Контроль ИТ

Общество с ограниченной ответственностью «Контроль ИТ»

(ООО «Контроль ИТ»)

Методические указания для выполнения лабораторных работ в учебной лаборатории Л400 МТУСИ

Дата последнего изменения

27 января 2026 г.

Оглавление

Термины, сокращения и условные обозначения

АРМ – автоматизированное рабочее место.

БД – база данных.

ЗА – Зонд агрегации.

ЗМ – Зонд мониторинга.

КД, AS – коммутатор доступа (Access Switch).

ЛВС, LAN (Local Area Network) – локальная вычислительная сеть; система соединённых между собой устройств, таких как компьютеры, серверы, сетевые принтеры и другие периферийные устройства, расположенных в пределах одного здания или территории.

ИБП – источник бесперебойного питания.

ОС – операционная система.

Система КМУТ – система контроля, мониторинга и управления трафиком; **система мониторинга**.

СН, Сниффер трафика – устройство или программное обеспечение, которое используется для мониторинга сетевого трафика, анализа пакетов, протоколов или сети.

Трафик – совокупность IP-пакетов, передаваемых по сети связи.

ТШ – телекоммуникационный шкаф.

ЦОД – центр обработки данных.

ЦСИ – центральный сервер измерений.

AS (Autonomous System) – группа IP-сетей и маршрутизаторов, управляемых одной организацией и имеющих единую политику маршрутизации.

BGP (Boarder Gateway Protocol) – протокол динамической маршрутизации, предназначенный для маршрутизации между автономными системами.

BPDU (Bridge Protocol Data Unit) – протокол, с помощью которого коммутаторы обмениваются информацией в сети для управления её топологией.

C-VLAN ID (Customer VLAN ID) – идентификатор сервиса клиента VLAN.

CE (Customer Edge) – оборудование, устанавливаемое на площадке Клиента и используемое для маршрутизации трафика из ЛВС Клиента в сеть Оператора и обратно.

CPE (Customer Premises Equipment) – оборудование для соединения Клиента с публичной или частной сетью Оператора.

CLI (Command Line Interface) – способ взаимодействия между человеком и компьютером путём отправки компьютеру команд, представляющих собой последовательность символов.

DHCP (Dynamic Host Configuration Protocol) – протокол, позволяющий сетевым устройствам автоматически получать IP-адрес и другие параметры, необходимые для работы в сети TCP/IP.

Ethernet – семейство технологий пакетной передачи данных между устройствами для компьютерных и промышленных сетей.

GRE (Generic Routing Encapsulation) – протокол туннелирования сетевых пакетов.

GUI (Graphical User Interface) – графический пользовательский интерфейс.

ICMP (Internet Control Message Protocol) – протокол межсетевых управляющих сообщений, который входит в стек протоколов TCP/IP.

IP-пакет – базовый блок передачи данных протокола IP.

LSDB (Link-State Database) – база данных каждого маршрутизатора и статусе каналов, которая содержит актуальную информацию о состоянии сети.

NAT (Network Address Translation) – процесс перевода внутренних или локальных IP адресов во внешние IP адреса.

NetFlow – сетевой протокол, предназначенный для учёта сетевого трафика.

MIB (Management Information Base) – виртуальная база данных, используемая для управления объектами в сети связи.

ML (Modeling Language) — язык моделирования, описывающий с помощью диаграмм и схем разнообразные процессы и структуры.

OSPF (Open Shortest Path First) – протокол динамической маршрутизации, основанный на технологии отслеживания состояния канала и использующий для нахождения кратчайшего пути алгоритм Дейкстры.

PAT (Port Address Translation) – технология трансляции адресов с использованием портов.

PD (Packet Delay) – время, которое требуется пакету данных, чтобы пройти от источника к месту назначения в локальной вычислительной сети.

PDV (Packet Delay Variation) – показатель, который измеряет вариацию времени, необходимого для прохождения пакетов данных по локальной вычислительной сети от источника к назначению.

PE (Provider Edge) – пограничный маршрутизатор, на границе провайдер — клиент.

PL (Packet Loss) – показатель потери пакетов данных.

Q-in-Q – технология, позволяющая создать несколько изолированных сегментов внутри одного VLAN за счет добавления дополнительного тега к заголовку Ethernet-пакета.

QoS (Quality of Service) – технология предоставления различным классам трафика различных приоритетов в обслуживании.

RS-232 (Recommended Standard 232) – стандарт физического уровня для асинхронного интерфейса; последовательный порт вычислительных устройств.

RSTP (Rapid Spanning Tree Protocol) – версия протокола STP с ускоренной реконфигурацией дерева, использующегося для исключения петель (исключения дублирующих маршрутов) в соединениях коммутаторов Ethernet с дублирующими линиями.

RTT (Round-Trip Time) – время, затраченное на отправку сетевого пакета (сигнала), плюс время, которое требуется для подтверждения, что сетевой пакет (сигнал) был получен.

SLA (Service Level Agreement) – соглашение между заказчиком и исполнителем о том, какие, когда и как будут предоставляться услуги в IT и сфере телекоммуникаций.

SNMP (Simple Network Management Protocol) – протокол для управления устройствами в IP-сетях.

SP-VLAN ID (Service Provider VLAN ID) – идентификатор сервиса провайдера VLAN.

SSH (Secure Shell) – сетевой протокол прикладного уровня, позволяющий производить удалённое управление операционной системой и туннелирование соединений.

TCP (Transmission Control Protocol) – протокол контроля передачи.

TCP/IP (Transmission Control Protocol / Internet Protocol) – сетевая модель передачи данных, представленных в цифровом виде.

TTL (Time to live) – предельный период времени или число итераций или переходов, которые набор данных (пакет) может осуществить (прожить) до своего исчезновения.

USB (Universal Serial Bus) – последовательный интерфейс для подключения периферийных устройств к вычислительной технике.

UDP (User Datagram Protocol) – протокол транспортного уровня.

UTC (SU) – национальная шкала координированного времени Российской Федерации.

VGA (Video Graphics Array) — компонентный видеоинтерфейс, используемый в мониторах и видеоадаптерах.

VLAN (Virtual Local Area Network) – виртуальная локальная компьютерная сеть.

VPN (Virtual Private Network) – обобщённое название технологий, позволяющих обеспечить одно или несколько сетевых соединений поверх чьей-либо другой сети.

WAN (Wide Area Network) – глобальная компьютерная сеть, которая объединяет устройства и локальные сети (LAN) на больших географических расстояниях.

Wi-Fi (Wireless Fidelity) – технология беспроводной локальной сети.

Введение

Настоящий документ представляет собой методические указания для выполнения лабораторных работ в учебной лаборатории Л400 (далее – Лаборатория) Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования «Московский технический университет связи и информатики» (далее – МТУСИ).

В качестве моделирующего устройства Лаборатории является Система контроля, мониторинга и управления трафиком (далее – Система КМУТ). Система КМУТ представляет собой аппаратно-программный комплекс, осуществляющий комплексное решение задач по оценке, анализу и управлению услугами связи по передаче данных в IP-сетях любой топологии и сложности, а также применяемый на реальных объектах.

Система КМУТ позволяет отслеживать оперативное состояние значений характеристик услуг связи по передаче данных на объектах мониторинга в режиме реального времени, а также осуществлять обработку и хранение полученных статистических данных. Измерения производятся с использованием метода «подмешивания» тестового трафика в активные соединения без ухудшения состояния параметров качества другого трафика.

1. Область применения

Оборудование Лаборатории позволяет моделировать любые неисправности на сети связи, при этом текущее состояние объектов мониторинга отображается в графическом интерфейсе при помощи цветового окраса, что позволяет сразу обнаружить неисправность. Система КМУТ позволяет создавать оповещения о нарушениях и неисправностях при имитации данных событий на оборудовании.

Основные задачи Лаборатории:

– обучение и практическое освоение навыков мониторинга и управления информационно-телекоммуникационными сетями различного масштаба и сложности;

– формирование навыков и умений пользования Системой КМУТ.

Все выполняемые лабораторные работы можно разделить на три модуля:

- Система мониторинга;
- Технологии коммутации, маршрутизации и диагностики в сетях связи. Сетевые технологии;
- Операционная система Linux.

В описании лабораторной работы приведены:

– ключевые слова;

– цель работы;

– краткие теоретические сведения;

– задание для выполнения работы;

– алгоритм выполнения лабораторной работы;

– содержание отчета.

При изучении курса соответствующей дисциплины студенты должны самостоятельно проработать необходимый материал по приведенному в лабораторной работе перечню ключевых слов и контрольных вопросов.

Выполнение лабораторных работ требует понимания структуры сетевой связности оборудования Лаборатории, назначения данного оборудования, а также способов доступа и авторизации на этом оборудовании. Данная информация содержится в пунктах 9–15.

Порядок выполнения лабораторной работы:

1. Перед выполнением каждой лабораторной работы

студент должен изучить ее описание, изложенное в настоящих методических указаниях, и проработать теоретический материал по теме работы. Перед началом лабораторной работы преподаватель проводит краткий опрос студентов с целью выяснения их подготовленности к проведению работы. Неподготовленные студенты к проведению работ не допускаются.

2. Перед началом работы в лаборатории студенты проходят инструктаж по технике безопасности с соответствующим оформлением в журнале.

3. Запись всех выполненных команд с результатом их выполнения заносятся каждым студентом в бланк отчета. При оформлении отчета по лабораторным работам для каждой работы следует привести название, цель работы, схему экспериментальной установки, результаты экспериментов и их обработки, придерживаясь формы таблиц, представленных в лабораторном практикуме, и выводы по работе.

4. Защита лабораторных работ проводится во время, назначаемое преподавателем. Основные контрольные вопросы к защите работы даны в конце описания каждой лабораторной работы.

2. Описание Системы КМУТ

Система КМУТ является технической системой с измерительными функциями, которые предназначены для измерений характеристик трафика в точках подключения к сети связи: количества переданной (принятой) информации (данных), продолжительности (длительности) сеансов передачи данных, пропускной способности канала передачи данных (PD), скорости передаваемой информации, средней задержки передачи пакетов данных, вариации задержки передачи пакетов данных (PDV), коэффициента потерь пакетов данных (PL). Системы КМУТ обеспечивают регистрацию времени проведения измерений с привязкой системной шкалы времени Системы КМУТ к национальной шкале времени Российской Федерации UTC (SU).

Функционально конструкция Системы КМУТ состоит из средств:

- проведения измерений характеристик трафика;
- сбора результатов измерений;
- передачи результатов измерений;
- обработки и представления результатов измерений.

В состав Системы КМУТ могут входить сервер (сервера) центрального узла и Зонды периферийного узла Системы КМУТ (далее – Зонды КМУТ) различных модификаций. Модификации Зондов КМУТ имеют конструктивные отличия в разъемах подсоединения и наличии информационных индикаторов, которые зависят от исполняемых функций.

Система КМУТ обеспечивает:

- измерение параметров сетей связи;
- сбор, обработку и хранение информации о характеристиках трафика;
- сбор, обработку и хранение информации о статистических оценках качества услуг связи, их целостности и устойчивости функционирования;
- управление элементами Системы КМУТ;
- сбор, обработку и хранение информации о наличии электропитания и о событиях выключения, временного интервала отсутствия и включения электропитания элементов Системы КМУТ.

Измерения задержек и вариаций задержек передачи пакетов данных осуществляются методом прямых измерений расхождения внутренней шкал времени измерительных каналов Системы КМУТ, синхронизованных с национальной шкалой времени Российской Федерации UTC (SU), со шкалами времени, синхронизованными с сетевыми событиями (отправка или приём пакетов данных).

