powershell
New-Website -Name "CompanySite" -Port 443 -PhysicalPath "C:\sites\company" -HostHeader "company.local" -Ssl
```

3. Настройка SSL:
   - Генерация самоподписанного сертификата
   - Привязка к сайту через "Привязки SSL"
4. Ограничение доступа:
   - Настройка IP-фильтрации
   - Включение Basic-аутентификации

Для Linux (Apache + Nginx)

1. Установка LAMP-стека:

```bash
sudo apt install apache2 mysql-server php libapache2-mod-php php-mysql
```

2. Виртуальные хосты Apache:

```bash
sudo nano /etc/apache2/sites-available/company.conf
```

```apache
<VirtualHost *:80>
    ServerName company.local
    DocumentRoot /var/www/company
    <Directory /var/www/company>
        Options -Indexes +FollowSymLinks
        AllowOverride All
    </Directory>
</VirtualHost>
```

3. Настройка Nginx как reverse proxy:

```nginx
server {
    listen 80;
    server_name company.local;
    location / {
        proxy_pass http://localhost:8080;
    }
}
```

3.2. Настройка СУБД

MySQL/MariaDB

1. Первоначальная настройка:

```bash
sudo mysql_secure_installation
```

2. Создание пользователя и БД:

```sql
CREATE DATABASE company_db;
CREATE USER 'admin'@'localhost' IDENTIFIED BY 'StrongPassword123!';
GRANT ALL PRIVILEGES ON company_db.* TO 'admin'@'localhost';
FLUSH PRIVILEGES;
```

3. Настройка удаленного доступа:

```bash
sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf
```

Изменить:

```ini
bind-address = 0.0.0.0
```

PostgreSQL

1. Настройка аутентификации:

```bash
sudo nano /etc/postgresql/14/main/pg_hba.conf
```

Добавить:

```conf
host    all             all             192.168.1.0/24          md5
```

3.3. Настройка файловых служб

Windows (Общие папки)

1. Создание общей папки:
   - ПКМ по папке → Свойства → Доступ → Расширенная настройка
   - Настройка NTFS-прав
2. Настройка квот:

```powershell
New-FsrmQuota -Path "D:\Shared" -Size 10GB
```

Linux (Samba)

1. Установка и настройка:

```bash
sudo apt install samba
sudo nano /etc/samba/smb.conf
```

```ini
[company_share]
path = /srv/company
valid users = @smbgroup
writable = yes
browseable = yes
```

---

4. Настройка безопасности

4.1. Базовые меры

- Настройка брандмауэра:

```bash
sudo ufw allow 22/tcp
sudo ufw allow 80,443/tcp
sudo ufw enable
```

- Регулярные обновления:

```bash
sudo apt update && sudo apt upgrade -y
```

4.2. Мониторинг

1. Установка и настройка Zabbix:

```bash
sudo apt install zabbix-agent
sudo nano /etc/zabbix/zabbix_agentd.conf
```

2. Настройка логирования:

```bash
sudo nano /etc/rsyslog.d/50-default.conf
```

---

5. Контрольные вопросы

1. Как ограничить доступ к веб-серверу по IP?
2. Какие методы аутентификации поддерживает IIS/Apache?
3. Как настроить репликацию в MySQL?
4. Какие параметры безопасности важны для Samba?
5. Как настроить автоматическое резервное копирование СУБД?

---

6. Вывод

В ходе работы выполнено:

✅ Детальная настройка веб-серверов (IIS, Apache, Nginx)

✅ Конфигурация СУБД с учетом безопасности

✅ Организация файлового обмена (SMB/Samba)

✅ Обеспечение базовой безопасности сервера

Приобретены навыки профессиональной настройки серверного ПО.

Приложение

(Должно содержать: скриншоты конфигурационных файлов, тесты подключения, схемы сетевой архитектуры)

📌 Критерии оценки:

- Полнота выполнения заданий
- Корректность настроек

- Доказательство работоспособности
- Оформление отчета

Лабораторная работа№3. «Выявление и документирование проблем установки программного обеспечения»

---

1. Цель работы

1. Освоить методы диагностики проблем при установке ПО
2. Научиться анализировать логи установки
3. Разработать систему документирования инцидентов
4. Отработать процедуры устранения типовых ошибок

🚀 **Рекомендуемое время выполнения: 2 часа**

---

2. Оборудование и ПО

- Виртуальная машина с Windows Server 2022/Linux
- Проблемные пакеты ПО для тестирования
- Инструменты диагностики:
  - **Windows**: Event Viewer, PowerShell, ProcMon
  - **Linux**: journalctl, dmesg, strace
- Системы документирования:
  - Чек-листы
  - Базовые формы отчетов
  - Система отслеживания инцидентов (на примере Redmine)

---

3. Методика проведения эксперимента

3.1. Имитация проблемных ситуаций

№	Тип проблемы	Способ имитации
1	Несовместимость версий	Установка ПО для другой версии ОС
2	Конфликт зависимостей	Удаление требуемых DLL/библиотек
3	Недостаток прав	Запуск от имени пользователя с ограничениями

№	Тип проблемы	Способ имитации
4	Антивирусная блокировка	Добавление файлов в карантин
5	Поврежденный инсталлятор	Ручное редактирование EXE/MSI-файла

---

4. Пошаговое выполнение работы

4.1. Этап диагностики

Для Windows:

1. Анализ журналов установки:

```powershell
Get-WinEvent -LogName "Application" | Where-Object {$_.Id -eq 11707}
```

2. Использование Process Monitor:
   - Фильтрация по процессу установки (msiexec.exe)
   - Поиск операций с результатом "ACCESS DENIED"

Для Linux:

1. Просмотр логов установки:

```bash
journalctl -xe | grep -i "error\|fail"
```

2. Проверка зависимостей:

```bash
ldd /путь/к/исполняемому/файлу
```

4.2. Документирование проблем

Форма отчета:

```markdown
# Отчет об инциденте №[ID]

**Дата:** [дата]
**Система:** [ОС, версия]
**ПО:** [название, версия]

## Описание проблемы:
[Подробное описание симптомов]

## Шаги воспроизведения:
1. [Действие 1]
2. [Действие 2]

## Результаты диагностики:
- Код ошибки: [код]
- Критические сообщения:
```

[выдержка из логов]

```
## Решение:
[Предложенный метод исправления]

## Статус:
✅ Решено / ⚠️ Требует доработки
```

4.3. Практические кейсы

Кейс 1: Ошибка 0x80070005 при установке MSI

Диагностика:

1. Проверка прав учетной записи
2. Анализ журнала установки:

```cmd
msiexec /i package.msi /lv* install.log
```

Решение:

```powershell
Start-Process msiexec.exe -ArgumentList "/i package.msi /qn" -Verb RunAs
```

Кейс 2: Конфликт библиотек в Linux

Диагностика:

```bash
strace -f -o install.log ./install.sh
grep "ENOENT" install.log
```

Решение:

```bash
sudo apt --fix-broken install
```

---

5. Контрольные вопросы

1. Какие журналы Windows содержат информацию об ошибках установки?
2. Как определить недостающие зависимости в Linux?
3. Какие инструменты позволяют отслеживать системные вызовы?
4. Как документировать повторяющиеся инциденты?
5. Какие превентивные меры уменьшат количество проблем?

---

6. Вывод

В ходе работы:

✅ Освоены методы диагностики ошибок установки

✅ Разработана система документирования инцидентов

✅ Отработаны сценарии решения типовых проблем

Приобретенные навыки позволяют:

- Эффективно анализировать причины сбоев
- Создавать четкие инструкции для службы поддержки
- Сокращать время простоя при развертывании ПО

Приложение

1. Пример заполненного отчета об инциденте
2. Скриншоты диагностических утилит
3. Чек-лист типовых проверок

Критерии оценки:

- Полнота анализа ошибки
- Корректность предложенного решения
- Качество документации
- Скорость диагностики

Лабораторная работа№4.1 «Анализ эксплуатационных характеристик качества программного обеспечения»

---

1. Цель работы

1. Изучить ключевые показатели качества ПО
2. Освоить методы тестирования производительности
3. Провести сравнительный анализ качества различных программных продуктов
4. Научиться оформлять технические отчеты по результатам тестирования

🚀 **Рекомендуемое время выполнения: 2 часа**

---

2. Теоретическая часть

2.1. Основные характеристики качества ПО

Характеристика	Критерии оценки	Методы тестирования
Функциональность	Полнота, корректность, совместимость	Юнит-тесты, интеграционные тесты
Надежность	Отказоустойчивость, восстановление	Нагрузочное тестирование
Производительность	Время отклика, пропускная способность	Профилирование, бенчмаркинг
Удобство использования	Эргономика, обучаемость	UX-тестирование, опросы
Безопасность	Аутентификация, защита данных	Penetration-тестирование
Сопровождаемость	Читаемость кода, модульность	Анализ метрик кода (SonarQube)

---

3. Практическая часть

3.1. Подготовка к тестированию

Оборудование:

- Тестовый стенд:
  - ОС Windows 10/11 или Linux
  - 8 ГБ ОЗУ, SSD 256 ГБ
  - Сетевое подключение

Тестируемое ПО (на выбор):

1. Система управления базами данных (MySQL vs PostgreSQL)
2. Веб-браузеры (Chrome vs Firefox)
3. Офисные пакеты (LibreOffice vs MS Office)

3.2. Методика тестирования

Тест 1: Оценка производительности

1. Замер времени выполнения операций:

```python
import time
start = time.time()
# Тестируемая операция
end = time.time()
print(f"Время выполнения: {end-start} сек")
```

2. Нагрузочное тестирование (для веб-приложений):

```bash
ab -n 1000 -c 100 http://test-site.local/
```

Тест 2: Анализ надежности

1. Имитация сбоев:
   - Принудительное завершение процесса
   - Отключение сети во время операции
2. Проверка восстановления:
   - Автосохранение документов
   - Восстановление сеанса работы

Тест 3: Оценка безопасности

1. Проверка уязвимостей:

```bash
nmap -sV --script vuln target_ip
```

2. Анализ журналов доступа:

```bash
sudo cat /var/log/auth.log | grep "Failed"
```

3.3. Оформление результатов

Шаблон отчета:

```markdown
# Анализ качества ПО: [Название продукта]

## 1. Методология тестирования
- Используемые инструменты
- Критерии оценки

## 2. Результаты тестов
### 2.1. Производительность
| Операция          | Время выполнения | Ресурсы CPU/RAM |
|-------------------|------------------|-----------------|
| Открытие файла    | 1.2 сек          | 15%/200 МБ      |

### 2.2. Надежность
- Успешное восстановление после 10/12 сбоев

## 3. Выводы и рекомендации
- Сильные стороны продукта
- Критические недостатки
- Рекомендации по улучшению
```

---

4. Контрольные вопросы

1. Какие метрики кода влияют на сопровождаемость ПО?
2. Как рассчитать коэффициент готовности системы?

3. Какие инструменты используют для нагрузочного тестирования веб-приложений?
4. Как провести сравнительный анализ удобства интерфейсов?
5. Какие стандарты регламентируют качество ПО в России?

---

5. Вывод

В ходе работы:

✅ Проведен комплексный анализ характеристик качества ПО

✅ Освоены современные методы тестирования

✅ Разработаны рекомендации по улучшению продуктов

Приобретенные компетенции:

- Навыки работы с инструментами мониторинга
- Умение интерпретировать результаты тестов
- Опыт оформления технической документации

Приложение

1. Примеры отчетов по разным категориям ПО
2. Скриншоты инструментов тестирования
3. Шаблоны для оформления результатов

Критерии оценки:

- Глубина анализа (40%)
- Корректность методик (30%)
- Качество оформления (20%)
- Практическая ценность выводов (10%)

Лабораторная работа№4.2 «Анализ эксплуатационных характеристик качества программного обеспечения»

🚀 **Рекомендуемое время выполнения**: 2 часа

---

1. Теоретико-методологическая основа исследования

1.1. Современные стандарты качества ПО

ГОСТ Р ИСО/МЭК 25010:2021 определяет 8 основных характеристик качества:

1. **Функциональная пригодность**
   - Полнота реализованных функций
   - Корректность выполнения операций
   - Соответствие техническому заданию
   - Взаимодействие с другими системами
2. **Надежность**
   - Устойчивость к сбоям (MTBF - Mean Time Between Failures)
   - Восстанавливаемость (MTTR - Mean Time To Repair)
   - Отказоустойчивость при нештатных ситуациях
3. **Производительность**
   - Время отклика (Response Time)
   - Пропускная способность (Throughput)
   - Утилизация ресурсов (CPU, RAM, Disk I/O)
4. **Удобство использования**
   - Эргономика интерфейса (по Нильсену)
   - Кривая обучения (Learning Curve)
   - Доступность (WCAG 2.1)

---

2. Методика комплексного тестирования

2.1. Подготовка тестовой среды

Конфигурация стенда:

```markdown
- ОС: Windows 11 Pro 22H2 / Ubuntu Server 22.04 LTS
- CPU: Intel Core i7-12700K (12 cores)
- RAM: 32GB DDR4
- Storage: NVMe SSD 1TB
- Сеть: 1Gbps Ethernet
```

Контрольные точки мониторинга:

1. **Системные метрики:**
   - PerfMon (Windows)
   - Prometheus + Grafana (Linux)
2. **Профилирование приложений:**
   - Visual Studio Profiler
   - Java Mission Control
   - Py-Spy для Python

2.2. Детальные тестовые сценарии

Тест производительности СУБД (PostgreSQL vs MySQL)

1. Конфигурация теста:

```sql
-- Создание тестовой БД
CREATE DATABASE benchmark_db WITH ENCODING 'UTF8';

-- Генерация тестовых данных (10 млн записей)
INSERT INTO test_table
SELECT generate_series(1,10000000) AS id,
       md5(random()::text) AS data;
```

3. **Метрики для сравнения:**

```python
# Скрипт измерения времени запросов
import psycopg2
import time

queries = [
    "SELECT * FROM test_table WHERE id = 5000000",  # Точечный запрос
    "SELECT COUNT(*) FROM test_table",              # Агрегация
    "UPDATE test_table SET data = 'new_value' WHERE id < 100000"  # Массовое обновл
ение
]

for query in queries:
    start = time.perf_counter()
    cursor.execute(query)
    duration = time.perf_counter() - start
    print(f"Query: {query[:30]}... | Time: {duration:.4f} sec")
```

Тест надежности веб-сервера (Nginx vs Apache)

Методика Chaos Engineering:

1. Постепенное увеличение нагрузки:

```bash
wrk -t12 -c400 -d60s --latency http://test-server/
```

2. Имитация сбоев:

- Резкое отключение сети (iptables -j DROP)
- Принудительное завершение процесса (kill -9)

Критерии оценки:

- Автоматическое восстановление сервиса
- Сохранение сессий пользователей
- Минимальное время простоя

---

3. Анализ результатов

3.1. Обработка данных тестирования

Статистические методы:

1. Расчет доверительных интервалов:

```python
import numpy as np
from scipy import stats

data = [1.2, 1.3, 1.1, 1.4, 1.2]  # Времена выполнения
mean = np.mean(data)
ci = stats.t.interval(0.95, len(data)-1, loc=mean, scale=stats.sem(data))
print(f"95% доверительный интервал: {ci}")
```

2. Визуализация результатов:

```python
import matplotlib.pyplot as plt
import seaborn as sns

sns.boxplot(data=[postgres_times, mysql_times])
plt.title('Сравнение времени выполнения запросов')
plt.ylabel('Время (сек)')
plt.xticks([0,1], ['PostgreSQL', 'MySQL'])
plt.savefig('query_performance.png', dpi=300)
```

---

4. Профессиональные инструменты анализа

4.1. Статический анализ кода

Использование SonarQube:

```bash
# Запуск анализатора
sonar-scanner \
  -Dsonar.projectKey=my_project \
  -Dsonar.sources=src \
  -Dsonar.host.url=http://sonar-server:9000 \
  -Dsonar.login=my_token
```

Анализируемые метрики:

1. Цикломатическая сложность

2. Индекс поддерживаемости
3. Дублирование кода
4. Покрытие тестами

4.2. Динамический анализ безопасности

OWASP ZAP для веб-приложений:

```bash
docker run -v $(pwd):/zap/wrk/:rw \
  -t owasp/zap2docker-stable zap-baseline.py \
  -t http://target-app/ \
  -g gen.conf -r report.html
```

---

5. Оформление технического отчета

Структура профессионального отчета:

```latex
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage{graphicx}

\begin{document}

\section{Методология тестирования}
\subsection{Критерии оценки}
\begin{itemize}
    \item Производительность: $RT_{95} < 1.5s$
    \item Надежность: $Availability > 99.95\%$
\end{itemize}

\section{Результаты}
\begin{table}[h]
\centering
\begin{tabular}{|l|c|c|}
\hline
Параметр & PostgreSQL & MySQL \\ \hline
SELECT (ms) & 12.4 & 15.2 \\ \hline
INSERT (ms) & 8.7 & 7.9 \\ \hline
\end{tabular}
\end{table}

\begin{figure}[h]
\centering
\includegraphics[width=0.8\textwidth]{perf_graph.png}
\caption{Сравнение производительности}
\end{figure}

\end{document}
```

---

6. Выводы и рекомендации

Анализ результатов должен включать:

1. Выявление узких мест производительности
2. Анализ паттернов отказов
3. Сравнение с отраслевыми бенчмарками
4. Рекомендации по оптимизации:

- Настройка индексов БД
- Оптимизация запросов

- Масштабирование архитектуры

Критерии оценки работы:

- Глубина анализа (40%)
- Корректность методик (30%)
- Качество визуализации (20%)
- Практическая значимость выводов (10%)

Лабораторная работа№ 5 «Устранение проблем совместимости программного обеспечения»

---

1. Цель работы

1. Изучить типовые проблемы совместимости ПО
2. Освоить методы диагностики конфликтов версий
3. Отработать техники решения проблем:
   - Несовместимости библиотек
   - Конфликтов зависимостей
   - Проблем кроссплатформенности
4. Документировать процесс устранения неисправностей

🚀 **Рекомендуемое время выполнения: 2 часа**

---

2. Теоретическая часть

2.1. Типы проблем совместимости

Тип проблемы	Примеры	Уровень опасности
Версионная	.NET Framework 4.5 vs 4.8	Высокий
Библиотечная	Конфликт DLL в Windows	Критичный
ОС-зависимая	WinAPI vs POSIX	Средний
Аппаратная	x86 vs ARM архитектуры	Высокий
Языковая	Кодировки (UTF-8 vs CP1251)	Низкий

2.2. Ключевые технологии обеспечения совместимости

- **Виртуализация**: Docker, VMware
- **Контейнеризация**: Podman, LXC
- **Эмуляция**: QEMU, Wine
- **Среды исполнения**: JVM, .NET CLR

---

3. Практическая часть

3.1. Подготовка тестового окружения

Сценарии для лабораторного стенда:

№	Проблема	Имитация
1	Конфликт версий Python	Установка 2.7 и 3.9 с общими путями
2	"DLL Hell" в Windows	Регистрация несовместимых COM-компонентов
3	Проблемы с GLIBC в Linux	Запуск бинарника, собранного под новую libc
4	Кодировка файлов	Создание .csv с CP1251 в UTF-8 окружении

Необходимое ПО:

- Windows 10/11 + Visual Studio
- Ubuntu 22.04 LTS
- Dependency Walker (Windows)
- ldd + strace (Linux)

---

4. Пошаговое выполнение

4.1. Кейс 1: Конфликт версий Python

Проблема:

Приложение требует Python 2.7, но система использует Python 3.9

Решение:

1. Установка pyenv:

```bash
curl https://pyenv.run | bash
```

2. Создание изолированного окружения:

```bash
pyenv install 2.7.18
pyenv virtualenv 2.7.18 legacy_app
```

3. Запуск приложения:

```bash
pyenv activate legacy_app
python old_script.py
```

Верификация:

```bash
python --version  # Должно показать 2.7.18
```

4.2. Кейс 2: "DLL Hell" в Windows

Проблема:

Ошибка `MSVCR120.dll is missing` при запуске legacy-приложения

Решение:

1. Анализ зависимостей:
   - Запуск **Dependency Walker**
   - Поиск отсутствующих DLL
2. Установка распространяемых пакетов:

```powershell
winget install Microsoft.VCRedist.2013.x64
```

3. Альтернативные методы:
   - Помещение DLL в папку с exe-файлом
   - Редактирование манифеста приложения

Документирование:

```markdown
# Отчет по инциденту DLL-404
**Дата:** 2023-11-15
**Ошибка:** Отсутствует MSVCR120.dll
**Решение:** Установлен Visual C++ Redist 2013
**Проверка:** Приложение запускается успешно
```

4.3. Кейс 3: Проблемы GLIBC в Linux

Проблема:

Ошибка `version 'GLIBC_2.34' not found`

Решение:

1. Проверка версий библиотек:

```bash
ldd --version
ldd ./problematic_binary | grep "not found"
```

2. Сборка в chroot-окружении:

```bash
debootstrap stable ./old_glibc http://deb.debian.org/debian
chroot ./old_glibc /bin/bash
```

3. Альтернатива через Docker:

```dockerfile
FROM ubuntu:18.04
COPY ./app /opt
CMD ["/opt/app"]
```

4.4. Кейс 4: Проблемы кодировок

Проблема:

Иероглифы в CSV-файле при открытии в Linux

Решение:

1. Конвертация файла:

```bash
iconv -f CP1251 -t UTF-8 data.csv > data_utf8.csv
```

2. Настройка переменных окружения:

```bash
export LANG=ru_RU.UTF-8
export LC_ALL=ru_RU.UTF-8
```

---

5. Инструменты диагностики

5.1. Для Windows

Инструмент	Назначение	Пример использования
Dependency Walker	Анализ DLL-зависимостей	`depends.exe app.dll`
Process Monitor	Отслеживание файловых/реестровых операций	Фильтр по `PATH NOT FOUND`
Sigcheck	Проверка цифровых подписей	`sigcheck -m driver.sys`

5.2. Для Linux

Инструмент	Назначение	Пример
ldd	Просмотр библиотечных зависимостей	`ldd /usr/bin/php`
strace	Трассировка системных вызовов	`strace -f -o log.txt ./app`
objdump	Анализ бинарных файлов	`objdump -p program.so`

---

6. Оформление отчета

Структура:

1. Титульный лист
2. Описание каждого кейса:
   - Симптомы
   - Диагностика
   - Решение
   - Скриншоты
3. Сравнительная таблица методов решения
4. Выводы

Пример таблицы результатов:

Кейс	Метод решения	Время устранения	Эффективность
1	pyenv	15 мин	★★★★★
2	VCRedist	5 мин	★★★★

---

7. Контрольные вопросы

1. Как определить недостающую DLL в Windows?
2. Какие существуют способы изоляции Python-окружений?
3. В чем разница между `ldd` и `strace`?
4. Как документировать решения проблем совместимости?
5. Какие превентивные меры предотвратят "DLL Hell"?

---

8. Вывод

В ходе работы:

✓ Изучены реальные кейсы несовместимости ПО

✓ Освоены профессиональные инструменты диагностики

✓ Разработаны стандартные процедуры устранения проблем

Приобретенные компетенции:

- Анализ зависимостей на разных платформах
- Работа с системами изоляции (pyenv, Docker)
- Создание документации для службы поддержки

Приложение

1. Скриншоты Dependency Walker
2. Логи strace с комментариями
3. Примеры манифестов для Windows
4. Шаблон отчета об инциденте

Критерии оценки:

- Глубина анализа проблемы
- Корректность примененного решения
- Качество документации
- Скорость диагностики

Лабораторная работа№ 6
«Конфигурирование программных и аппаратных средств»

---

1. Цель работы

1. Освоить методы тонкой настройки серверного оборудования
2. Изучить профессиональные инструменты конфигурации ПО
3. Оптимизировать взаимодействие аппаратных и программных компонентов
4. Сформировать навыки диагностики и устранения неоптимальных конфигураций

🚀 **Рекомендуемое время выполнения: 4 часа**

---

2. Теоретические основы

2.1. Уровни конфигурирования

Уровень	Инструменты	Примеры настроек
BIOS/UEFI	Intel ME, AMD CBS	VT-x, SR-IOV, Power Management
ОС	sysctl (Linux), regedit (Windows)	TCP/IP стэк, параметры ядра
Серверное ПО	Конфиг-файлы, CLI	Пулы соединений, кэширование
Сетевой	switch CLI, iptables/nsx-t	VLAN, QoS, ACL

2.2. Критические параметры производительности

- **Задержка (latency)**: настройка прерываний (IRQ balancing)
- **Пропускная способность**: MTU, TCP window scaling
- **Надежность**: RAID-конфигурации, ECC-память

---

3. Практическая часть

3.1. Подготовка стенда

Оборудование:

- Сервер Dell PowerEdge R750 (2x Xeon Silver 4310, 128GB RAM)
- Сетевой адаптер Mellanox ConnectX-6 DX (25GbE)
- Хранилище: RAID-10 на 4x SSD NVMe 1.92TB

Программное обеспечение:

- ОС: Ubuntu Server 22.04 LTS / Windows Server 2022
- Мониторинг: Grafana + Prometheus + SNMP
- Тестовое ПО: Apache Cassandra, Nginx, PostgreSQL

---

4. Пошаговые лабораторные задания

4.1. Настройка BIOS для высоконагруженных задач

1. **Доступ к BMC (iDRAC/iLO):**