3. Описание Зондов КМУТ

Зонды КМУТ предназначены для выполнения функций автоматизированного контроля параметров каналов связи.

Принцип действия Зондов КМУТ основан на формировании тестового трафика в активных соединениях сети связи, измерении и регистрации характеристик этого трафика при прохождении по сети связи, анализа измеренных характеристик с целью получения статистических оценок целостности и устойчивости функционирования каналов сети связи. Весь проходящий через Зонды КМУТ трафик по желанию администратора может быть описан при помощи протокола NetFlow.

Измерения задержек и вариаций задержек передачи пакетов данных осуществляются методом прямых измерений расхождения внутренней шкалы времени Зондов КМУТ, синхронизованной с национальной шкалой времени Российской Федерации UTC (SU), со шкалами времени, синхронизованными с сетевыми событиями (отправка или приём пакетов данных).

Измерению подлежат характеристики трафика между Зондами КМУТ или Зондами КМУТ и серверами Системы контроля, мониторинга и управления трафиком, в том числе центральным сервером измерений.

Зонды КМУТ обладают следующими функциональными возможностями:

– измерение характеристик трафика в сети связи с пропускной способностью до 1 Гбит/с;

– резервирование каналов связи (услуг связи) с использованием протоколов динамической маршрутизации;

– одновременный контроль характеристик трафика двух независимых каналов связи;

– определение наличия напряжения в сети электропитания с привязкой к системной шкале времени (режим синхронизации от сервера Системы КМУТ) относительно национальной шкалы времени Российской Федерации UTC (SU), хранение в памяти и выдача информации на сервер центрального узла Системы КМУТ о событиях выключения, временного интервала отсутствия и включения электропитания.

Конструктивно Зонды КМУТ выполнены в виде моноблоков, в которых размещены специализированные электронные платы. На

боковых панелях корпусов расположены соответствующие разъемы для подключения к сети связи, подачи электропитания. Защитные корпуса моноблоков изготавливаются из штампованного металла или пластика и имеют съемную боковую или нижнюю панель, крепление которой осуществляется с помощью винтов.

Зонды КМУТ обеспечивают непрерывный и круглосуточный режим функционирования.

3.1. Органы управления и порты Зонда КМУТ М5

Зонд КМУТ М5 включает в себя порт управления «RS-232», 4 порта «Ethernet» (0-3), 2 порта USB 3.0.

Включение и выключение Зонда КМУТ М5 осуществляется нажатием на переключатель питания.

Для выключения питания нажать переключатель питания.

Для редактирования конфигурационных файлов в системе используется встроенный текстовый редактор «VI».

Обозначение на Зонде КМУТ М5 индикаторов, портов, интерфейсов и органов управления (рис. 1):

Обозначен	Описание
УПР	Порт для подключения консольным кабелем (RS-232, DE-9)
0,1,2,3	Порты для подключения кабелей Ethernet (RJ-45)
	Порты для подключения USB устройств
	Разъем электропитания с кнопкой включения электропитания
СТАТУС 1, 2, 3	Индикаторы состояния Зонда КМУТ М5

Рис. 1. Схема расположения индикаторов и портов на передней панели Зонда КМУТ М5

Порт RS-232 (DE-9) служит для доступа к Зонду КМУТ М5 непосредственно через консоль управления и используется для первичной настройки Зонда КМУТ М5.

Параметры подключения к данному порту следующие:

Параметр	Значение
Скорость передачи данных	115200 бод
Формат передачи данных	8 бит
Стоповый бит	1
Контроль четности	отсутствует

Интерфейсы «0», «1», «2» и «3» (10/100/1000BaseT) служат для подключения Зонда КМУТ М5 к операторам связи и локальной сети объекта.

Кнопка «I»-«O» обеспечивает включение/выключение электропитания Зонда КМУТ М5 (рис. 2).

Разъем электропитания «220В 50Гц 1А» предназначен для подключения Зонда КМУТ М5 к источнику электропитания.

Рис. 2. Схема расположения элементов на задней панели Зонда КМУТ М5

3.2. Органы управления и порты Зонда КМУТ М7

Зонд КМУТ М7 имеет один интерфейс «0» для подключения кабеля Ethernet.

Для включения электропитания необходимо подключить внешний блок питания.

Для редактирования конфигурационных файлов в системе используется встроенный текстовый редактор «VI».

Обозначение на Зонде КМУТ М7 индикаторов, портов, интерфейсов и органов управления (рис. 3):

Обозначение	Описание
	Кнопка для перезагрузки Зонда КМУТ М7
	Интерфейс «0» для подключения кабеля Ethernet (RJ-45)
	Разъем для подключения внешнего блока питания
	Индикаторы состояния Зонда КМУТ М7
F	Кнопка возврата к заводским настройкам

Рис. 3. Схема расположения индикаторов и портов на передней панели Зонда КМУТ М7

Интерфейс «0» (10/100/1000BaseT) служит для подключения Зонда КМУТ М7 к оборудованию объекта.

Для возврата устройства к заводским настройкам требуется нажать и удерживать кнопку «F» нижней панели (рис. 4) более трех секунд, пока индикатор на передней панели не начнет мигать красным светом. В течение 5 секунд произойдёт автоматическая

перезагрузка устройства, индикатор загорится постоянным красным светом. По завершению загрузки индикатор загорится желтым светом.

Разъем электропитания предназначен для подключения внешнего блока питания.

Рис. 4. Схема расположения элементов на нижней панели Зонда КМУТ М7

3.3. Органы управления и порты Зонда КМУТ М11

Зонд КМУТ М11 включает в себя порт управления «RS-232», 4 порта «Ethernet» (0-3), 2 порта USB.

Для включения питания необходимо подключить внешний блок питания в электросеть.

Для редактирования конфигурационных файлов в системе используется встроенный текстовый редактор «VI».

Обозначение на Зонде КМУТ М11 индикаторов, портов, интерфейсов и органов управления (рис. 5):

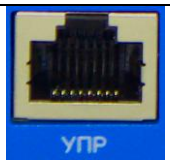
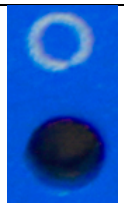
Обозначение	Описание
	Порт для подключения консольным кабелем (RS-232, RJ-45)
0,1,2,3	Порты для подключения кабелей Ethernet (RJ-45)
	Порты для подключения USB устройств
	Кнопка включения/выключения Зонда КМУТ
	Кнопка перезагрузки
	Индикатор электропитания устройства

Рис. 5. Схема расположения индикаторов и портов на передней панели Зонда КМУТ М11

Порт RS-232 (RJ-45) служит для доступа к Зонду КМУТ М11 непосредственно через консоль управления. Данный порт используется для первичного конфигурирования Зонда КМУТ М11. Параметры подключения к данному порту следующие:

Параметр	Значение
Скорость передачи данных	115200 бод
Формат передачи данных	8 бит
Стоповый бит	1
Контроль четности	отсутствует

Необходимо убедиться, что терминал или ПК, который используется для подключения, настроен в соответствии с данными настройками.

Интерфейсы «0», «1», «2» и «3» (10/100/1000BaseT) служат для подключения Зонда КМУТ М11 к операторам связи и локальной сети объекта.

Разъем «СЕТЬ» предназначен для подключения КМУТ М11 к внешнему блоку электропитания напряжением 12 Вольт (рис. 6).

Рис. 6. Схема расположения элементов на задней панели Зонда КМУТ М11

4. Описание интерфейса Системы КМУТ ML

4.1. Вход в интерфейс Системы мониторинга КМУТ ML

Для подключения к интерфейсу Системы мониторинга КМУТ ML необходимо на АРМ открыть браузер и перейти по адресу:

https://10.19.47.252

Если появляется ошибка типа «Подключение не защищено» (ERR_CERT_AUTHORITY_INVALID), игнорировать ошибку и подтвердить переход на сайт.

Доступность адреса гарантируется на АРМ только в пределах Лаборатории.

В окне «Авторизация» необходимо ввести учетные данные для входа в интерфейс Системы мониторинга КМУТ ML (рис. 7). Данные для входа представлены в разделе 11.

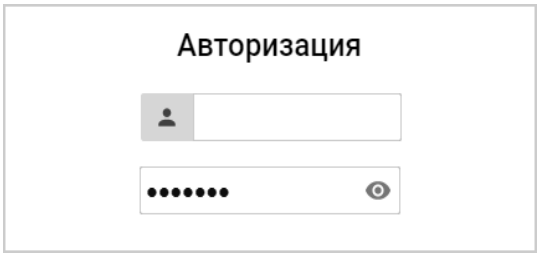


Рис. 7. Поле авторизации в Системе мониторинга КМУТ ML

После успешной авторизации пользователь выбирает один из доступных ему интерфейсов. В зависимости от типа учетной записи Система мониторинга КМУТ ML предоставляет три независимых интерфейса через единую точку входа (рис. 8):

- Интерфейс-конструктор – предназначен для заведения классов, связей и автоматов предметной области. Также в этом интерфейсе производится настройка макетов отображения в интерфейсе-витрина;
- Интерфейс-менеджер – предоставляются инструменты создания и редактирования объектов предметной области и программных сущностей. Формы объектов предметной области – автогенерируемые;
- Интерфейс-витрина – визуализирует объекты предметной области и их состояния. Также этот интерфейс предоставляет средства доступа к служебным инструментам, таким как создание отчетов, массовых операций и т.д.

Рис. 8. Единая точка входа

4.2. Описание интерфейса «Витрина»

Интерфейс-витрина является полностью настраиваемым.

Общая концепция:

- Меню навигации по объектам указанных классов. До настройки слева – аккордеон списков объектов классов-вершин (аналогично главному меню в интерфейсе-менеджере);
- Стартовая страница, составленная из виджетов (если не настроено – аналогична стартовой странице интерфейсе-менеджера;
- До настройки – автогенерируемая карточка объекта по аналогии с автогенерируемой формой редактирования объектов в интерфейсе-менеджере.

4.2.1. Зоны интерфейса «Витрина»

Зоны интерфейса «Витрина» показаны на рис. 9.

Рис. 9. Зоны интерфейса «Витрина»

Общий принцип составляющих Витрины:

1. Зона авторизации в системе с выпадающими списками;
2. Зона меню пользователя, где отображены переход между интерфейсами системы в зависимости от прав юзера. Возможность смены пароля или включение «темной темы»;
3. Зона поиска и принудительной повторной загрузки сущностей системы;

4. Зона логотипа проекта (рис. 10);

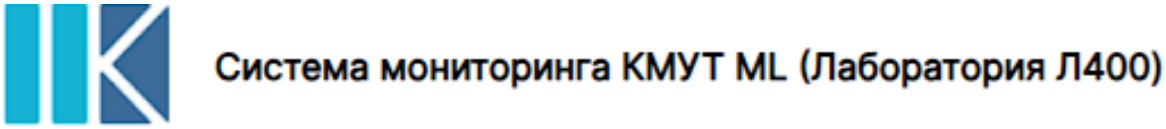


Рис. 10. Зона логотипа проекта

5. Зона бокового меню. Его наполнение – задача администраторов системы, которые создавая «Макеты объектов» или «Общие макеты», могут привязать URL макета к странице, для которой можно сделать кнопку в этом меню, а также выбрать графическое представление для соответствующего пункта меню;

6. Зона виджета «Дерево», отображающего все заведенные объекты в системе (рис. 11). Может быть одинарным («Виджет основного дерева») или двойным связанным (сверху «Основное дерево» и снизу «Дочернее дерево»). При использовании связанных деревьев, содержимое «Дочернего дерева» зависит от того, какой пункт «Основного дерева» был выбран;

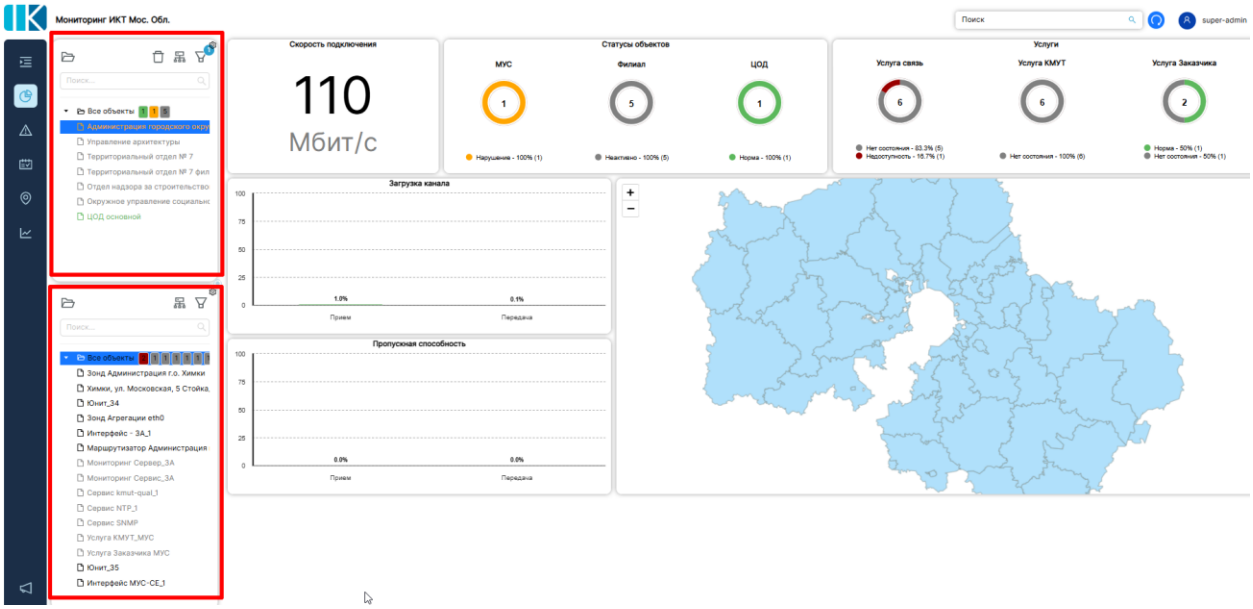


Рис. 11. Зона виджета «Дерево»

7. Зона настраиваемых виджетов.

4.2.2. Настройка виджета «Дерево» и «Связанные деревья»

Данный виджет предназначен для компактного вывода всех объектов системы в виде иерархического списка, включающего цветовую индикацию объектов – с одной стороны, и для навигации по индивидуальным страницам объектов – с другой стороны.

В режиме связанных деревьев пользователю предоставляется возможность вывести список дочерних объектов (в нижнем дереве) при выборе одного из родительских объектов (в верхнем дереве).

Виджет дерева состоит из четырех зон (рис. 12):

1. Зона управления;
2. Зона расширенного управления и настроек;
3. Зона поиска;
4. Зона со списком объектов (в том числе и сгруппированных).

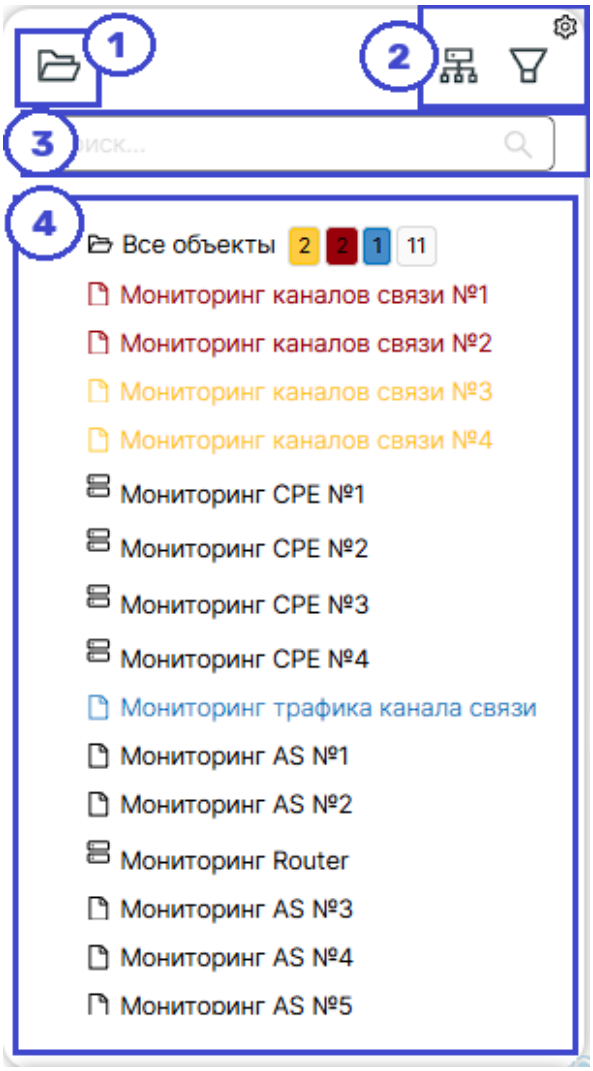


Рис. 12. Виджет дерева

4.2.3. Зона управления

Данная зона содержит кнопку быстрого развертывания и свертывания всех ветвей (при использовании группировки).

4.2.4. Зона расширенного управления и настройки

Состоит из:

- Компонента управления группировкой, позволяющего выбрать в качестве критерия группировки класс объекта (рис. 13);

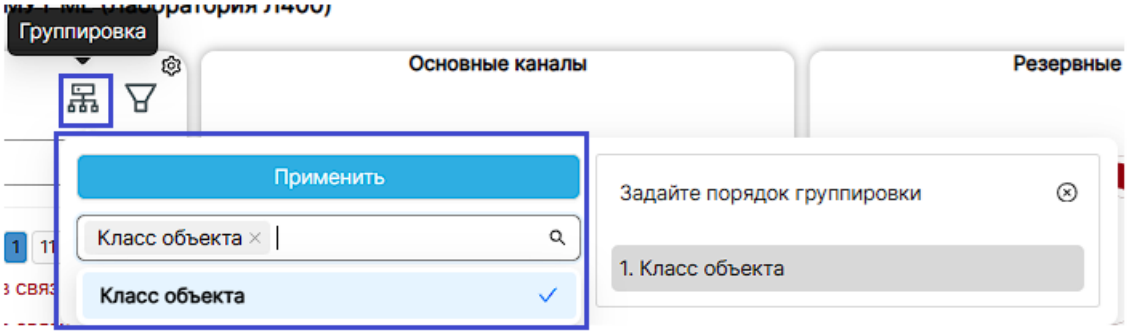


Рис. 13. Компонент управления группировкой

- Компонента фильтрации, позволяющего выбрать критерии фильтрации, включая фильтрацию по классам и связанным объектам (рис. 14);

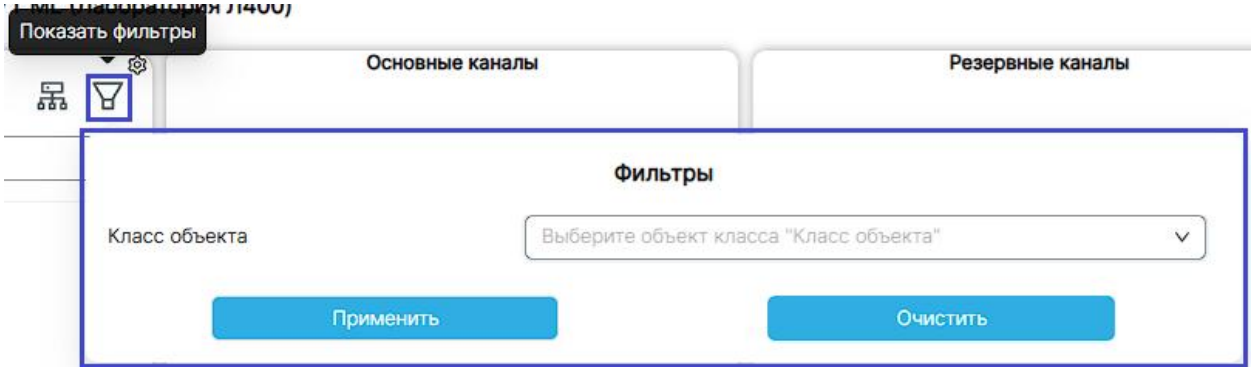


Рис. 14. Компонент фильтрации

- Компонента настройки виджета (рис. 15), позволяющего указать те классы объектов, которые будут отображены в дереве (см. элемент №1), а также связующие классы для них (см. элемент №2). Также присутствует элемент управления отслеживанием, который применяется в Дочернем дереве при использовании связанных деревьев и должен содержать основной класс из Родительского дерева для взаимной синхронизации (элемент 3).

Рис. 15. Компонент настройки виджета

4.2.5. Зона поиска

Зона поиска предоставляет возможность быстрой фильтрации содержимого дерева по поисковой подстроке, которая применяется к наименованиям отображаемых в дереве объектов.

4.2.6. Зона со списком объектов (в том числе и сгруппированных)

Зона содержит список объектов, сгруппированных согласно настройкам виджета. Зона состоит из:

- Корневой автоматически выводимой группы (элемент 1);
- Статистики корневой группы в виде цветных элементов, отображающих количество объектов группы, находящихся том или ином состоянии (элемент 2);
- Одной или нескольких дочерних групп объектов (элемент 3);
- Списка объектов, вложенных в группу, включая названия объектов, а также цветовую индикацию их текущего состояния (элемент 4).

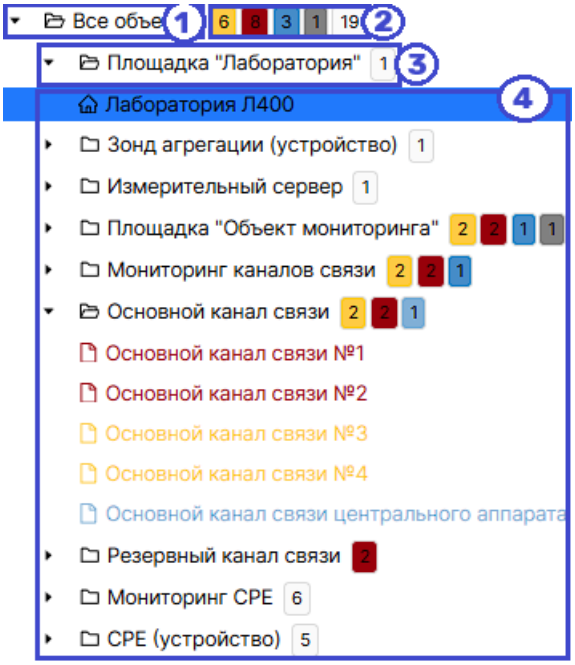
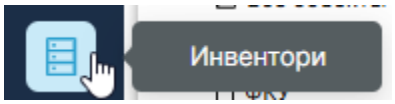


Рис. 16. Зона со списком объектов

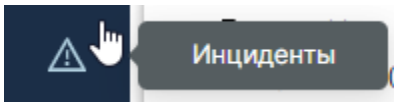
4.2.7. Структура бокового меню

Боковое меню состоит из набора кнопок.

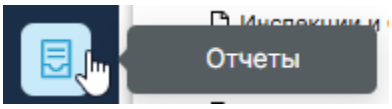
Позволяет свернуть боковое дерево



Кнопка «Активы»



Пункт меню «Инциденты»



Пункт меню «Отчеты»

4.2.8. Модуль «Активы»

Позволяет увидеть табличное расположение всех объектов-устройств в системе со всеми их атрибутами. Сортировка в каждом столбце поможет отсортировать нужные объекты. Также справа вверху видно количество объектов на каждой странице, возможность переключения между ними. Есть кнопка выгрузки в формате Excel и настройка выводимой таблицы.