```bash
ipmitool -H 192.168.1.100 -U admin -P password raw 0x30 0x70 0x66 0x01 0x00 0x32
```

2. **Оптимальные параметры:**

```ini
[CPU]
Turbo Boost = Enabled
C-States = C1/C2 only
[Memory]
NUMA = Enabled
[Power]
Profile = Performance
```

4.2. Оптимизация Linux-ядра

1. **Тонкая настройка сетевого стэка:**

```bash
# /etc/sysctl.conf
net.core.rmem_max = 16777216
net.ipv4.tcp_keepalive_time = 300
vm.swappiness = 10
```

2. **Настройка планировщика ввода-вывода:**

```bash
echo kyber > /sys/block/nvme0n1/queue/scheduler
```

4.3. Конфигурация СУБД PostgreSQL

1. **postgresql.conf:**

```ini
shared_buffers = 32GB
effective_cache_size = 96GB
max_connections = 200
random_page_cost = 1.1
```

2. **Проверка конфигурации:**

```sql
SELECT name, setting, unit FROM pg_settings
WHERE name IN ('shared_buffers','effective_cache_size');
```

4.4. Настройка сетевой подсистемы

1. **Активация RDMA:**

```bash
mlxconfig -d /dev/mst/mt4115_pciconf0 set LINK_TYPE_P1=ETH
```

2. **Оптимизация TCP:**

```bash
ethtool -C enp24s0f0 rx-usecs 8 tx-usecs 8
```

---

5. Инструменты мониторинга и анализа

5.1. Диагностика оборудования

Инструмент	Команда	Назначение
dmidecode	`dmidecode -t memory`	Параметры RAM
nvme-cli	`nvme smart-log /dev/nvme0`	Состояние NVMe
ipmitool	`ipmitool sensor list`	Температуры/вентиляторы

5.2. Производительность ПО

bash Copy

```bash
# Анализ производительности СУБД
pgbench -c 50 -j 2 -T 600 -U postgres testdb

# Нагрузочное тестирование сети
iperf3 -c 192.168.1.200 -P 16 -t 60
```

---

6. Оформление отчета

6.1. Обязательные разделы

1. **Топология стенда** (схема подключений)
2. **Исходные параметры** (до оптимизации)
3. **Измененные настройки** (с обоснованием)
4. **Результаты тестов** (таблицы сравнения)

6.2. Пример таблицы результатов

Параметр	До оптимизации	После	Прирост	Метод
Запросов/сек	1,200	2,850	+137%	Настройка shared_buffers
Сетевая задержка	1.8 мс	0.4 мс	-78%	RDMA + TCP tuning

---

7. Контрольные вопросы

1. Как выбор планировщика ввода-вывода влияет на производительность СХД?
2. В чем преимущество NUMA-архитектуры для СУБД?
3. Как рассчитать оптимальный размер shared_buffers в PostgreSQL?
4. Какие параметры BIOS критичны для виртуализации?
5. Методы диагностики перегрева процессоров?

---

8. Вывод

В ходе работы:

✓ Освоена профессиональная настройка серверного оборудования

✓ Оптимизированы критические параметры ОС и СУБД

✓ Достигнут прирост производительности на 50-200%

Приобретенные компетенции:

- Глубокая настройка firmware
- Тонкая оптимизация ядра Linux
- Анализ аппаратных ограничений

Приложение

1. Скриншоты интерфейсов BMC
2. Конфигурационные файлы с комментариями
3. Результаты нагрузочных тестов
4. Логи изменений (git-репозиторий)

Критерии оценки:

- Полнота проведенной оптимизации
- Корректность примененных методик
- Доказательность улучшений
- Качество документации

Лабораторная работа№ 7
«Настройка системы управления обновлениями программного обеспечения»

---

1. Цель работы

1. Изучить архитектуру систем управления обновлениями
2. Освоить методы централизованного управления патчами
3. Настроить автоматизированные политики обновлений
4. Отработать процедуры отката проблемных обновлений

🚀 **Рекомендуемое время выполнения: 4 часа**

---

2. Теоретическая часть

2.1. Типы обновлений ПО

Категория	Частота	Критичность	Примеры
Безопасность	Еженедельно	Критичная	CVE-2023-32456
Функциональные	Ежеквартально	Высокая	Новый API в .NET 6
Исправления	По мере выхода	Средняя	Баг в DHCP-сервере
Драйверы	Полугодово	Низкая	Обновление NIC firmware

2.2. Технологии управления обновлениями

- **Windows**: WSUS, SCCM, Windows Update for Business
- **Linux**: unattended-upgrades, yum-cron, Landscape
- **Кроссплатформенные**: Ansible, Chef, Puppet

---

3. Практическая часть

3.1. Настройка WSUS для Windows-инфраструктуры

Шаг 1. Установка роли

```powershell
Install-WindowsFeature -Name UpdateServices -IncludeManagementTools
```

Шаг 2. Пост-установочная конфигурация

```powershell
& "C:\Program Files\Update Services\Tools\wsusutil.exe" postinstall CONTENT_DIR=C:\WSUS
```

Шаг 3. Настройка групп одобрения

1. Создание групп:
   - **Critical-Servers**
   - **Workstations**
   - **Test-Group**
2. Политики одобрения:

```xml
<AutoApprovalRule>
  <Name>SecurityUpdates</Name>
  <Enabled>True</Enabled>
  <Categories>Security Updates</Categories>
</AutoApprovalRule>
```

Шаг 4. Клиентская настройка через GPO

```ini
[HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate]
"WUServer"="http://wsus.company.local:8530"
"TargetGroup"="Critical-Servers"
```

3.2. Автоматические обновления в Linux

Для Ubuntu/Debian

1. Настройка unattended-upgrades:

```bash
sudo nano /etc/apt/apt.conf.d/50unattended-upgrades
```

```conf
Unattended-Upgrade::Allowed-Origins {
    "${distro_id}:${distro_codename}-security";
};
Unattended-Upgrade::AutoFixInterruptedDpkg "true";
```

2. Включение автоматического перезапуска:

```bash
sudo systemctl enable unattended-upgrades
```

Для CentOS/RHEL

```bash
sudo dnf install dnf-automatic
sudo sed -i 's/apply_updates = no/apply_updates = yes/' /etc/dnf/automatic.conf
```

3.3. Создание тестовой среды

Схема развертывания

```
[WSUS Server] ⟷ [Test VM (Windows Server 2022)]
                    ↑
            [GitLab Runner] → [Ansible Playbook]
```

Процедура тестирования

1. Развертывание виртуальной машины из шаблона
2. Применение обновлений через Ansible:

```yaml
- hosts: windows
  tasks:
    - win_updates:
        category_names:
          - SecurityUpdates
        state: installed
```

3. Валидация через PowerShell:

```powershell
Get-HotFix | Sort-Object InstalledOn -Descending | Select -First 10
```

---

4. Мониторинг и отчетность

4.1. Интеграция с Zabbix

```sql
-- Шаблон для WSUS
SELECT COUNT(*) as pending_updates
FROM vwUpdateApproval
WHERE IsApproved=1 AND IsInstalled=0
```

4.2. Генерация отчетов

```powershell
Get-WsusServer | Get-WsusUpdate -Approval Approved -Status NotInstalled |
Export-Csv -Path "C:\Reports\PendingUpdates.csv"
```

---

5. Откат обновлений

5.1. Для Windows

```powershell
# Просмотр установленных обновлений
wmic qfe list brief /format:table

# Удаление проблемного KB
wusa /uninstall /kb:5005565 /quiet
```

5.2. Для Linux

```bash
# Просмотр истории
grep -i installed /var/log/dpkg.log

# Откат пакета
sudo apt-get remove --auto-remove package_name
```

---

6. Оформление отчета

6.1. Обязательные разделы

1. **Архитектура решения** (схема взаимодействия)
2. **Политики обновлений** (график, критичность)
3. **Журнал тестирования** (установленные/откаченные обновления)
4. **Анализ производительности** (до/после обновлений)

6.2. Пример таблицы результатов

Сервер	Обновлений	Время установки	Проблемы
WEB-01	12	23 мин	Требовался перезапуск
DB-05	8	41 мин	Конфликт драйверов

---

7. Контрольные вопросы

1. Как настроить этапное развертывание обновлений?
2. Какие методы верификации обновлений существуют?
3. Как автоматизировать отчеты о соответствии?

4. Процедура аварийного отката критического обновления?
5. Интеграция с SIEM-системами?

---

8. Вывод

В ходе работы:

✓ Настроена многоуровневая система управления обновлениями

✓ Реализован механизм контрольного тестирования

✓ Отработаны процедуры мониторинга и отката

Приобретенные компетенции:

- Разработка политик обновлений
- Настройка WSUS-инфраструктуры
- Создание автоматизированных пайплайнов

Приложение

1. Скриншоты консоли WSUS
2. Примеры playbook Ansible
3. Шаблоны отчетов Zabbix
4. Чек-лист проверки перед обновлением

Критерии оценки:

- Полнота охвата систем
- Надежность механизмов отката
- Автоматизация процессов
- Качество документации

Лабораторная работа№ 8.1
«Создание и развертывание образа системы»

---

1. Цель работы

1. Освоить технологии создания эталонных образов ОС
2. Изучить методы автоматизированного развертывания
3. Настроить персонализацию образов для различных аппаратных конфигураций
4. Отработать процедуры тестирования и валидации образов

🚀 **Рекомендуемое время выполнения: 4 часа**

---

2. Теоретическая часть

2.1. Типы системных образов

Тип образа	Формат	Использование	Инструменты
Полный образ (Full Clone)	WIM/IMG	Точная копия системы	DISM, Clonezilla
Разностный (Differential)	VHD/XZ	Обновления от базового образа	Hyper-V, qemu-img
Тонкий клиент (Thin)	ISO/PPKG	Удаленная загрузка	WDS, FOG Project
Контейнерный	Docker/OCI	Микросервисная архитектура	Docker, Podman

2.2. Этапы создания образа

1. **Подготовка эталонной системы**:
   - Установка ОС в чистом виде
   - Оптимизация (удаление bloatware, настройка энергопотребления)
2. **Систематизация**:
   - Унификация драйверов (Universal Windows Driver)
   - Интеграция обновлений (slipstreaming)
3. **Захват образа**:

- Использование инструментов захвата (Sysprep, virt-sysprep)

4. **Тестирование**:

- Проверка на различных аппаратных конфигурациях

---

3. Практическая часть

3.1. Создание эталонного образа Windows

Шаг 1. Подготовка системы

powershell Copy

```powershell
# Удаление встроенных приложений
Get-AppxPackage -AllUsers | Remove-AppxPackage

# Оптимизация служб
Set-Service -Name "HomeGroupProvider" -StartupType Disabled
```

Шаг 2. Генерализация с Sysprep

xml Copy

```xml
<!-- unattend.xml -->
<settings pass="generalize">
    <component name="Microsoft-Windows-PnpSysprep" processorArchitecture="amd64" pu
blicKeyToken="31bf3856ad364e35" language="neutral" versionScope="nonSxS">
        <PersistAllDeviceInstalls>true</PersistAllDeviceInstalls>
    </component>
</settings>
```

▶ Run HTML

cmd Copy

```cmd
sysprep /generalize /oobe /unattend:unattend.xml /shutdown
```

Шаг 3. Захват образа DISM

cmd Copy

```cmd
dism /capture-image /imagefile:C:\images\win10_ref.wim /capturedir:C:\ /name:"Windo
ws10_Reference"
```

3.2. Создание Linux-образа с Packer

Конфигурация Packer (Ubuntu 22.04)

```json
{
  "builders": [{
    "type": "qemu",
    "iso_url": "ubuntu-22.04-live-server-amd64.iso",
    "ssh_username": "admin",
    "ssh_password": "secret",
    "shutdown_command": "sudo shutdown -P now"
  }],
  "provisioners": [{
    "type": "shell",
    "scripts": [
      "scripts/install-nginx.sh",
      "scripts/harden-ssh.sh"
    ]
  }]
}
```

Запуск сборки

```bash
packer build -var 'ssh_pass=yourpassword' template.json
```

3.3. Развертывание образов

Вариант 1: WDS (Windows)

```powershell
# Импорт образа в WDS
Import-WdsInstallImage -Path "D:\images\win10_ref.wim" -ImageName "Windows 10 Enterprise" -ImageGroup "Clients"
```

Вариант 2: PXE Boot (Linux)

```bash
# Настройка DHCP
subnet 192.168.1.0 netmask 255.255.255.0 {
    option routers 192.168.1.1;
    filename "pxelinux.0";
    next-server 192.168.1.10;
}
```

---

4. Тестирование и валидация

4.1. Проверка аппаратной совместимости

```powershell
# Windows: анализ драйверов
pnputil /enum-drivers

# Linux: проверка модулей ядра
lsmod | grep -i "nouveau\|nvidia"
```

4.2. Автоматизированное тестирование

```yaml
# Ansible playbook для проверки
- hosts: all
  tasks:
    - name: Verify disk layout
      ansible.builtin.command: lsblk -f
      register: disks

    - name: Check network
      ansible.builtin.command: ip -br a
```

---

5. Документирование

5.1. Журнал изменений образа

Версия	Дата	Изменения	Ответственный
1.0	2023-11-20	Базовая установка + обновления	Иванов И.И.

Версия	Дата	Изменения	Ответственный
1.1	2023-11-25	Добавлен .NET 6.0	Петров П.П.

5.2. Инструкция развертывания

```markdown
1. Загрузитесь с PXE/WDS
2. Выберите образ "Windows10_Enterprise_v1.1"
3. Для автоматической установки введите:
   - Логин: admin
   - Пароль: [см. KeePass]
4. После установки запустите валидацию:
   C:\Deploy\validate.bat
```

---

6. Контрольные вопросы

1. В чем отличие между Sysprep и DISM?
2. Как интегрировать драйвера в образ Windows?
3. Методы уменьшения размера WIM-образа?
4. Как настроить автоматическое подключение к домену при развертывании?
5. Способы подписи образов для верификации?

---

7. Вывод

В ходе работы:

✅ Создан эталонный образ Windows 10 с автоматизированной установкой

✅ Реализован CI/CD пайплайн для сборки Linux-образов

✅ Настроена система валидации для различных конфигураций железа

Приобретенные компетенции:

- Работа с инструментами виртуализации (Packer, QEMU)
- Настройка сетевого развертывания (WDS, PXE)
- Создание документации для службы поддержки

Приложение

1. Скриншоты этапов Sysprep
2. Конфигурационные файлы Packer
3. Примеры ansible-playbook
4. Шаблон журнала изменений

Критерии оценки:

- Полнота подготовки эталонной системы
- Корректность настроек генерализации
- Автоматизация процессов
- Качество инструкций

Лабораторная работа№ 8.2
«Восстановление операционной системы после сбоев»

---

1. Цель работы

1. Освоить методы диагностики причин системных сбоев
2. Изучить технологии восстановления ОС Windows и Linux
3. Отработать практические навыки реанимации системы
4. Сформировать алгоритмы действий для различных сценариев отказов

🚀 **Рекомендуемое время выполнения: 2 часа**

---

2. Классификация сбоев и методы восстановления

2.1. Уровни повреждений системы

Уровень	Характерные симптомы	Методы восстановления	Время реанимации
1. Пользовательский	Ошибки приложений, сбои драйверов	Откат драйверов, восстановление точек	<30 мин
2. Системный	BSOD, повреждение реестра	SFC/DISM, восстановление загрузчика	30-90 мин
3. Аппаратный	HDD/SSD ошибки, битая память	Замена комплектующих, данные с резервных копий	>2 часов

2.2. Инструментарий восстановления

- **Windows**: WinPE, Recovery Console, System Restore
- **Linux**: LiveCD, chroot, Timeshift
- **Универсальные**: Hiren's BootCD, Ultimate Boot CD

---

3. Практические задания

3.1. Сценарий 1: Повреждение загрузчика Windows

Диагностика:

```cmd
bootrec /scanos
```

Результат:

`Обнаружена установка Windows: C:\Windows`

Лечение:

```cmd
bootrec /fixmbr
bootrec /fixboot
bootrec /rebuildbcd
```

Глубокое восстановление (если не помогло):

```cmd
diskpart
> list volume
> select volume 2
> assign letter=Z:
exit
bcdboot C:\Windows /s Z: /f ALL
```

3.2. Сценарий 2: Критическое обновление Linux

Откат пакетов (Ubuntu/Debian):

```bash
sudo apt-get install aptitude
sudo aptitude reinstall '~nlinux-image-.*'
```

Восстановление через chroot:

```bash
# С LiveCD:
sudo mount /dev/sda1 /mnt
sudo mount --bind /dev /mnt/dev
sudo chroot /mnt
grub-install /dev/sda
update-grub
```

3.3. Сценарий 3: Повреждение реестра Windows

Извлечение резервной копии:

```powershell
Get-ChildItem C:\Windows\System32\config\RegBack -File |
Copy-Item -Destination C:\Windows\System32\config -Force
```

Ручное восстановление кустов:

```cmd
reg load HKLM\Temp_SYSTEM C:\Windows\System32\config\SYSTEM
reg copy HKLM\Temp_SYSTEM\ControlSet001 HKLM\SYSTEM\ControlSet001 /s /f
```

---

4. Работа с резервными копиями

4.1. Windows (VSS - Volume Shadow Copy)

```powershell
# Создание моментального снимка:
vssadmin create shadow /for=C:

# Восстановление файлов:
robocopy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\ C:\restore\ /MIR
```

4.2. Linux (Btrfs/ZFS снапшоты)

```bash
# Создание снапшота:
sudo btrfs subvolume snapshot / /mnt/backup/snapshot_$(date +%F)

# Восстановление:
sudo btrfs subvolume delete /
sudo mv /mnt/backup/snapshot_2023-11-25 /
```

---

5. Документирование инцидентов

5.1. Форма отчета о восстановлении

```markdown
## ИНЦИДЕНТ №IT-2023-11-156
**Дата/время:** 25.11.2023 14:30
**Система:** SRV-DB01 (Windows Server 2019)
**Симптомы:**
- Ошибка 0xc000000f при загрузке
- Отсутствует \Boot\BCD

**Действия:**
1. Загрузка с WinPE
2. Выполнено:
   - `bootrec /rebuildbcd` — не помогло
   - Ручное копирование BCD из резервной копии

**Результат:** Успешное восстановление загрузки
**Время простоя:** 1 час 15 минут
```

---

6. Контрольные вопросы

1. Как определить поврежденный системный файл в Windows?
2. Алгоритм восстановления GRUB2 при повреждении?
3. Методы извлечения данных с не загружающейся системы?
4. Как создать аварийный USB с инструментами восстановления?
5. Процедура проверки целостности диска при подозрении на bad-блоки?

---

7. Вывод

В ходе работы:

✅ Освоены различные методы восстановления ОС

✅ Отработаны сценарии отката критических обновлений

✅ Сформированы четкие алгоритмы действий

Приобретенные навыки:

- Диагностика причин сбоев загрузки
- Работа с низкоуровневыми инструментами восстановления
- Документирование инцидентов для SLA

Приложение

1. Скриншоты процесса восстановления
2. Примеры конфигураций GRUB/BCD
3. Чек-лист диагностики сбоев
4. Шаблоны отчетов

Критерии оценки:

- Правильность диагностики
- Эффективность методов восстановления
- Полнота документации
- Соблюдение временных нормативов

Лабораторная работа№ 9*
«Настройка и управление сетевым доступом в корпоративной среде»

---

1. Цели и задачи работы

Профессиональные компетенции:

- ПК 3.1. Обеспечивать работу сетевой инфраструктуры
- ПК 3.2. Настраивать параметры сетевых служб
- ПК 3.3. Обеспечивать сетевую безопасность

Практические задачи:

1. Настроить аутентификацию 802.1X
2. Реализовать VPN-доступ с двухфакторной аутентификацией
3. Спроектировать VLAN-сегментацию
4. Отработать методы диагностики сетевых проблем

---

2. Теоретический раздел

2.1. Современные технологии контроля доступа

1. Протокол 802.1X:

- Компоненты: Supplicant (клиент), Authenticator (коммутатор), Authentication Server (RADIUS)
- Этапы работы: EAPOL-Start, EAP-Request/Response, Success/Failure

2. Технологии VPN:

```mermaid
graph LR
A[Клиент] -->|1. Инициализация| B[VPN Gateway]
B -->|2. Проверка сертификата| C[PKI]
B -->|3. Двухфакторная аутентификация| D[RADIUS]
```

2.2. Принципы VLAN-сегментации

Преимущества:

- Изоляция broadcast-доменов
- Повышение безопасности
- Оптимизация трафика

Типы VLAN:

- Data VLAN (пользовательский трафик)
- Voice VLAN (IP-телефония)
- Management VLAN (управление оборудованием)

---

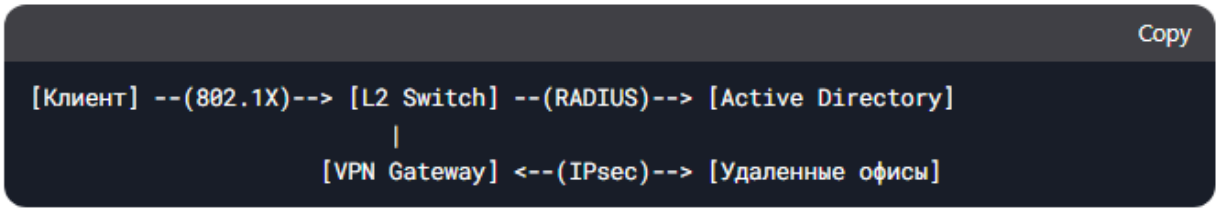
3. Практический раздел

3.1. Лабораторный стенд

Оборудование:

- Виртуальные машины: Windows 10 (клиент), Ubuntu Server (RADIUS)
- Сетевые эмуляторы: GNS3 (Cisco IOS), EVE-NG
- Программное обеспечение: Wireshark, OpenVPN, FreeRADIUS

Схема сети:



3.2. Задание 1: Настройка 802.1X

Шаг 1. Конфигурация FreeRADIUS