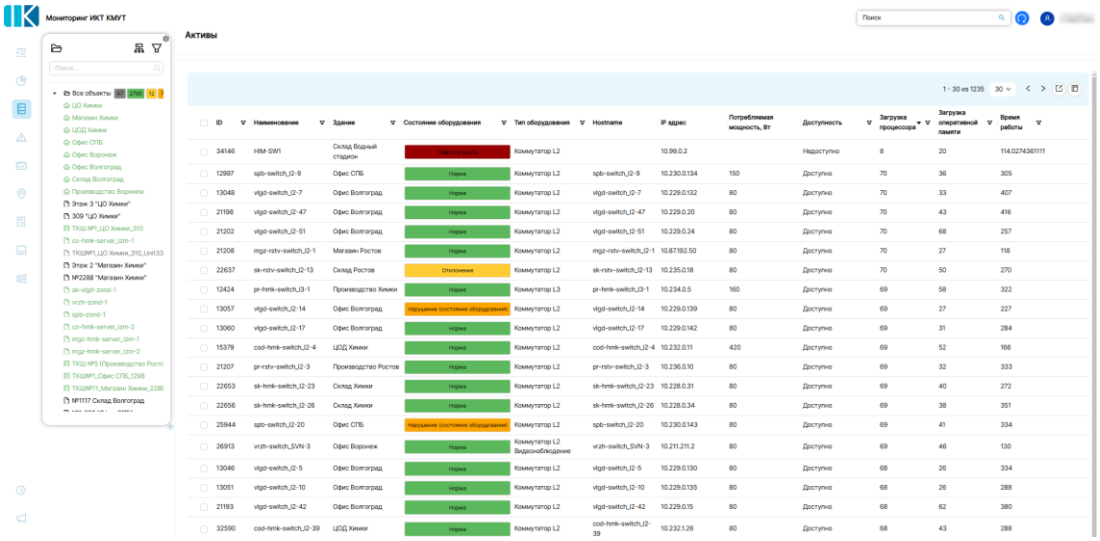


Рис. 17. Окно таблицы модуля «Активы»

Редактирование выводимой таблицы активы (рис. 18). В ней перечислены все атрибуты объектов-устройств. Можем скрыть ненужные столбцы, поменять их местами или задать нужную ширину столбца. Это удобно делать для того, чтобы таблица помещалась на одной странице без использования горизонтального скролла.

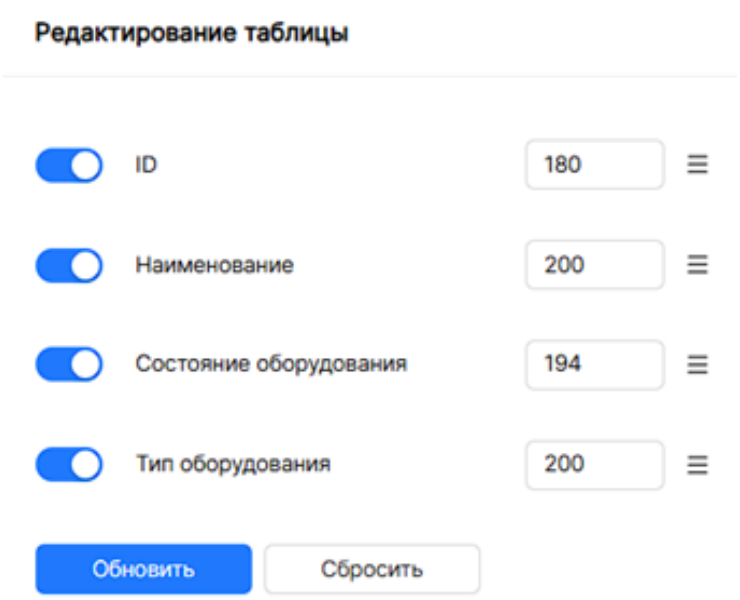


Рис. 18. Редактирование выводимой таблицы

4.2.9. Модуль «Инциденты»

Инциденты отображают данные по неисправности оборудования или каналов связи (рис. 19). Правила оценки состояния объектов системы и выявления неисправностей настраивается в модуле «Автоматы» интерфейса «Конструктор». Критерии оценки состояний, в том числе и неисправностей оборудования, настраивается индивидуально для каждого проекта.

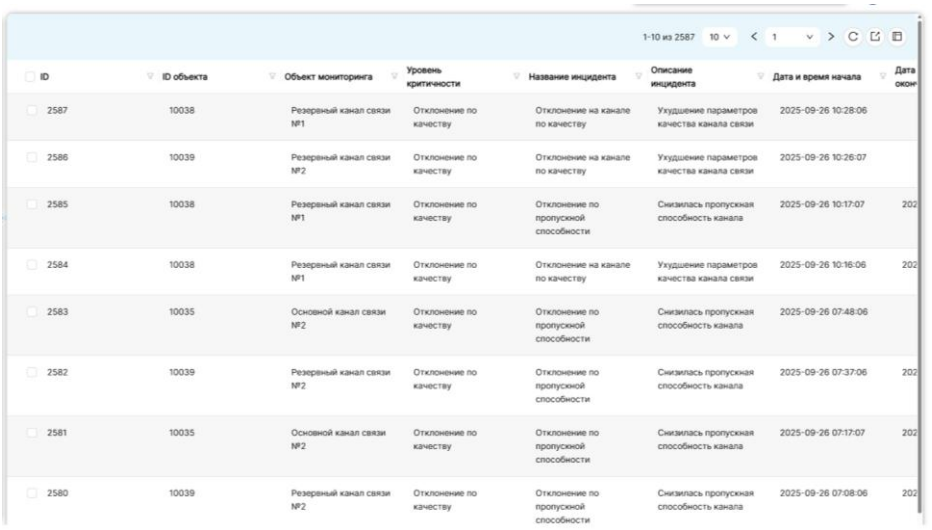


Рис. 19. Модуль «Инциденты»

Также инциденты отображаются на большом графике в зоне настраиваемых виджетов (на главной странице), что позволяет визуально представлять количество инцидентов за определенное время (рис. 20).

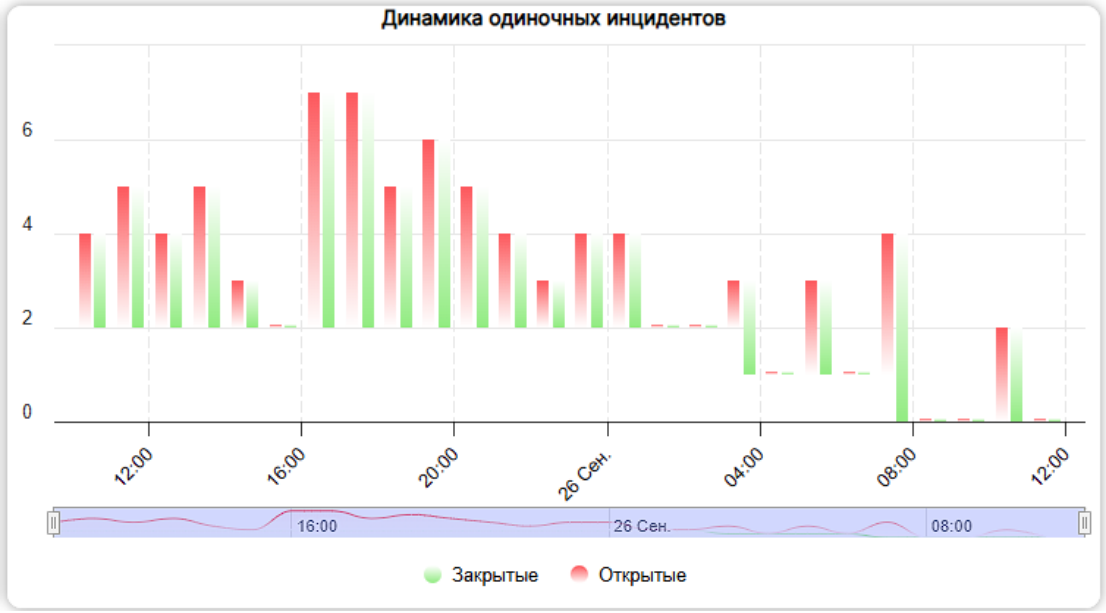


Рис. 20. Визуальное представление количества инцидентов

4.2.10. Модуль «Отчеты»

Модуль отчеты предназначен для построения отчетов по форме заказчика или в свободной форме. Отчеты в системе могут быть сгенерированы в различных «офисных» форматах – XLSX, PDF, DOCX, ODT, CSV и так далее.

Модуль отчетов представлен таблицей заданий на построение отчетов, а также формой создания задания на построение.

В таблице заданий в виде списка выводятся созданные пользователями системы задания на построение отчетов, включая тип отчета, генерируемые форматы, интервал дат построения, текущий статус выполнения задания и инструменты скачивания готового отчета (рис. 21).

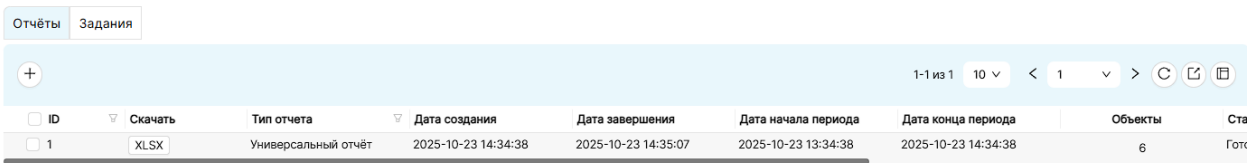


Рис. 21. Таблица заданий

Форма создания отчета состоит из трех «Шагов». На первом выбираются основные параметры отчета, на втором производится конструирование полей для универсальных отчетов, на третьем – выбираются объекты, которые будут представлены в отчете.

Первый шаг формы отчета состоит из следующих полей (рис. 22):

1. Индикатор текущего шага;

2. Выбор периодичности построения;
3. Выбор регулярности;
4. Выбор даты начала (только для периодических отчетов);
5. Тип отчета (из набора активированных в системе);
6. Доступные форматы отчета (зависят от типа отчета);
7. Период построения отчета (из набора предустановленных);
8. Дата начала периода построения (если предустановленный период не выбран);
9. Дата конца периода построения (если предустановленный период не выбран).

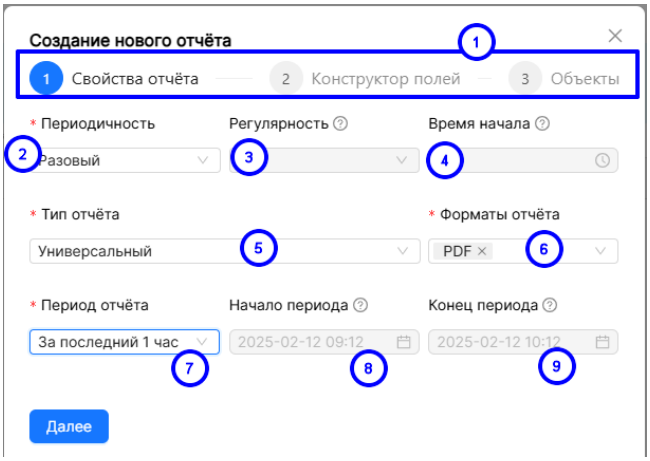


Рис. 22. Форма создания отчета

Второй шаг появляется только для одного типа отчета – «Универсального» и предназначен для конструирования его полей (рис. 23). Описание этого экрана – предмет отдельного документа.