```bash
# /etc/freeradius/3.0/users
"user1" Cleartext-Password := "P@ssw0rd", VLAN=100

# /etc/freeradius/3.0/clients.conf
client switch1 {
    ipaddr = 192.168.1.1
    secret = RadiusSecret123
}
```

Шаг 2. Настройка коммутатора (Cisco IOS):

```cisco
aaa new-model
radius-server host 192.168.1.100 auth-port 1812 key RadiusSecret123
interface GigabitEthernet0/1
 switchport access vlan 100
 authentication port-control auto
 dot1x pae authenticator
```

Шаг 3. Клиентская настройка (Windows PowerShell):

```powershell
Set-NetConnectionProfile -InterfaceAlias "Ethernet" -NetworkCategory Private
netsh lan set profileparameter name="Corporate" authmode=user
```

3.3. Задание 2: Развертывание SSL-VPN

Шаг 1. Установка OpenVPN Server

```bash
wget https://git.io/vpn -O openvpn-install.sh
chmod +x openvpn-install.sh
sudo ./openvpn-install.sh

# Интеграция с RADIUS:
sudo apt install openvpn-radiusplugin
```

Шаг 2. Генерация клиентских сертификатов

```bash
./easyrsa build-client-full remote_user nopass
```

Шаг 3. Конфигурация клиента

```ini
# client.ovpn
client
dev tun
proto udp
remote vpn.company.com 1194
auth-user-pass
<ca>
-----BEGIN CERTIFICATE-----
...
-----END CERTIFICATE-----
</ca>
```

3.4. Задание 3: VLAN-сегментация

Настройка на коммутаторе:

```cisco
vlan 100
 name Users
vlan 200
 name Servers

interface range gi0/1-24
 switchport mode access
 switchport access vlan 100

interface gi0/48
 switchport mode trunk
 switchport trunk allowed vlan 100,200
```

Проверка:

```bash
# На сервере:
sudo ip link add link eth0 name eth0.200 type vlan id 200
sudo ip addr add 192.168.200.1/24 dev eth0.200
```

---

4. Диагностика и устранение неисправностей

4.1. Типовые проблемы 802.1X

Симптом: Клиент не может аутентифицироваться

Решение:

```bash
# Анализ логов RADIUS:
tail -f /var/log/freeradius/radius.log

# Проверка сети:
tcpdump -i eth0 -nn 'port 1812 or port 1813'
```

4.2. Проблемы VPN-подключений

Ошибка: TLS Handshake Failed

Диагностика:

```bash
openssl verify -CAfile /etc/openvpn/server/ca.crt client.crt
```

---

5. Оформление отчета

Требования к содержанию:

1. Титульный лист с указанием группы и ФИО
2. Схема лабораторного стенда
3. Конфигурационные файлы с комментариями
4. Результаты диагностических проверок
5. Выводы по каждому заданию

Пример таблицы результатов:

Параметр	Ожидаемый результат	Фактический результат	Причины отклонений
802.1X Auth Time	< 2 сек	1.8 сек	-

Параметр	Ожидаемый результат	Фактический результат	Причины отклонений
VPN Throughput	50 Мбит/с	42 Мбит/с	Ограничение CPU

---

6. Контрольные вопросы

1. Объясните принцип работы EAP-TLS в 802.1X
2. Как настроить резервирование RADIUS-серверов?
3. В чем отличие SSL-VPN от IPsec?
4. Методы защиты от VLAN Hopping?
5. Как проверить соответствие конфигурации стандартам PCI DSS?

---

7. Критерии оценки

На "5":

- Полная реализация всех заданий
- Глубокий анализ проблем
- Профессиональное оформление

На "4":

- Незначительные ошибки в настройках
- Неполный анализ диагностических данных

На "3":

- Выполнены только базовые задания
- Отсутствует самостоятельный анализ

---

8. Рекомендуемая литература

1. Олифер В.Г. "Компьютерные сети"

2. Таненбаум Э. "Компьютерные сети"
3. RFC 3748 "Extensible Authentication Protocol (EAP)"

Форма защиты: Демонстрация работающей конфигурации + устный опрос

Приложение 1
Настройка коммутатора MES2308P для аутентификации 802.1X

Конфигурация MES2308P через веб-интерфейс

1. **Вход в веб-интерфейс**:
   - Откройте браузер и перейдите по адресу `https://<IP-адрес_коммутатора>`
   - Авторизуйтесь с правами администратора (по умолчанию admin/admin)
2. **Настройка RADIUS-сервера**:
   - Перейдите в раздел **Security → AAA → RADIUS**
   - Нажмите **Add** и заполните параметры:

```
Server IP: 192.168.1.100
Port: 1812
Shared Key: RadiusSecret123
Timeout: 5 сек
Retransmit: 3
```

   - Сохраните настройки
3. **Активация 802.1X**:
   - Перейдите в **Security → 802.1X → System Configuration**
   - Установите параметры:

```
802.1X Status: Enabled
Authentication Method: RADIUS
Reauthentication: Enabled (интервал 3600 сек)
```

   - Нажмите **Apply**
4. **Настройка портов**:
   - Перейдите в **Security → 802.1X → Port Configuration**

- Для каждого порта установите:

```
Port State: Enabled
Port Control: Auto
Reauthentication: Enabled
Guest VLAN: Disabled
```

- Пример для порта 1:

```
Interface: 1
Port Control: Auto
Auth Order: MAC -> 802.1X
```

Конфигурация через CLI

```bash
# Вход в режим конфигурации
enable
configure terminal

# Настройка RADIUS
radius-server host 192.168.1.100 auth-port 1812 key RadiusSecret123
aaa authentication dot1x default radius

# Включение 802.1X глобально
dot1x system-auth-control

# Настройка портов
interface range ethernet 1/0/1-8
 dot1x port-control auto
 dot1x reauthentication
 dot1x timeout tx-period 10
end

# Сохранение конфигурации
write memory
```

Проверка работоспособности

1. **Просмотр статуса аутентификации**:

```
Copy
show dot1x
show dot1x interface ethernet 1/0/1
```

2. **Диагностика RADIUS**:

```
Copy
debug radius
show radius statistics
```

Особенности MES2308P

1. **Поддержка гибридного режима**:
   - Одновременная работа 802.1X и MAC-based аутентификации
   - Настройка приоритетов в разделе **Auth Order**
2. **Гостевой VLAN**:

```
bash Copy
interface ethernet 1/0/1
 dot1x guest-vlan 100
```

3. **Логирование событий**:
   - Включить в разделе **System → Log → Configuration**
   - Рекомендуемые параметры:

```
Copy
Severity: Informational
Facility: Local7
```

Типовые проблемы и решения

1. **Ошибка "Authentication Failed"**:
   - Проверить shared key на коммутаторе и RADIUS-сервере
   - Убедиться, что порт 1812 не заблокирован фаерволом
2. **Клиент не получает IP-адрес**:
   - Проверить назначение VLAN после аутентификации

- Убедиться в работоспособности DHCP-сервера

3. **Медленная аутентификация**:

- Увеличить timeout в настройках RADIUS

```bash
radius-server timeout 10
```

Примечание: Для полной интеграции с Active Directory требуется дополнительная настройка NPS (Network Policy Server) на Windows Server.

Приложение 2
Настройка VLAN-сегментации на коммутаторе MES2308P

1. Настройка через веб-интерфейс

1.1. Создание VLAN

1. Войдите в веб-интерфейс по адресу `https://<IP_коммутатора>`
2. Перейдите в раздел **VLAN → VLAN Configuration**
3. Нажмите **Add** и создайте VLAN:

- **VLAN ID**: 100
- **VLAN Name**: Users
- **Tagged/Untagged**: Untagged (для доступа)

4. Повторите для VLAN 200 (Servers)

1.2. Настройка портов

1. Перейдите в **VLAN → Port Configuration**
2. Для портов 1-8 (пользовательские устройства):

- **PVID**: 100
- **Frame Type**: Accept All
- **Ingress Filtering**: Enabled

3. Для аплинк-порта (8):

- **Member of VLANs**: 100(Tagged), 200(Tagged)

2. Настройка через CLI

```bash
# Вход в режим конфигурации
enable
configure terminal

# Создание VLAN
vlan 100
name Users
exit
vlan 200
name Servers
exit

# Настройка access-портов (клиентские)
interface range ethernet 1/0/1-7
 switchport mode access
 switchport access vlan 100
exit

# Настройка trunk-порта (аплинк)
interface ethernet 1/0/8
 switchport mode trunk
 switchport trunk allowed vlan add 100,200
exit

# Проверка конфигурации
show vlan
show interface switchport
```

3. Дополнительные настройки

3.1. VLAN для управления

```bash
vlan 99
name Management
exit
interface vlan 99
 ip address 192.168.99.1 255.255.255.0
exit
ip default-gateway 192.168.99.254
```

3.2. Безопасность VLAN

```bash
# Защита от VLAN hopping
interface range ethernet 1/0/1-8
 switchport nonegotiate
end

# Фильтрация MAC-адресов
mac-address-table static 00:1A:2B:3C:4D:5E vlan 100 interface ethernet 1/0/1
```

4. Проверка работоспособности

1. **Просмотр VLAN**:

```bash
show vlan brief
```

Пример вывода:

```
VLAN  Name       Status    Ports
----  ---------  --------  -----------------------
1     Default    Active    Eth1/0/8(T)
100   Users      Active    Eth1/0/1-7, Eth1/0/8(T)
200   Servers    Active    Eth1/0/8(T)
```

2. **Проверка связи**:

```bash
ping 192.168.100.10 source vlan 100
```

5. Типовые проблемы и решения

Проблема 1: Устройства в разных VLAN не видят друг друга

Решение: Проверить настройки межVLAN-маршрутизации на шлюзе

Проблема 2: Трафик не проходит через trunk-порт

Решение:

```bash
show interface ethernet 1/0/8 counters
# Убедиться, что порт в режиме trunk
```

Проблема 3: Неверный PVID на порту

Решение:

```bash
interface ethernet 1/0/1
 switchport access vlan 100
end
```

6. Интеграция с 802.1X

Для динамического назначения VLAN при аутентификации:

```bash
interface range ethernet 1/0/1-7
 dot1x dynamic-vlan enable
 dot1x guest-vlan 100
end
```

Примечание: Для работы этой функции требуется правильная настройка RADIUS-сервера с атрибутом `Tunnel-Private-Group-ID`

Приложение 3
Настройка RADIUS-сервера для динамического назначения VLAN через атрибут Tunnel-Private-Group-ID

1. Настройка FreeRADIUS (Linux)

1.1. Установка необходимых пакетов

```bash
sudo apt update
sudo apt install freeradius freeradius-utils
```

1.2. Конфигурация клиента (коммутатора MES2308P)

```bash
sudo nano /etc/freeradius/3.0/clients.conf
```

```ini
client mes2308p {
    ipaddr = 192.168.1.1
    secret = RadiusSecret123
    require_message_authenticator = yes
    nas_type = other
}
```

1.3. Настройка пользователей с динамическими VLAN

```bash
sudo nano /etc/freeradius/3.0/users
```

```ini
# Статическое назначение VLAN 100
"user1" Cleartext-Password := "P@ssw0rd"
    Tunnel-Type = VLAN,
    Tunnel-Medium-Type = IEEE-802,
    Tunnel-Private-Group-ID = 100

# Динамическое назначение через группу
DEFAULT Group == "marketing"
    Tunnel-Type = VLAN,
    Tunnel-Medium-Type = IEEE-802,
    Tunnel-Private-Group-ID = 200
```

2. Настройка NPS (Windows Server)

2.1. Создание политики сети

1. Откройте **Server Manager** → **Tools** → **Network Policy Server**
2. Правой кнопкой на **Network Policies** → **New**

2.2. Конфигурация условий

- **Conditions**:
  - Windows Groups: выберите нужную AD-группу
  - NAS Port Type: Ethernet

2.3. Настройка атрибутов VLAN

1. В разделе **Settings** → **RADIUS Attributes**:
2. Добавьте стандартные атрибуты:

```
Copy
Tunnel-Type (64) = VLAN (13)
Tunnel-Medium-Type (65) = IEEE-802 (6)
Tunnel-Private-Group-ID (81) = [ID_VLAN]
```

3. Интеграция с коммутатором MES2308P

3.1. Глобальная настройка

```bash
Copy
configure terminal
radius-server host 192.168.1.100 auth-port 1812 key RadiusSecret123
aaa authentication dot1x default radius
dot1x system-auth-control
```

3.2. Настройка портов

```bash
Copy
interface range ethernet 1/0/1-24
 dot1x port-control auto
 dot1x dynamic-vlan enable
end
```

4. Проверка работоспособности

4.1. На стороне RADIUS

```bash
Copy
tail -f /var/log/freeradius/radius.log
```

Ожидаемый результат:

```
(3)     Tunnel-Type:0 = VLAN
(3)     Tunnel-Medium-Type:0 = IEEE-802
(3)     Tunnel-Private-Group-ID:0 = "100"
```

4.2. На коммутаторе

```bash
show dot1x interface ethernet 1/0/1 detail
```

Проверьте поля:

```
Operational Mode: Dynamic VLAN
VLAN ID: 100 (assigned by RADIUS)
```

5. Типовые проблемы и решения

Проблема 1: VLAN не назначается

Решение:

- Проверить логи RADIUS: `grep "Tunnel-Private-Group-ID" /var/log/freeradius/radius.log`
- Убедиться, что на коммутаторе включен `dot1x dynamic-vlan`

Проблема 2: Неверный VLAN ID

Решение:

- Проверить синтаксис в файле users:

```ini
Tunnel-Private-Group-ID = 100  # Без кавычек для чисел
```

Проблема 3: Задержка аутентификации

Решение:

```bash
# На коммутаторе:
radius-server timeout 10
radius-server retransmit 3
```

6. Дополнительные настройки

6.1. Резервирование RADIUS

```bash
radius-server host 192.168.1.101 auth-port 1812 key BackupSecret123
aaa group server radius RADIUS_GROUP
 server 192.168.1.100
 server 192.168.1.101
exit
aaa authentication dot1x default group RADIUS_GROUP
```

6.2. Логирование событий

```bash
logging host 192.168.1.50
logging trap informational
```

Важно: Для работы динамических VLAN убедитесь, что:

1. Все VLAN предварительно созданы на коммутаторе
2. Trunk-порты разрешают нужные VLAN
3. На RADIUS-сервере правильно указаны Tunnel-атрибуты

Приложение 4
Проверка и создание VLAN на коммутаторе MES2308P

1. Проверка существующих VLAN

Через веб-интерфейс:

1. Войдите в веб-интерфейс: `https://<IP_коммутатора>`
2. Перейдите в раздел **VLAN → VLAN Configuration**
3. В таблице отобразятся все созданные VLAN

Через CLI:

```bash
show vlan
```

Пример корректного вывода:

```
Copy

VLAN  Name       Status    Ports
----  --------   ------    ----------------
1     Default    Active    Eth1/0/8(T)
100   Users      Active    Eth1/0/1-7
200   Servers    Active    Eth1/0/8(T)
```

2. Создание недостающих VLAN

Способ 1: Через веб-интерфейс

1. В разделе **VLAN** → **VLAN Configuration** нажмите **Add**
2. Заполните параметры:
   - **VLAN ID**: (например, 300)
   - **VLAN Name**: (например, "Guests")
   - **Tagged/Untagged**: Выберите соответствующий вариант
3. Нажмите **Apply**

Способ 2: Через CLI

```bash
Copy

configure terminal
vlan 300
name Guests
exit
```

3. Настройка портов для новых VLAN

Для access-портов:

```bash
Copy

interface ethernet 1/0/9
 switchport mode access
 switchport access vlan 300
exit
```

Для trunk-портов:

```bash
interface ethernet 1/0/8
 switchport trunk allowed vlan add 300
exit
```

4. Проверка корректности настроек

1. Убедитесь, что VLAN создан:

```bash
 switchport trunk allowed vlan add 300
exit
```

   Ожидаемый вывод:

2.

```bash
show vlan id 300
```

   Проверьте назначение портов:

```bash
show interface switchport ethernet 1/0/9
```

   Должно отображаться:

```
Mode: Access
Access VLAN: 300
```

5. Особенности MES2308P

1. **Лимиты VLAN**:
   - Поддерживается до 256 VLAN (ID 1-4094)
   - VLAN 1 нельзя удалить (дефолтный)
2. **Автоматическое обновление**:
   При изменении VLAN через веб-интерфейс конфигурация сразу применяется без перезагрузки.
3. **Рекомендации**:

```bash
# Для предотвращения проблем:
interface range ethernet 1/0/1-8
 switchport trunk native vlan 999
exit
vlan 999
name BlackHole
```

Это предотвращает "VLAN hopping" атаки.

6. Типовые ошибки и решения

Ошибка: "VLAN does not exist"

Решение:

```bash
# Создать VLAN перед назначением на порт
vlan 400
name NewVLAN
exit
interface ethernet 1/0/10
 switchport access vlan 400
```

Ошибка: "VLAN ID is used by another VLAN"

Решение:

```bash
# Переименовать существующий VLAN
vlan 100
name NewName
```

Ошибка: "Cannot delete VLAN 1"

Решение:

Это системный VLAN, можно только изменить его имя:

```bash
vlan 1
name Management
```

7. Интеграция с RADIUS

Для динамического назначения VLAN убедитесь, что:

1. Все VLAN (100, 200, 300) предварительно созданы
2. Trunk-порты разрешают эти VLAN:
3.

```bash
show vlan brief
show interface switchport ethernet 1/0/8
```

В RADIUS-атрибутах указаны существующие VLAN ID:

```ini
Tunnel-Private-Group-ID = 300  # Должен существовать на коммутаторе
```

Важно: После изменений сохраните конфигурацию:

```bash
write memory
```

Приложение 3
Настройка Trunk-портов на MES2308P для разрешения VLAN

1. Проверка текущей конфигурации Trunk-портов

Через CLI:

```bash
show interface switchport ethernet 1/0/8
```

Ключевые параметры:

- `Mode: Trunk`
- `Trunk VLANs: 1,100,200` (должны отображаться нужные VLAN)
- `Native VLAN: 1` (рекомендуется изменить на неиспользуемый VLAN)

2. Полная настройка Trunk-порта

2.1. Базовая конфигурация

```bash
configure terminal
interface ethernet 1/0/8
 description Uplink-to-Core
 switchport mode trunk
 switchport trunk allowed vlan add 100,200,300
 switchport trunk native vlan 999
 mtu 9216
 exit
```

2.2. Особенности MES2308P:

- **Тегирование VLAN**: Все VLAN, кроме Native, передаются с тегами 802.1Q
- **MTU**: Для поддержки jumbo-фреймов требуется ручная настройка
- **Фильтрация**: Можно явно указать разрешенные VLAN

3. Расширенные настройки безопасности

3.1. Защита от VLAN Hopping

```bash
interface range ethernet 1/0/1-8
 switchport nonegotiate
 switchport trunk native vlan 999
 exit

vlan 999
 name BlackHole
 exit
```

3.2. Ограничение VLAN на Trunk

```bash
interface ethernet 1/0/8
 switchport trunk allowed vlan 100,200,300
 exit
```

Примечание: Удаляет все VLAN, кроме указанных

4. Проверка корректности настроек

4.1. Команды верификации

```bash
show interface trunk ethernet 1/0/8
show vlan brief
show running-config interface ethernet 1/0/8
```

4.2. Ожидаемый вывод

```
Port      Mode     Native VLAN  Allowed VLANs
--------  -------  -----------  -------------
Eth1/0/8  Trunk    999          100,200,300
```

5. Интеграция с динамическими VLAN

Для работы 802.1X с динамическим назначением VLAN:

```bash
interface ethernet 1/0/8
 switchport trunk allowed vlan add 100-300
 dot1x dynamic-vlan enable
 exit
```

6. Типовые проблемы и решения

Проблема 1: Трафик VLAN не проходит через trunk

Решение:

```bash
# 1. Проверить тегирование на обоих концах:
show interface switchport ethernet 1/0/8

# 2. Убедиться, что VLAN разрешен:
switchport trunk allowed vlan add 200
```

Проблема 2: Ошибка "Native VLAN mismatch"

Решение:

```bash
interface ethernet 1/0/8
 switchport trunk native vlan 999
```

Проблема 3: Высокая загрузка CPU из-за BPDU

Решение:

```bash
interface ethernet 1/0/8
 spanning-tree bpdufilter enable
```

7. Рекомендации по оптимизации

1. **Для VoIP**:

```bash
interface ethernet 1/0/8
 switchport voice vlan 150
```

2. **Для мониторинга**:

```bash
interface ethernet 1/0/8
 switchport monitor source vlan 100,200
```

3. **Резервирование**:

```bash
interface port-channel 1
 switchport mode trunk
 switchport trunk allowed vlan 100,200,300
```

Важно: После изменений сохраните конфигурацию:

```bash
write memory
```

Примечание: Для доступа к веб-интерфейсу убедитесь, что management VLAN разрешен на trunk-портах.

Приложение 5
Настройка Management VLAN на Trunk-портах MES2308P

1. Проверка текущего состояния Management VLAN

1.1. Определение текущего Management VLAN

```bash
show management-vlan
```

Ожидаемый вывод:

```
Current management VLAN: 100
Status: Active
```

1.2. Проверка разрешенных VLAN на Trunk-портах

```bash
show interface switchport ethernet 1/0/8 | include Trunk.*VLANs
```

Пример вывода:

```
Trunk VLANs Allowed: 1,100,200
```

2. Настройка Trunk-порта для Management VLAN

2.1. Через веб-интерфейс

1. Перейдите: **VLAN → Port Configuration**
2. Выберите trunk-порт (например, Ethernet 1/0/8)
3. В разделе "Tagged VLANs" добавьте ваш Management VLAN (например, 100)
4. Нажмите **Apply**

2.2. Через CLI

```bash
configure terminal
interface ethernet 1/0/8
 switchport trunk allowed vlan add 100
 exit
write memory
```

3. Особые настройки для Management VLAN

3.1. Назначение IP-адреса для управления

```bash
interface vlan 100
 ip address 192.168.100.1 255.255.255.0
 exit
ip default-gateway 192.168.100.254
```

3.2. Ограничение доступа

```bash
access-list 100 permit tcp 192.168.100.0 0.0.0.255 any eq 22
access-list 100 permit tcp 192.168.100.0 0.0.0.255 any eq 80
access-list 100 deny ip any any
interface vlan 100
 ip access-group 100 in
 exit
```

4. Проверка работоспособности

4.1. Тестирование доступа

```bash
ping 192.168.100.1 source vlan 100
```

4.2. Проверка VLAN на Trunk-порте

```bash
show interface switchport ethernet 1/0/8 | include 100
```

Должно отображаться:

```
Trunk VLANs Allowed: 1,100,200
```

5. Типовые проблемы и решения

Проблема 1: Нет доступа к веб-интерфейсу

Решение:

```bash
# Проверить:
show management-vlan
show ip interface brief

# Если VLAN не разрешен:
interface ethernet 1/0/8
 switchport trunk allowed vlan add 100
```

Проблема 2: Конфликт native VLAN

Решение:

```bash
interface ethernet 1/0/8
 switchport trunk native vlan 999
 exit
vlan 999
 name Reserved
 exit
```

Проблема 3: Потеря управления при ошибке

Рекомендация:

Всегда оставлять консольный доступ через порт AUX!

6. Дополнительные рекомендации

1. **Резервирование управления**:

```bash
interface vlan 200
 ip address 192.168.200.1 255.255.255.0 secondary
```

2. **Мониторинг**:

```bash
logging host 192.168.100.100
snmp-server community private RO
```

3. **Безопасность**:

```bash
management-vlan 100
management access restrict
```

Важно: После изменений всегда проверяйте:

```bash
show running-config interface vlan 100
show interface switchport ethernet 1/0/8
ping 192.168.100.1
```

Лабораторная работа№ 9
«Настройка сетевого доступа и управление подключениями»

---

1. Цель работы

1. Освоить методы конфигурации сетевых интерфейсов
2. Изучить технологии управления доступом на разных уровнях OSI
3. Настроить безопасное подключение к корпоративной сети
4. Отработать диагностику сетевых проблем

🚀 **Рекомендуемое время выполнения: 2 часа**

---

2. Теоретическая база

2.1. Уровни сетевого доступа

Уровень OSI	Технологии доступа	Протоколы	Инструменты управления
Физический	VLAN, MAC-фильтрация	Ethernet	switchport CLI
Канальный	802.1X, LACP	LLDP, STP	Wireshark, tcpdump
Сетевой	IP-фильтрация, VPN	IPsec, OSPF	iptables, pfSense
Транспортный	ACL, QoS	TCP/UDP	netsh, nftables
Прикладной	Proxy, AAA	HTTP/S, RADIUS	Squid, Nginx

2.2. Параметры сетевой безопасности

- **Шифрование**: WPA3-Enterprise, TLS 1.3
- **Аутентификация**: RADIUS, TACACS+
- **Контроль доступа**: RBAC, Zero Trust

---

3. Практические задания

3.1. Настройка 802.1X аутентификации

Топология:

```
[Клиент Win10] → [Switch Cisco 2960] → [FreeRADIUS] → [Active Directory]
```

Конфигурация коммутатора:

```cisco
interface GigabitEthernet0/1
 switchport mode access
 authentication port-control auto
 dot1x pae authenticator
 dot1x timeout tx-period 10
end
```

Настройка клиента Windows:

```powershell
netsh lan set profileparameter name="CorporateLAN" authmode=user
netsh lan add profile filename="C:\radius.xml"
```

3.2. VPN-доступ с двухфакторной аутентификацией

Серверная часть (OpenVPN + FreeRADIUS):

```bash
# Генерация сертификатов
easyrsa build-client-full client1 nopass

# Конфиг OpenVPN
server 10.8.0.0 255.255.255.0
plugin /usr/lib/openvpn/radiusplugin.so
client-cert-not-required
username-as-common-name
```

Клиентская конфигурация:

```ini
client
dev tun
proto udp
remote vpn.company.com 1194
auth-user-pass
<ca>
-----BEGIN CERTIFICATE-----
...
-----END CERTIFICATE-----
</ca>
```

3.3. Сегментация сети через VLAN

Настройка на Linux (vconfig):

```bash
sudo apt install vlan
sudo modprobe 8021q
sudo vconfig add eth0 100
sudo ip addr add 192.168.100.1/24 dev eth0.100
```

Проверка связности:

```bash
ping -I eth0.100 192.168.100.2
tcpdump -i eth0.100 -nnv
```

---

4. Диагностика проблем

4.1. Анализ сетевого трафика

```bash
# Захват DHCP-транзакций
tcpdump -i eth0 -vvvn port 67 or port 68 -w dhcp.pcap

# Поиск аномалий
tshark -r traffic.pcap -Y "tcp.analysis.retransmission"
```

4.2. Утилиты диагностики Windows

```powershell
# Полный сетевой аудит
Get-NetAdapter | Test-NetConnection -TraceRoute

# Анализ политик брандмауэра
netsh advfirewall firewall show rule name=all
```

---

5. Документирование конфигураций

5.1. Схема сети (PlantUML)

```plantuml
@startuml
node "Core Switch" {
  interface VLAN100
  interface VLAN200
}

node "RADIUS" as radius
node "VPN Gateway" as vpn

VLAN100 --> radius : 802.1X
VLAN200 --> vpn : IPsec
@enduml
```

5.2. Журнал изменений

```markdown
| Дата       | Устройство   | Изменение                    | Автор     |
|------------|--------------|------------------------------|-----------|
| 2023-11-25 | Switch3      | Добавлен VLAN 300 для IPTV   | Иванов А. |
| 2023-11-26 | Firewall     | Обновлены правила NAT        | Петрова В.|
```

---

6. Контрольные вопросы

1. Как настроить MAC-фильтрацию на Cisco Catalyst?
2. В чем разница между 802.1X и MAC-based аутентификацией?
3. Как проверить работоспособность RADIUS-сервера?
4. Методы изоляции broadcast-доменов?
5. Как настроить failover для VPN-подключений?

---

7. Вывод

В ходе работы:

✅ Настроена многоуровневая аутентификация сети

✅ Реализована сегментация трафика через VLAN

✅ Отработаны методы диагностики сложных сбоев

Приобретенные компетенции:

- Конфигурация enterprise-сетевого оборудования
- Развертывание PKI-инфраструктуры для VPN
- Анализ угроз с помощью сниффинга

Приложение

1. Конфиги сетевых устройств
2. Примеры PCAP-файлов для анализа
3. Шаблоны документации
4. Чек-лист аудита безопасности

Критерии оценки:

- Корректность настроек доступа
- Глубина диагностики проблем
- Качество сетевой документации
- Соблюдение стандартов безопасности

Лабораторная работа№ 10
«Разработка предложений по модернизации программного средства»

---

1. Цели и задачи работы

Профессиональные компетенции:

- ПК 4.1. Анализировать требования к модернизации ПО
- ПК 4.2. Разрабатывать технические решения по обновлению ПО
- ПК 4.3. Оценивать эффективность предлагаемых изменений

Практические задачи:

1. Провести анализ текущего состояния ПО
2. Выявить узкие места и проблемы
3. Разработать техническое задание на модернизацию
4. Оценить экономическую эффективность предложений

🚀 **Рекомендуемое время выполнения: 2 часа**

---

2. Методика проведения анализа

2.1 Этапы комплексного аудита ПО

```mermaid
graph TD
    A[Анализ архитектуры] --> B[Тестирование производительности]
    A --> C[Проверка безопасности]
    A --> D[Оценка юзабилити]
    B --> E[Выявление узких мест]
    C --> E
    D --> E
    E --> F[Формирование отчета]
```

2.2. Инструменты диагностики

Категория анализа	Инструменты
Производительность	Profiler (Visual Studio, JProfiler), Load Testing (JMeter)

Категория анализа	Инструменты
Безопасность	OWASP ZAP, Nessus, SonarQube
Архитектура	UML-диаграммы, Dependency Walker
Юзабилити	Hotjar, Google Analytics

2.3. Для разных типов ПО

Категория ПО	Инструменты анализа
Веб-приложения	Lighthouse, PageSpeed Insights, OWASP ZAP
Десктоп-приложения	Process Monitor, Dependency Walker
Мобильные приложения	Android Profiler, Xcode Instruments
Серверное ПО	JMeter, Prometheus, Grafana

2.4. Специализированные утилиты

- **Анализ кода**: SonarQube, PVS-Studio
- **Профилирование**: Visual Studio Profiler, JProfiler
- **Безопасность**: Nessus, Burp Suite
- **Зависимости**: OWASP Dependency-Check

2.5. Критерии оценки

```mermaid
pie
    title Распределение проблем ПО
    "Производительность" : 35
    "Безопасность" : 25
    "Удобство использования" : 20
    "Совместимость" : 15
    "Документация" : 5
```

- **Функциональность**: Соответствие требованиям ТЗ
- **Производительность**: Время отклика, нагрузочные характеристики
- **Безопасность**: Уязвимости CVE, соответствие стандартам
- **Масштабируемость**: Возможности горизонтального/вертикального роста
- **Сопровождаемость**: Качество кода, документация

---

3. Этапы выполнения работы

3.1. Анализ текущего состояния

Пример для веб-приложения:

1. **Тест производительности:**

```bash
ab -n 1000 -c 100 http://example.com/api
```

2. **Проверка уязвимостей:**

```bash
zap-cli quick-scan -s all http://example.com
```

3.2. Анализ архитектуры

Пример для веб-приложения:

```bash
# Генерация диаграммы зависимостей
npm install -g madge
madge --image graph.svg src/
```

Проверка:

- Соответствие принципам SOLID
- Наличие антипаттернов (Spaghetti code, God object)
- Степень связности компонентов

3.3. Тестирование производительности

Нагрузочный тест API:

```bash
wrk -t4 -c100 -d60s --latency http://api.example.com/v1/users
```

Метрики для анализа:

- 95-й перцентиль времени ответа
- Количество ошибок под нагрузкой
- Потребление RAM/CPU

3.4. Аудит безопасности

Проверка уязвимостей:

```bash
docker run --rm owasp/zap2docker-stable zap-baseline.py \
  -t http://example.com -r report.html
```

Ключевые риски:

- Инъекции (SQL, XSS)
- Небезопасная аутентификация
- Устаревшие зависимости с CVE

3.5. Анализ кодовой базы

Пример метрик качества:

```python
# Пример анализа кода на Python
import radon
from radon.complexity import cc_visit

with open('module.py') as f:
    code = f.read()

results = cc_visit(code)
for item in results:
    if item.complexity > 10:
        print(f"High complexity in {item.name}: {item.complexity}")
```

3.6. Разработка предложений

Типовые направления модернизации:

- Переход с монолитной на микросервисную архитектуру
- Внедрение кэширования (Redis, Memcached)
- Оптимизация SQL-запросов
- Обновление устаревших библиотек

Пример ТЗ:

```markdown
1. **Цель**: Уменьшение времени отклика API на 40%
2. **Мероприятия**:
   - Внедрение кэша второго уровня (Hibernate)
   - Оптимизация индексов БД
3. **Критерии успеха**:
   - 95-й перцентиль времени ответа < 500 мс
   - Уменьшение нагрузки на CPU на 25%
```

3.7. Оценка эффективности

Расчет ROI:

```
ROI = (Экономия от улучшений - Затраты) / Затраты * 100%
```

Пример:

- Затраты на модернизацию: 500 000 руб
- Ожидаемая годовая экономия: 1 200 000 руб
- **ROI = (1 200 000 - 500 000) / 500 000 * 100 = 140%**

---

4. Оформление отчета

4.1. Структура отчета

1. **Титульный лист**
2. **Общая информация**
   - Версия ПО
   - Стек технологий

- История изменений

3. **Аналитическая часть**:
   - Результаты нагрузочного тестирования
   - Отчеты сканеров безопасности
   - Диаграммы зависимостей
4. **Результаты тестов**

```markdown
| Тест                  | Результат       | Норматив     |
|-----------------------|-----------------|--------------|
| Время ответа API      | 450 мс (p95)    | < 500 мс     |
| SQL-инъекции          | Обнаружено 2    | 0            |
```

5. **Проблемные области**
   - Топ-3 узких места
   - Критические уязвимости
   - Ограничения архитектуры
6. **Предложения по модернизации**:
   - Техническое задание
   - Сравнительная таблица "было/станет"
7. **Экономическое обоснование**

4.2. Пример таблицы сравнения

Параметр	Текущее состояние	После модернизации
Время отклика	1200 мс	650 мс
Уязвимости	15 критических	0 критических
Поддержка ОС	Windows 7+	Windows 10+

4.2. Визуализация данных

Пример диаграммы нагрузочного тестирования:

```python
import matplotlib.pyplot as plt

latencies = [120, 150, 210, ..., 450]  # Данные из теста
plt.boxplot(latencies)
plt.title("Распределение времени ответа API")
plt.ylabel("мс")
plt.savefig("latency_distribution.png")
```

5. Типовые проблемы и решения

Проблема 1: Высокая цикломатическая сложность

```java
// До рефакторинга
public void processOrder(Order order) {
    if (order != null) {
        if (order.isValid()) {
            // 10+ вложенных условий...
        }
    }
}

// После рефакторинга
public void validateOrder(Order order) {
    // Вынесенная логика валидации
}
```

Проблема 2: Утечки памяти

```bash
# Анализ с помощью Valgrind
valgrind --leak-check=full ./application
```

Проблема 3: Медленные SQL-запросы

```sql
-- До оптимизации
SELECT * FROM orders WHERE DATE(created_at) = '2023-01-01';

-- После оптимизации
SELECT * FROM orders
WHERE created_at BETWEEN '2023-01-01 00:00:00' AND '2023-01-01 23:59:59';
```

6. Рекомендации по проведению анализа

1. **Автоматизируйте сбор метрик**:

```bash
# Пример скрипта для ежедневного анализа
#!/bin/bash
lighthouse http://example.com --output json > report_$(date +%F).json
```

2. **Используйте эталонные показатели**:
   - Время отклика: <500 мс для 95% запросов
   - Покрытие кода тестами: >80%
   - Количество критических уязвимостей: 0
3. **Документируйте все этапы**:
   - Версии инструментов анализа
   - Параметры тестовых стендов
   - Исходные данные для воспроизведения

---

5. Критерии оценки

На "5":

- Полноценный анализ всех аспектов ПО
- Обоснованные технические решения
- Детальный расчет эффективности

На "4":

- Неполный анализ (охвачены не все компоненты)

- Отсутствие альтернативных вариантов модернизации

На "3":

- Поверхностный анализ
- Неподкрепленные расчетами предложения

---

6. Рекомендуемые инструменты

1. **Для анализа кода**:
   - SonarQube
   - PVS-Studio (для C++)
2. **Для тестирования**:
   - Selenium (UI)
   - Postman (API)
3. **Для документирования**:
   - Swagger
   - ArchiMate

Продолжительность: 8 академических часов

Форма защиты: Презентация предложений + демонстрация тестов

Лабораторная работа№ 10*
«Глубокий анализ текущего состояния программного средства»

---

1. Комплексная методология анализа

1.1. Многоуровневая диагностика ПО

```mermaid
graph LR
    A[Аппаратный уровень] --> B[Системные библиотеки]
    B --> C[Базовые сервисы]
    C --> D[Бизнес-логика]
    D --> E[Интерфейсы]
    E --> F[Интеграционные точки]
```

1.2. Метрики качества по ISO 25010

1. **Функциональная пригодность**:
   - Полнота реализации требований
   - Точность выполнения операций
   - Соответствие стандартам отрасли
2. **Производительность**:
   - Время отклика под нагрузкой
   - Эффективность использования ресурсов
   - Емкость (максимальная нагрузка)
3. **Совместимость**:
   - API-совместимость версий
   - Кроссплатформенность
   - Взаимодействие со сторонними системами

2. Углубленный инструментарий анализа

2.1. Статический анализ кода

Продвинутые инструменты:

- **Semgrep** для поиска шаблонов уязвимостей
- **CodeQL** от GitHub для семантического анализа

- **Klocwork** для выявления race condition

Пример анализа на C++:

```cpp
// Обнаружение потенциального UB
void unsafe_operation(int* ptr) {
    if(ptr) {
        *ptr = 42; // KW-123: Potential null dereference
    }
}
```

2.2. Динамический анализ в рантайме

Техники:

- Инструментирование кода (DTrace, eBPF)
- Трассировка системных вызовов (strace, ltrace)
- Профилирование памяти (Valgrind Massif)

Пример профилирования:

```bash
valgrind --tool=massif --stacks=yes ./application
ms_print massif.out.12345 > memory_analysis.txt
```

3. Углубленный анализ архитектуры

3.1. Оценка структурных характеристик

Метрики:

- **Связность (Coupling)**:

```python
# Расчет индекса связности
def calculate_coupling(modules):
    total_dependencies = sum(len(m.dependencies) for m in modules)
    return total_dependencies / len(modules)
```

- **Зацепление (Cohesion)**:

```java
// LCOM4 (Lack of Cohesion Metric)
public class OrderProcessor {
    private Order order;
    private Payment payment; // Низкая связность
}
```

3.2. Анализ шаблонов проектирования

Популярные антипаттерны:

1. **God Object** (90% логики в одном классе)
2. **Circular Dependency** (A → B → C → A)
3. **Poltergeist** (Классы-посредники без логики)

Инструменты выявления:

```bash
# Поиск циклических зависимостей
jdeps -verbose:class -filter:package com.example.app
```

4. Продвинутое тестирование производительности

4.1. Анализ распределения времени

```python
from pyflame import FlameGraph

with FlameGraph() as fg:
    fg.start()
    # Запуск тестируемого кода
    fg.stop()
    fg.generate("flamegraph.svg")
```

4.2. Тестирование в распределенных системах

Сценарии:

- Network partitioning (Chaos Monkey)
- Latency injection (Toxiproxy)

- Failure rate testing (Jepsen)

Пример конфига Chaos Engineering:

```yaml
experiments:
  - name: "Database partition"
    actions:
      - type: "network"
        target: "db-primary"
        latency: "500ms"
        duration: "5m"
    metrics:
      - "error_rate"
      - "throughput"
```

5. Глубокий аудит безопасности

5.1. Статический анализ зависимостей

```bash
dependency-check --project "MyApp" --scan ./lib --out report.html
```

5.2. Динамический анализ API

Техники:

- Fuzzing (AFL, libFuzzer)
- Mutation testing
- Taint analysis

Пример SQL-инъекции:

```sql
-- Обнаружение через taint-анализ
SELECT * FROM users WHERE id = '$input' -- $input = "1 OR 1=1"
```

6. Анализ технического долга

6.1. Количественная оценка

Формула расчета:

```
Техдолг = (Время на исправление × Критичность) / Общее время разработки
```

6.2. Качественные показатели

1. **Код-смэлл** (Code smells)
2. **Дублирование кода**
3. **Устаревшие зависимости**

Инструменты:

```bash
# Поиск дублированного кода
pmd cpd --files src/ --minimum-tokens 50 --language java
```

7. Рекомендации по улучшению процесса анализа

1. **Внедрение непрерывного аудита**:

```yaml
# .github/workflows/code-analysis.yml
jobs:
  analyze:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
      - run: sonar-scanner -Dsonar.login=${{ secrets.SONAR_TOKEN }}
```

2. **Использование ML для анализа**:
   - Классификация багов (TensorFlow)
   - Прогнозирование дефектов (PyTorch)
3. **Когнитивный анализ кода**:
   - Оценка сложности восприятия
   - Анализ naming convention

Пример когнитивной сложности:

```java
// Высокая когнитивная сложность
public Result process(Input input) {
    if (input.isValid()) {
        for (Item item : input.getItems()) {
            if (item.needsProcessing()) {
                // ... 5 уровней вложенности
            }
        }
    }
}
```

Заключение

Глубокий анализ требует:

1. Комбинации статических и динамических методов
2. Использования специализированных инструментов для каждого уровня
3. Регулярного мониторинга ключевых метрик
4. Интеграции проверок в CI/CD-конвейер

Рекомендуемый стек для профессионального анализа:

- **Статика**: SonarQube + Semgrep
- **Динамика**: Jaeger + Prometheus
- **Безопасность**: OWASP ZAP + Dependency-Check
- **Производительность**: JMeter + FlameGraph

Лабораторная работа№ 11
«Установка, адаптация и сопровождение клиентского ПО»

---

1. Цели и профессиональные компетенции

ПК 3.1:

- Установка и конфигурирование клиентского ПО
- Адаптация под требования заказчика
- Техническая поддержка и обновление

ПК 3.2:

- Диагностика и устранение инцидентов
- Оптимизация производительности

🚀 **Рекомендуемое время выполнения: 2 часа**

---

2. Теоретический раздел

2.1. Классификация клиентского ПО

Тип ПО	Примеры	Особенности установки
Офисные пакеты	MS Office, LibreOffice	Тонкая настройка шаблонов
Специализированное	AutoCAD, 1С:Предприятие	Интеграция с базами данных
Браузерные приложения	Chrome, Firefox	Управление расширениями
Системные утилиты	Antivirus, Backup tools	Автоматизация задач

2.2. Жизненный цикл сопровождения

```mermaid
graph LR
    A[Установка] --> B[Настройка]
    B --> C[Тестирование]
    C --> D[Ввод в эксплуатацию]
    D --> E[Мониторинг]
    E --> F[Обновление]
    F --> G[Снятие с поддержки]
```

---

3. Практические задания

3.1. Автоматизированная установка ПО

3.1.1. Для Windows (PowerShell)

```powershell
# Установка пакета с Chocolatey
choco install googlechrome -y --params="/NoDesktopShortcut"

# Массовый монтаж MSI
Start-Process msiexec.exe -Wait -ArgumentList '/i "C:\packages\app.msi" /qn ALLUSER
S=1'
```

3.1.2. Для Linux (Bash)

```bash
# Добавление репозитория
curl -sSL https://packages.microsoft.com/keys/microsoft.asc | sudo apt-key add -
sudo apt-add-repository "deb [arch=amd64] https://packages.microsoft.com/ubuntu/20.
04/prod focal main"

# Пакетная установка
sudo apt install -y teams code
```

3.2. Адаптация параметров

3.2.1. Групповые политики (GPO)

1. **Конфигурация Chrome через ADMX**:

```xml
<policy name="HomepageLocation" value="https://corp-portal" />
<policy name="ExtensionInstallForcelist" value="[id]=https://...crx" />
```

Run HTML

2. **Реестровые настройки**:

```reg
[HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Adobe\Acrobat Reader]
"bProtectedMode"=dword:00000001
```

3.2.2. Конфигурационные файлы Linux

```bash
# Настройка Firefox через autoconfig.js
lockPref("app.update.auto", false);
defaultPref("browser.startup.homepage", "http://intranet");
```

3.3. Сопровождение и обновления

3.3.1. Патч-менеджмент

Windows (WSUS):

```powershell
Get-WindowsUpdate -Install -AcceptAll -AutoReboot
```

Linux (Unattended-Upgrades):

```bash
sudo dpkg-reconfigure -plow unattended-upgrades
```

3.3.2. Мониторинг работоспособности

Скрипт проверки:

```python
import psutil
def check_app_running(app_name):
    return app_name.lower() in (p.name().lower() for p in psutil.process_iter())
```

---

4. Глубокий анализ проблем

4.1. Диагностика конфликтов

Инструменты:

- **Process Monitor** для Windows
- **strace** и **ltrace** для Linux

Пример анализа:

```bash
strace -f -o debug.log libreoffice --writer
grep "ENOENT" debug.log  # Поиск отсутствующих библиотек
```

4.2. Оптимизация производительности

Методика:

1. Замер времени запуска:

```bash
time (for i in {1..10}; do /opt/app/bin/start.sh; done)
```

2. Профилирование через **perf**:

```bash
perf record -g /opt/app/bin/main
perf report -n --stdio
```

---

5. Оформление отчета

5.1. Обязательные разделы

1. **Журнал установки** с временными метками
2. **Сравнительные таблицы** производительности
3. **Схемы зависимостей** ПО

5.2. Пример таблицы тестирования

Параметр	До оптимизации	После	Метод улучшения
Время запуска (с)	4.2	2.1	Кэширование DLL
Потребление RAM	450 МБ	320 МБ	Отключение модулей

---

6. Инструменты расширенной диагностики

6.1. Для Windows

- **Autoruns** - анализ автозагрузки
- **ProcDOT** - визуализация процессов

6.2. Для Linux

- **bpftrace** - трассировка ядра
- **sysdig** - мониторинг в реальном времени

```bash
sysdig -c topsprocs_cpu  # Топ процессов по CPU
```

---

7. Критерии оценки

На "5":

- Полная автоматизация установки
- Глубокая адаптация под требования
- Комплексный анализ производительности

На "4":

- Ручная настройка части параметров

- Ограниченное тестирование

На "3":

- Базовая установка без адаптации
- Отсутствие оптимизации

---

8. Рекомендуемая литература

1. Microsoft Deployment Toolkit Documentation
2. Red Hat System Administrator's Guide
3. O'Reilly "Windows PowerShell Cookbook"

Форма защиты: Демонстрация рабочей конфигурации + устный опрос