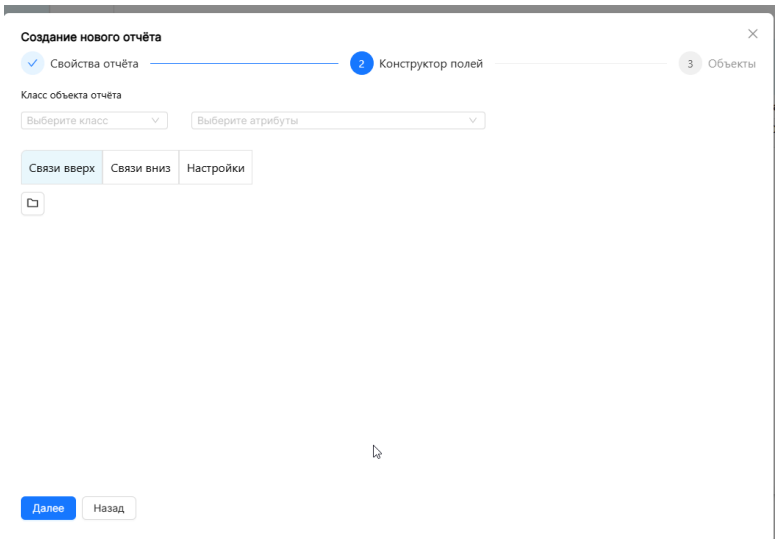


Рис. 23. Окно конструирования универсального отчета

Третий шаг предназначен для выбора объектов, которые будут отображены в отчете (рис. 24).

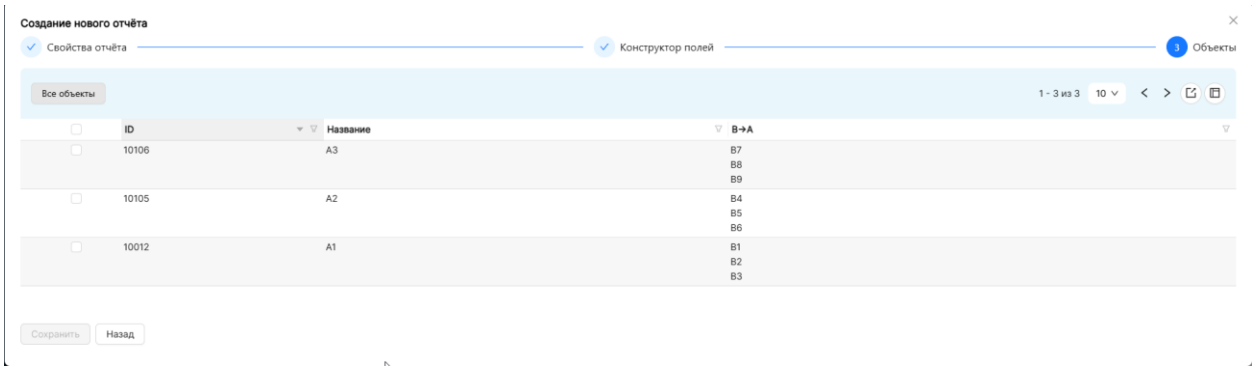


Рис. 24. Окно выбора объектов

Выполнение лабораторной работы по модулю «Отчеты» представлено в разделе 7.1.5. «Изучение отчетных форм по результатам мониторинга».

5. Требования для подключения Системы КМУТ к сети электропитания и связи

5.1. Обеспечить Центральный сервер измерений (далее – ЦСИ) доступом к ресурсам сети Интернет ежедневно 24 часа в сутки, в том числе в праздничные и выходные дни, с неограниченным объемом безлимитного входящего и исходящего трафика в точке подключения, за исключением проведения профилактических и ремонтных работ.

5.2. Обеспечить информирование о проведении профилактических и ремонтных работ на сети Университета, влияющих на доступность ЦСИ, не менее чем за 24 часа до их проведения.

5.3. Обеспечить скорость доступа входящего трафика до ЦСИ не менее 100 Мбит/сек.

5.4. Обеспечить скорость доступа исходящего трафика до ЦСИ не менее 100 Мбит/сек.

5.5. Обеспечить выделение и маршрутизацию одного постоянного (статического, «белого») IP адреса для ЦСИ. Допускается устанавливать защищенное соединение с помощью протоколов туннелирования через сеть Интернет между МТУСИ и ООО «Контроль ИТ».

5.6. Обеспечить подключение ЦСИ к сети Интернет с использованием интерфейса Ethernet.

5.7. Обеспечить прохождение TCP трафика на порт 22, 80, 443, 5000, 5001, 8022, 9022 и UDP трафика на порт 123, 161, 2000, 2161, 5001 между ЦСИ и стойкой оборудования в Учебной лаборатории.

5.8. Обеспечить прохождение TCP трафика на порт 22, 8022 ЦСИ в сторону сети Интернет.

5.9. Обеспечить стабильное напряжение электропитания ЦСИ и Учебной лаборатории напряжением 220 ± 22 В и частотой 50 Гц.

5.10. Сеть электропитания оборудования учебной лаборатории должна иметь отдельный проводник защитного заземления.

5.11. Телекоммуникационная стойка, установленная в помещении учебной лаборатории, должна подключаться к контуру заземления.

5.12. Обеспечить климатический режим в помещении, где установлено оборудование ЦСИ.

5.13. При установке в Учебной лаборатории точки доступа Wi-Fi должен обеспечиваться безопасный беспроводной доступ за счет использования современных технологий в области аутентификации и авторизации.

6. Техника безопасности при выполнении лабораторных работ

6.1. Общие требования:

6.1.1. Все студенты, связанные с работой в лаборатории, обязаны пройти инструктаж по безопасному выполнению работ, о чем расписываются в журнале инструктажа по технике безопасности.

6.1.2. К работам по эксплуатации оборудования допускаются лица, прошедшие медицинский осмотр и инструктаж по охране труда. Не электротехническому персоналу, эксплуатирующему оборудование до 1000 В, прошедшему инструктаж и проверку знаний по электробезопасности, присваивается I квалификационная группа допуска с оформлением в журнале установленной формы с обязательной росписью проверяющего и проверяемого.

6.1.3. Все лица, связанные с работой в лаборатории, должны соблюдать правила внутреннего трудового распорядка, установленные режимы труда и отдыха.

6.1.4. При работе в лаборатории и эксплуатации оборудования до 1000 Вольт возможно воздействие на работающих следующих опасных производственных факторов:

- поражение электрическим током при прикосновении к токоведущим частям;
- неисправности изоляции или заземления;
- воздействие влажности.

6.1.5. Действие факторов: вследствие неисправности кабеля, электрической вилки (розетки) или замыкания цепи работающий попадает под напряжение.

6.1.6. При эксплуатации оборудования до 1000 Вольт в лаборатории должны использоваться следующие средства индивидуальной защиты: указатель напряжения, инструмент с изолированными ручками.

6.1.7. Лица, эксплуатирующие оборудование до 1000 Вольт, обязаны строго соблюдать правила пожарной безопасности, знать места расположения первичных средств пожаротушения, а также отключающих устройств (рубильников) для снятия напряжения.

6.1.8. О каждом несчастном случае пострадавший или

очевидец несчастного случая обязан немедленно сообщить преподавателю или администрации МТУСИ. При неисправности оборудования прекратить работу, снять с него напряжение и сообщить преподавателю или администрации.

6.1.9. В процессе эксплуатации оборудования персонал должен соблюдать правила использования средств индивидуальной защиты, соблюдать правила личной гигиены, содержать в чистоте рабочее место.

6.1.10. Оборудование, установки, приборы, инструменты должны использоваться только по прямому назначению.

6.1.11. В лаборатории должны проводиться только те виды работ, которые соответствуют плану работы лаборатории. Запрещается проводить другие виды работ.

6.1.12. Лица, допустившие невыполнение или нарушение инструкции по охране труда, привлекаются к дисциплинарной ответственности в соответствии с правилами внутреннего трудового распорядка и, при необходимости, подвергаются внеочередной проверке знаний норм и правил охраны труда.

6.2. Требования безопасности перед началом работы:

6.2.1. Ознакомиться с инструкцией по технике безопасности на рабочем месте и получить дополнительный инструктаж, о чем делается соответствующая запись в журнале инструктажа по технике безопасности.

6.2.2. Привести в порядок свою рабочую одежду.

6.2.3. Проверить освещение рабочего места – оно должно быть достаточным, свет не должен слепить глаза.

6.2.4. Проверить отсутствие внешних повреждений оборудования, наличие и исправность оборудования и т.п.

6.2.5. Убедиться в целостности крышек электророзеток и выключателей, электровилки и подводящего электрокабеля.

6.2.6. Убедиться в наличии и целостности заземляющего проводника корпуса оборудования.

6.2.7. Проверить наличие и исправность средств индивидуальной защиты, отсутствие их внешних повреждений.

6.2.8. Внимательно осмотреть рабочее место и привести его в порядок, убрать посторонние предметы.

6.2.9. Запрещается:

- производить работу установки без разрешения

преподавателя;

- включать силовые и осветительные рубильники без разрешения преподавателя;
- входить в технические помещения без особого разрешения преподавателя;
- включать схему, работающую под напряжением, без предварительной проверки и без разрешения преподавателя;
- работать с незаземленным оборудованием. Отключать и обрывать провода защитного заземления;
- снимать и перевешивать предупреждающие и запрещающие плакаты.

6.3. Требования безопасности во время работы:

6.3.1. Перед включением оборудования в электрическую сеть, при необходимости, встать на диэлектрический коврик (если покрытие пола выполнено из токопроводящего материала).

6.3.2. Не включать оборудование в электрическую сеть мокрыми и влажными руками.

6.3.3. Соблюдать правила эксплуатации оборудования, не подвергать его механическим ударам, не допускать падений.

6.3.4. Не касаться проводов и других токоведущих частей, находящихся под напряжением, без средств индивидуальной защиты.

6.3.5. Запрещается передвижение по лаборатории без необходимости.

6.3.6. Запрещается находиться в лаборатории в верхней одежде, а также вешать ее на лабораторное оборудование.

6.3.7. Все работающие в лаборатории обязаны бережно относиться к оборудованию.

6.3.8. Следить за исправной работой электроустановки, целостностью изоляции и заземления.

6.3.9. Не разрешается работать на оборудовании в случае его неисправности, искрения, нарушения изоляции и заземления.

6.3.10. В случае пореза следует оказать первую помощь пострадавшему, воспользоваться аптечкой, остановить кровотечение, обратиться к преподавателю.

6.4. Требования безопасности в аварийных ситуациях:

6.4.1. При появлении неисправности в работе оборудования, искрении, нарушении изоляции проводов или обрыве

заземления, прекратить работу и отключить общий рубильник. Сообщить преподавателю или администрации о случившемся. Работу продолжать только после устранения неисправности специалистами.

6.4.2. При обнаружении оборванного электрического провода, свисающего или касающегося пола (земли), не приближаться к нему, немедленно сообщить преподавателю или администрации, самому оставаться на месте и предупреждать других людей об опасности.

6.4.3. В случае загорания оборудования, немедленно отключить ее от электрической сети, а пламя тушить только песком, углекислотным или порошковым огнетушителем.

6.4.4. При поражении электрическим током немедленно отключить напряжение; при отсутствии дыхания и пульса у пострадавшего сделать ему искусственное дыхание или провести непрямой (закрытый) массаж сердца до восстановления дыхания и пульса, сообщить о несчастном случае преподавателю или администрации, при необходимости отправить пострадавшего в ближайшее лечебное учреждение.

6.4.5. После происшедшего несчастного случая выяснить причины аварии и пути ее устранения.

6.5. Требования безопасности после окончания работы

6.5.1. Отключить АРМ от электрической сети. При отключении от электророзетки не дергать за электрический шнур (кабель).

6.5.2. Привести в порядок рабочее место.

7. Лабораторные работы

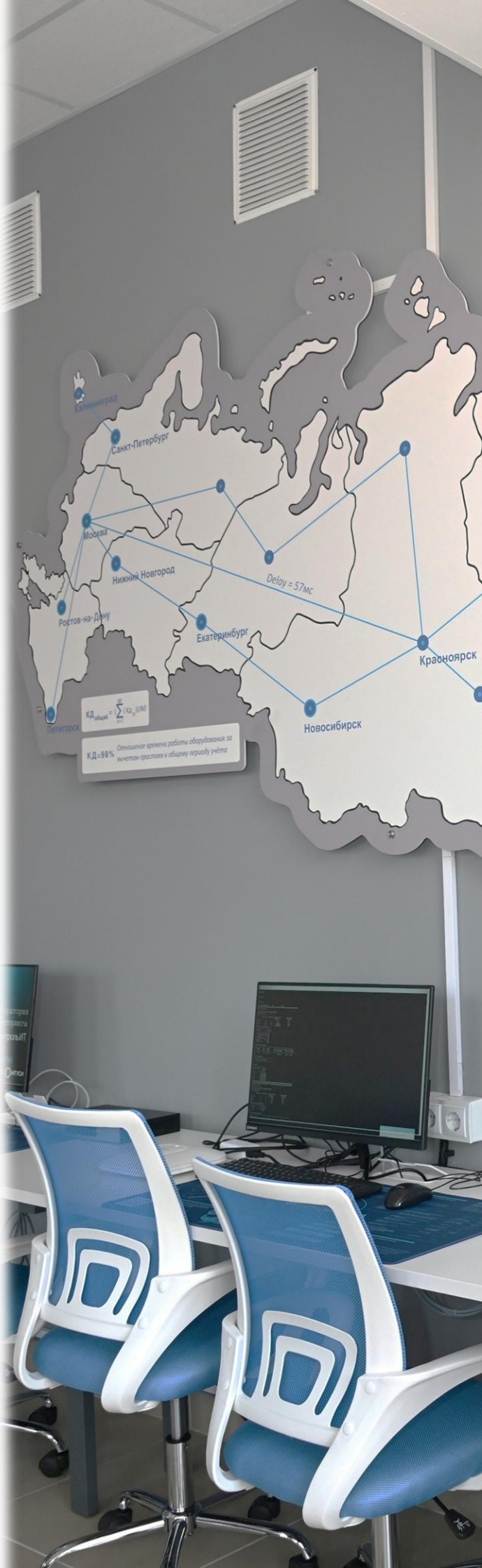
7.1. Перечень лабораторных работ по модулю «Система мониторинга»

В результате выполнения лабораторных работ вы познакомитесь с Системой мониторинга КМУТ ML:
научитесь работать с её интерфейсом,
анализировать параметры сети,
исследовать трафик и схемы включения Зондов КМУТ.

Освоите:
создание отчетов и управление Зондами КМУТ.

Получите:
навыки подключения и настройки Зондов КМУТ.

Поймете:
механизмы выявления и устранения инцидентов в сети связи.



7.1.1 Ознакомление с Системой мониторинга КМУТ ML

Ключевые слова

Система мониторинга КМУТ ML.

Цель работы

Ознакомиться с системой мониторинга КМУТ ML, ее основными элементами, функциями, вариантами применения и интерфейсом.

Краткие теоретические сведения

Система мониторинга используется для дистанционного сбора, хранения и анализа данных о состоянии подконтрольных элементов ИТ инфраструктуры.

В комплексную систему мониторинга входит: мониторинг сетей, мониторинг серверов и рабочих станций, мониторинг приложений и сервисов, мониторинг бизнес-процессов, предоставление данных в виде отчетов и графиков.

Существуют две основные модели мониторинга:

- Push-модель – сервер мониторинга ожидает подключений от агентов для получения метрик. Модель обычно используется для мониторинга больших систем, где количество устройств может быть слишком большим для ручного сбора данных;
- Pull-модель – сервер мониторинга сам подключается к агентам мониторинга и забирает данные.

Существует большое количество систем мониторинга, которые используются в зависимости от задач. Одни из самых популярных: Zabbix, Prometheus, PRTG, Nagios, Cacti. Для объединения разрозненных систем мониторинга используют зонтичную систему мониторинга, которая служит точкой входа и обработки данных, полученных от специализированных систем мониторинга.

Система мониторинг КМУТ ML – проприетарная система мониторинга с закрытым исходным кодом. Входит в реестр отечественного программного обеспечения. Основное назначение – измерение качества каналов связи, состояния оборудования, оповещение о возникающих инцидентах.

Сетевая схема Лаборатории (пункт 13 Методического указания) состоит из Системы мониторинга, в которую входят:

- Сервер КМУТ (kmut-ml-mtusi) – ядро системы, которое включает в себя следующие компоненты: хранилище, веб-интерфейс, измерительная подсистема, система создания и оповещения об инцидентах, модули инвентаризации и отчетности;
- Зонд агрегации (ZA-M11) – метрологически поверенное средство измерения. Используется для проведения измерений, получения результатов измерений от других Зондов мониторинга, передачи результатов измерений на сервер Системы КМУТ. Также необходим для снижения нагрузки на сервер Системы КМУТ.
- Зонд мониторинга (CPE1-M5, CPE2-M5, CH-M11, CE1-CE14, P1, P2) – метрологически поверенное средство измерения, до него проводятся измерения каналов связи от сервера Системы КМУТ или Зонда агрегации;
- КМУТ-агент – приложение, которое устанавливается на сервера, АРМ и т.п. для контроля локальных ресурсов и приложений. Не используется в оборудовании Лаборатории;
- Интеграционные модули с другими системами мониторинга. Не используются в оборудовании Лаборатории.

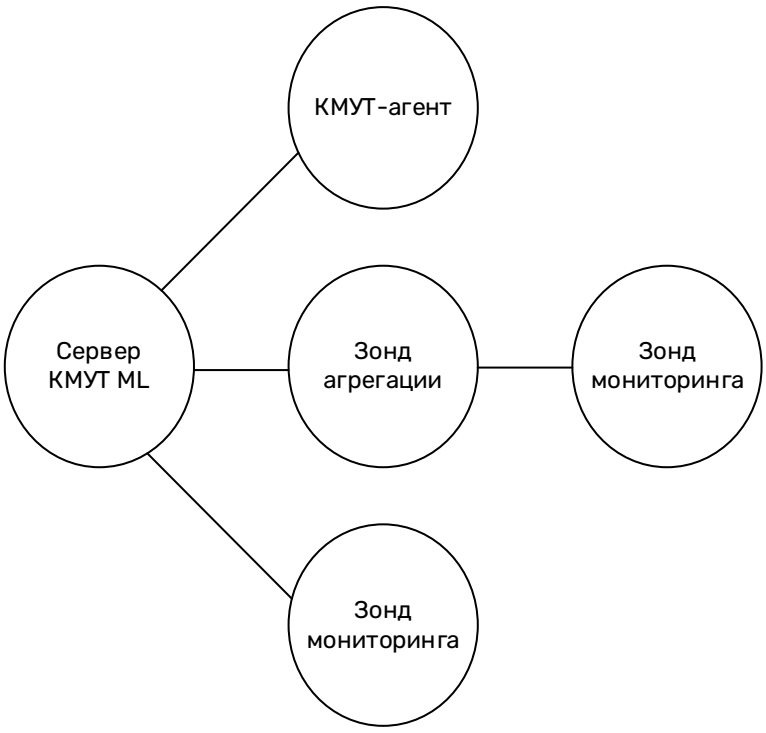


Рис. 24. Состав системы мониторинга

Используемые технические средства.

АРМ с установленным веб-браузером;

Сервер с Системой мониторинга КМУТ ML.

Задание

Ознакомиться с интерфейсом системы: информационная панель, инвентаризация, отчеты, инциденты.

Алгоритм выполнения лабораторной работы

1. На АРМ открыть браузер и перейти по адресу https://10.19.47.252 (согласно п.4.1.);
2. Ввести данные для авторизации (указаны в разделе 11) и нажать кнопку «Войти» (рис. 25):

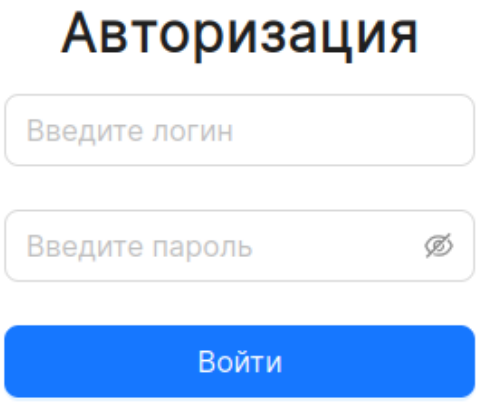


Рис. 25. Окно авторизации

3. В момент загрузки (рис. 26) при нажатии кнопки «Показать данные загрузки» внизу страницы отобразятся загружаемые модули и данные;

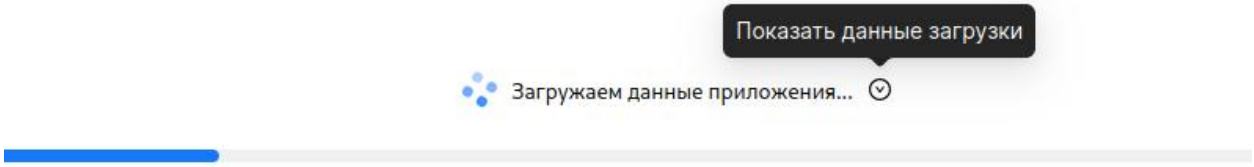


Рис. 26. Окно загрузки данных приложения

4. После загрузки откроется страница веб-интерфейса Системы мониторинга КМУТ ML, состоящая из нескольких элементов: верхняя панель, боковая панель, информационная панель:

a. Информационная панель. Основной источник получения оперативных данных о состоянии измеряемого оборудования. Гибко настраивается за счет большого набора элементов отображения информации.

b. Боковая панель. Здесь расположены кнопки перехода в основные модули системы:

- Информационная панель;
- Активы;
- Инциденты;

- Отчеты;

с. Верхняя панель.

На данной панели расположены:

- Логотип и название проекта
- Поисковая строка.
- Информация о пользователе и управление настройками пользователя, такие как: наименование учетной записи, настройка темы, смена пароля и выход из учетной записи.

Содержание отчета

– Титульный лист;
– Цель работы;
– Результаты выполнения заданий;
– Вывод.

Контрольные вопросы

1. Что такое система мониторинга?
2. Что входит в комплексную систему мониторинга?
3. Приведите примеры систем мониторинга?
4. Для чего используется зонтичная система мониторинга?
5. Зачем используется зонд агрегации в Системе мониторинга?

7.1.2 Изучение характеристик различных каналов связи сети пакетной передачи данных в Системе мониторинга.

Ключевые слова

Система мониторинга, коэффициент потерь пакетов, время задержки, вариация времени задержки, коэффициент пропускной способности.

Цель работы

Изучить характеристики измерения каналов связи;

Изучить разницу в измеренных характеристиках для различных линий связи;

Изучить влияние соглашения об уровне предоставления услуг SLA на отображение данных на графиках.

Краткие теоретические сведения

Коэффициент потери пакетов – это отношение количества пакетов данных, потерянных во время теста, к общему количеству отправленных пакетов.

Время задержки передачи пакетов – время, требуемое единице информации для достижения приёмника.

Вариация времени задержки – описывает вариабельность времени прихода пакетов данных в компьютерных сетях.

Канал связи – часть сети связи, связывающая между собой каждую пару ее оконечных устройств и состоящая из технических средств передачи и приема данных, включая линию связи, а также средств программного обеспечения и протоколов.

Каналы связи бывают: персональные – обеспечивают прямое общение между отправителем и получателем (например, по телефону); безличные – адресованы большому количеству получателей (радио); односторонние – средства массовой информации (телевидение); двунаправленные – связь осуществляется через канал постоянного прямого взаимодействия между отправителем и получателем (телефонная связь).

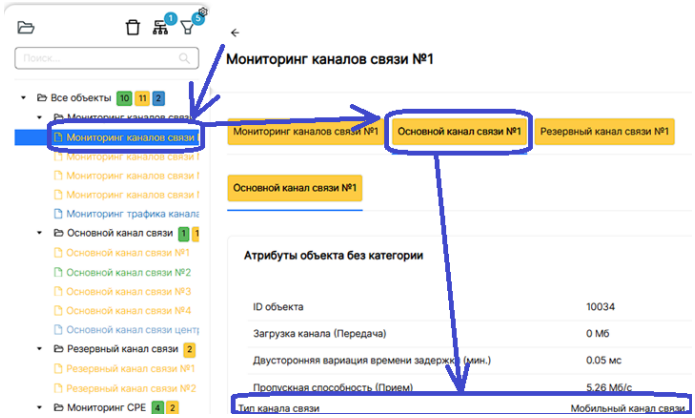
Линия связи – совокупность технических устройств и физической среды распространения сигналов от передатчика к приёмнику, обеспечивающая образование одного (одноканальный) или нескольких (многоканальный) каналов связи.

К типам линий связи относятся:

- витая пара, то есть 2 провода, изолированных и свитых между собой (телефонный кабель, например); достоинства: невысокая стоимость; недостатки: невысокая скорость передачи данных, чувствительность к помехам;
- экранированная витая пара является усовершенствованием витой пары и позволяет повысить помехозащищённость и скорость передачи; недостаток: повышается цена на этот тип канала связи по сравнению с обычной витой парой;
- коаксиальный кабель, если сравнить с витой парой, имеет большую механическую прочность и помехозащищённость; виды: толстый (лучше передает данные) и тонкий; недостаток: дороже, по сравнению с витой парой;
- оптоволоконный кабель считается идеальным каналом, так как не подвергается электромагнитным полям и почти не имеет излучения; преимущество: высокая скорость;
- беспроводное оборудование при определенных обстоятельствах оказывается дешевле по сравнению с кабельным; положительные стороны: объединит ряд компьютеров в сеть даже там, где невозможно провести кабель;
- радиорелейные линии связи, как правило, используются для обеспечения телефонных каналов связи;
- спутниковый доступ работает по такому принципу: сигнал со станции Земли посылается на спутник, где усиливается, обрабатывается и посылается снова на Землю; преимущества: обеспечивает связь на больших расстояниях.
- лазерная связь относится к беспроводным оптическим системам связи. Достоинства: высокая пропускная способность, эффективность при низком коэффициенте шума, недорогая стоимость, низкая мощность, гибкость и устойчивость к радиопомехам; недостатки: передатчик и приемник должны находиться в пределах прямой видимости.

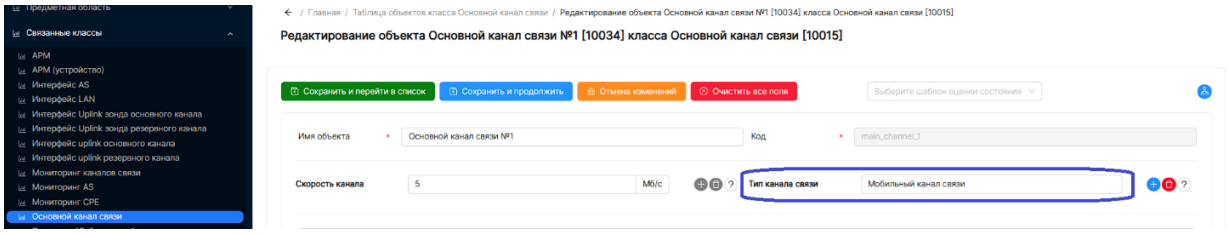
Где посмотреть информацию о типе канала связи в Системе мониторинга?

В атрибутах объекта «Основной канал связи» или «Резервный канал связи».



Где можно изменить информацию о типе канала связи в Системе мониторинга?

В интерфейсе «Менеджер» выбрать «Связанные классы», далее выбрать требуемый объект «Основной канал связи» или «Резервный канал связи» и найти атрибут выбранного объекта «Тип канала связи».



Качество услуги связи

SLA (Service Level Agreement) – это степень соответствия показателей, характеризующих потребительские характеристики услуги, требованиям, предъявляемым к показателям функционирования сети связи, и требованиям, закрепленным договором об оказании услуг связи и/или соглашением между оператором связи и абонентом об уровне качества.

Особенности SLA для различных услуг каналов связи:

- При аренде выделенных каналов основными параметрами SLA являются надёжность и качество передачи. Как правило, перед передачей канала клиенту оператор проводит специальные испытания, во время которых измеряются параметры SLA и подтверждается их соответствие принятым нормам.

- При работе по каналам Frame Relay и ATM кроме параметров надёжности и качества передачи в SLA важны параметры производительности (информационная скорость и задержка). Контроль параметров каналов FR и ATM может осуществляться средствами мониторинга сети, анализаторами и с помощью инструментальных средств сбора информации от коммутаторов и зондов.
- Для услуги интегрированной передачи голоса и данных в IP-сетях также важны параметры надёжности, качества и производительности. Операторы, предоставляющие данную услугу, часто используют собственные системы измерения параметров SLA, которые определяют такие объективные данные о качестве голосовой связи, как количество потерянных пакетов, уровни шума и задержек.
- Для услуг VPN особенностью SLA является определение параметров безопасности, качества обслуживания, управляемости и надёжности.
- При реализации SLA на услуги прикладного уровня важным является время отклика приложения и обеспечение сохранности информации.

Имитация задержек на канале связи

В лабораторной работе производится имитация предоставления услуг для двунаправленного канала связи пакетной сети передачи данных. Имитируются следующие каналы связи: мобильный, спутниковый, медный, Wi-Fi.

Для моделирования характеристик используется модуль ядра netem операционной системы Linux, который управляется утилитой tc из пакета iproute2.

Параметры настроек интерфейсов для имитации задержек с помощью утилиты tc на Зонде КМУТ CPE1-M5:

```
root@CPE1-M5:~# tc qdisc add dev eth0 root netem delay 200ms 100ms 50 loss 5% 2%
```

Здесь задана задержка на интерфейсе eth0 в промежутке от 100 до 300 миллисекунд и будет 50-процентная корреляция со значением задержки для предыдущего пакета, потеря 5% пакетов с корреляцией 2%.

```
root@CPE1-M5:~# tc qdisc add dev eth1 root netem delay 700ms 200ms 40 loss 10% 5%
```

Здесь задана задержка на интерфейсе eth1 в промежутке от 500 до 900 миллисекунд и будет 40-процентная корреляция со

значением задержки для предыдущего пакета, потеря 10% пакетов с корреляцией 5%.

Параметры настроек интерфейсов для имитации задержек с помощью утилиты tc на Зонде КМУТ CPE2-M5:

```
root@CPE2-M5:~# tc qdisc add dev eth1 root netem delay 40ms 20ms loss 5% 2%
```

Здесь задана задержка на интерфейсе eth1 в промежутке от 20 до 60 миллисекунд, потеря 5% пакетов с корреляцией 2%.

Вместо команды add (добавление правила) можно использовать команду change для изменения правила или команду del для удаления правила.

Параметры настроек интерфейсов для имитации задержек сбрасываются при перезагрузке Зонда КМУТ. В случае отсутствия имитации задержек на графиках Системы мониторинга необходимо ввести повторно в консольной оболочке необходимые на Зондах КМУТ команды.

Используемые технические средства.

- АРМ с установленным веб-браузером;
- Сервер с установленной системой мониторинга КМУТ ML;
- Зонды КМУТ CPE1-M5, CPE2-M5.

Задание

Ознакомиться с системой мониторинга, зафиксировать в таблице измеренные значения. Оценить, входят ли полученные значения в требования SLA.

Алгоритм выполнения лабораторной работы

1. Изучить расположение различных каналов связи на объектах по сетевой схеме Лаборатории (п.13). Как правило, для объектов с двумя каналами связи основной канал связи имеет наименьшее значение времени задержек передачи пакетов. Таким образом, для объекта №1 основным каналом связи будет мобильный, а резервный - спутниковый. Для объекта №2 основной канал связи - медный, резервный - Wi-Fi;
2. Открыть браузер, перейти по адресу https://10.19.47.252. Ввести данные для авторизации и нажать кнопку «Войти»;
3. Зайти в объект, имеющий мобильный канал связи. Найти график с измеряемой характеристикой, зафиксировать значение в соответствующей ячейке таблицы. Оценить, удовлетворяют ли измеренные значения требованиям SLA– потери не более 2%,

задержки не более 80 мс, вариация задержек не более 30 мс, коэффициент пропускной способности более 95%;

4. Зайти в объект, имеющий спутниковый канал связи. Найти график с измеряемой характеристикой, зафиксировать значение в соответствующей ячейке таблицы. Оценить, удовлетворяют ли измеренные значения требованиям SLA – потери не более 5%, задержки не более 800 мс, вариация задержек не более 50 мс, коэффициент пропускной способности более 95%;

5. Зайти в объект, имеющий медный канал связи. Найти график с измеряемой характеристикой, зафиксировать значение в соответствующей ячейке таблицы. Оценить, удовлетворяют ли измеренные значения требованиям SLA – потери не более 1%, задержки не более 10 мс, вариация задержек не более 5 мс, коэффициент пропускной способности более 95%;

6. Зайти в объект, имеющий канал связи Wi-Fi. Найти график с измеряемой характеристикой, зафиксировать значение в соответствующей ячейке таблицы. Оценить, удовлетворяют ли измеренные значения требованиям SLA – потери не более 1%, задержки не более 50 мс, вариация задержек не более 20 мс, коэффициент пропускной способности более 95%.

Содержание отчета

- Титульный лист;
- Цель работы;
- Результаты выполнения заданий;

Результат включает в себя заполнение таблицы:

	Мобильный		Спутниковый		Медный		Wi-Fi	
	Значение	SLA	Значение	SLA	Значение	SLA	Значение	SLA
Двусторонний коэффициент потерь пакетов, %								
Двусторонняя задержка передачи пакетов (сред.), мс								
Двусторонняя вариация времени задержки (сред.), мс								
Относительная пропускная способность (Прием), %								

- Анализ и выводы.

Контрольные вопросы

1. Что такое канал связи?
2. Что такое линия связи?
3. Назвать основные характеристики каналов связи?

4. Назвать типы каналов связи?
5. Привести примеры линий связи?
6. Что такое система мониторинга?

7.1.3. Изучение состава трафика в Системе мониторинга.

Ключевые слова:

Трафик, состав трафика, генерация трафика, утилита mausezahn.

Цель работы

Изучить состав трафика, выполняемый для типов трафика различных протоколов: туннельный; голосовой; почтовый; ftp, web, и прочий.

Изучить возможность «окрашивания» различных типов трафика для обеспечения гарантированного качества необходимого сервиса в Системе мониторинга. Изучить работу QoS.

Изучить функции генерации трафика с помощью утилиты mausezahn.

Важное замечание:
Во избежание рисков сбоя в работе Системы мониторинга КМУТ желательно не выполнять каких-либо действий по изменению функций, а только выполнить визуальное ознакомление с действующими настройками без внесения каких-либо изменений.

Краткие теоретические сведения

Рассмотрим распространенные типы трафика.

Туннельный трафик (Tunneling)

Это трафик, который инкапсулирует («упаковывает») один сетевой протокол в другой. Он используется для создания защищенных соединений (VPN) или передачи данных между филиалами через публичную сеть (Интернет).

Ключевые характеристики:

- Высокая важность: часто является магистральным, то есть через туннель проходит весь остальной трафик сети (голос, данные, видео). Если туннель «падает», пропадает связь целиком;
- Чувствительность к потере пакетов (Loss-Sensitive): потеря инкапсулирующих пакетов ведет к потере всего исходного пакета данных;

- Чувствительность к задержкам (Latency-Sensitive): добавляет накладные расходы на инкапсуляцию/деинкапсуляцию.

Основные протоколы туннельного трафика:

- IPsec (AH/ESP): стандарт для secure VPN;
- GRE: простой туннельный протокол без шифрования;
- SSL/TLS: основа для VPN типа SSL (например, OpenVPN, AnyConnect);
- L2TP: часто используется в паре с IPsec.

Голосовой трафик (VoIP — Voice over IP)

Это трафик реального времени, передающий оцифрованную человеческую речь.

Ключевые характеристики:

- Сверхнизкая задержка (Low Latency): задержка более 150 мс становится заметной;
- Низкий джиттер (Low Jitter): колебания задержки должны быть минимальными;
- Умеренная чувствительность к потерям (Loss-Sensitive): благодаря кодекам может переносить невысокий процент потерь (1–2%);
- Неэластичный (Inelastic): если пакет опоздал, он бесполезен и выбрасывается;
- Постоянная скорость (Constant Bit Rate): пакеты генерируются с постоянным интервалом.

Основные протоколы голосового трафика:

- RTP (Real-time Transport Protocol): непосредственно передает голосовые данные. Именно его нужно приоритизировать;
- RTCP (RTP Control Protocol): управляет качеством передачи RTP;
- SIP (Session Initiation Protocol): протокол сигнализации, который устанавливает, управляет и завершает сеанс связи (звонок). Важен, но не так критичен к задержкам, как RTP.

Видеотрафик (Video)

Видеотрафик делится на два основных типа:

А. Интерактивное видео (Видеоконференции)

Характеристики: почти идентичны голосовому трафику: низкая задержка, низкий джиттер, неэластичность. Требует больше полосы пропускания.

Протоколы: RTP (для данных), RTCP, SIP, H.323.

B. Потоковое видео (Streaming Video — YouTube, Netflix)

Характеристики:

- Буферизация: может буферизовать данные, поэтому менее чувствительно к задержкам и джиттеру;
- Чувствительность к потере пакетов: высокая, так как потери ведут к артефактам изображения;
- Эластичный (Elastic): может адаптироваться к доступной bandwidth (например, понижать качество);

Протоколы: RTMP (устаревший), HLS, MPEG-DASH, WebRTC (для интерактива в браузере).

Почтовый трафик (E-mail)

Это трафик приложений для отправки и получения электронной почты.

Ключевые характеристики:

- Допускает задержки (Delay-Tolerant): письмо может идти несколько секунд или даже минут — это некритично;
- Чувствительность к потере пакетов (Loss-Sensitive): потеря пакета с частью письма недопустима. Протоколы требуют повторной передачи;
- Взрывной характер (Bursty): передача больших вложений создает интенсивную нагрузку на канал.

Основные протоколы:

- SMTP (Simple Mail Transfer Protocol, порт 25): для отправки почты между серверами;
- POP3 (Post Office Protocol v3, порт 110): для получения почты с сервера на локальное устройство (обычно с удалением с сервера);
- IMAP (Internet Message Access Protocol, порт 143): для управления почтой непосредственно на сервере (синхронизация папок, статусов и т. д.).

Веб-трафик (Web — HTTP/HTTPS)

Это трафик просмотра веб-страниц.

Ключевые характеристики:

- Взаимодействие с пользователем: пользователь ожидает отклика, поэтому большие задержки (>1-2 сек) воспринимаются негативно;
- Чувствительность к потерям (Loss-Sensitive): потеря пакетов приводит к повторным передачам и увеличению времени загрузки;
- Взрывной характер (Bursty): при открытии страницы происходит множество одновременных подключений для загрузки HTML, CSS, изображений, JS;
- Эластичный (Elastic): может «уступить» более важному трафику, просто загружаясь медленнее.

Основные протоколы:

- HTTP (порт 80): передает данные в открытом виде;
- HTTPS (HTTP over SSL/TLS, порт 443): шифрованный веб-трафик. Важно: из-за шифрования невозможно заглянуть внутрь и классифицировать трафик по содержимому, только по IP-адресу/порту.

Трафик передачи файлов (FTP и подобные)

Это трафик, основной задачей которого является перемещение файлов между системами.

Ключевые характеристики:

- Высокая пропускная способность (Throughput-Intensive);
- Высокая чувствительность к потерям (Loss-Sensitive): любая ошибка приводит к повторной передаче части файла;
- Допускает задержки (Delay-Tolerant): не критично, если файл скачается за 10 секунд или за 15.

Основные протоколы:

- FTP (File Transfer Protocol, порты 20/21): классический протокол;
- SFTP (SSH File Transfer Protocol, порт 22): FTP поверх SSH, безопасный;
- TFTP (Trivial FTP, порт 69): для простых передач (например, загрузка ПО на сетевые устройства).

Прочий трафик (служебный, сетевой)

Это трафик, которым обмениваются сами сетевые устройства и операционные системы для своей работы.

Ключевые характеристики: критически важен для функционирования сети, но обычно генерирует очень мало данных.

Основные протоколы:

- BGP, OSPF, EIGRP: протоколы маршрутизации. Потеря их пакетов может привести к сбоям в сети;
- ICMP: ping, traceroute. Используется для диагностики;
- DNS (порт 53): преобразует имя сайта (ya.ru) в IP-адрес. Крайне важен, так как от него зависит работа почти всех сетевых приложений. Чувствителен к задержкам;
- NTP (порт 123 UDP): синхронизация времени. Важен для безопасности, логирования. Чувствителен к джиттеру.

Качество обслуживания QoS

QoS — это набор технологий и механизмов, используемых в сетях для управления производительностью и обеспечением предсказуемого уровня обслуживания для критически важного трафика.

Основная цель:

Гарантировать, что важный трафик (например, голосовая связь, видео, данные реального времени) будет доставлен с минимальными задержками, дрожанием (jitter) и потерей пакетов, даже в условиях перегрузки канала.

Ключевые концепции QoS:

- Пропускная способность (Bandwidth): Максимальная скорость передачи данных по каналу;
- Задержка (Latency): Время, которое требуется пакету для прохождения от источника к получателю;
- Дрожание (Jitter): Изменение задержки между пакетами. Критично для голоса и видео;
- Потери пакетов (Packet Loss): Процент пакетов, которые не дошли до получателя.

Основные механизмы реализации QoS:

Классификация и маркировка:

- Классификация: Идентификация трафика по типу (например, голос, видео, веб) с помощью ACL, портов и т.д.;

- Маркировка: Помечание пакетов меткой, указывающей их приоритет. Происходит на L2 (Frame Relay DE, 802.1Q/p CoS) и L3 (IP Precedence, DSCP).

Очереди (Queuing):

- FIFO (First-In, First-Out): Простая очередь без приоритетов;
- Priority Queuing (PQ): Строгий приоритет. Высокоприоритетная очередь обслуживается всегда, пока не опустеет;
- Custom Queuing (CQ): Выделение фиксированной доли пропускной способности для каждого типа трафика;
- Weighted Fair Queuing (WFQ): «Честное» распределение между потоками, с учетом весов;
- Class-Based Weighted Fair Queuing (CBWFQ): Расширение WFQ, позволяющее создавать классы трафика и назначать им гарантированную пропускную способность;
- Low Latency Queuing (LLQ): Комбинация PQ и CBWFQ. Для голоса и видео создается очередь строгого приоритета, которая обслуживается в первую очередь, чтобы минимизировать задержку и джиттер.

Управление перегрузкой (Congestion Management):

- Полиция трафика (Traffic Policing): отбрасывает или remark'ает трафик, превышающий установленный лимит;
- Формирование трафика (Traffic Shaping): буферизует избыточный трафик и передает его позже, чтобы "сгладить" поток и избежать перегрузки;
- Избежание перегрузок (Congestion Avoidance):
- Weighted Random Early Detection (WRED): предотвращает переполнение очередей, selectively отбрасывая пакеты до наступления перегрузки. Предпочтение отдается пакетам с низким приоритетом.

Тип сервиса TOS

TOS — это устаревшее 8-битное поле в заголовке IPv4, изначально предназначенное для указания приоритета и требований к маршрутизации пакета.

Эволюция поля TOS:

IP Precedence (Приоритет IP):

- Старая модель, использовавшая только первые 3 бита поля TOS;
- Позволяла задать 8 уровней приоритета (0-7), где 7 — наивысший (сетевой контроль), 5 — голос, 0 — обычный трафик;
- Недостаток: слишком мало градаций для гибкого управления;

DSCP (Differentiated Services Code Point — Кодовая точка дифференцированных услуг):

- Современная модель, использующая первые 6 бит поля TOS (переименованного в DS Field);
- Позволяет создать 64 возможных класса обслуживания (0-63), что обеспечивает гораздо более гибкую классификацию трафика;

Структура DSCP:

- Класс селектора (Class Selector): Значения, обратно совместимые с IP Precedence (например, CS0, CS1, ..., CS7);
- Assured Forwarding (AF): 4 класса (AF1x, AF2x, AF3x, AF4x), в каждом по 3 уровня вероятности отбрасывания (низкий, средний, высокий). Например, AF31 — трафик с высоким приоритетом, но который можно отбросить при перегрузке;
- Expedited Forwarding (EF): Значение 46 (101110). Предназначен для трафика с низкой задержкой и джиттером (голос, видео);

Последние 2 бита поля (ECN — Explicit Congestion Notification) используются для явного уведомления о перегрузке;

Рассмотрим связь TOS и QoS:

- TOS (а точнее, поле DS / DSCP) — это инструмент маркировки трафика на 3-м (сетевом) уровне модели OSI.
- QoS — это общая архитектура и набор механизмов, которые используют эту маркировку для принятия решений о приоритизации, организации очередей и управлении трафиком на сетевых устройствах (маршрутизаторах, коммутаторах).

Генератор трафика Mausezahn

На Зонде агрегации ZA-M11 установлена утилита **Mausezahn** – это генератор трафика.

Назначение генераторов трафика:

- Тестирования производительности сети — моделируют различные типы трафика (от L2/L3 до сложных взаимодействий на уровне приложений) с настраиваемыми параметрами (адрес источника, адрес назначения, пропускная способность и др.);
- Эмуляции сетевых протоколов — генераторы могут имитировать поведение на уровне управления соседнего сетевого узла (протоколы маршрутизации и коммутации, MPLS и др.);
- Генерации трафика приложений — для функциональных тестов, основанных на распознавании приложений более высокого уровня.

Задание

1. Изучить теоретические сведения;
2. Изучить состав трафика в Системе мониторинга КМУТ ML;
3. Научиться «окрашивать» различные типы трафика;
4. Выполнить анализ данных по проделанной работе;
5. Сделать выводы.

Алгоритм выполнения лабораторной работы

1. В интерфейсе «Менеджер» в разделе «Измерительная система» выбрать «Тип трафика» (рис. 27).

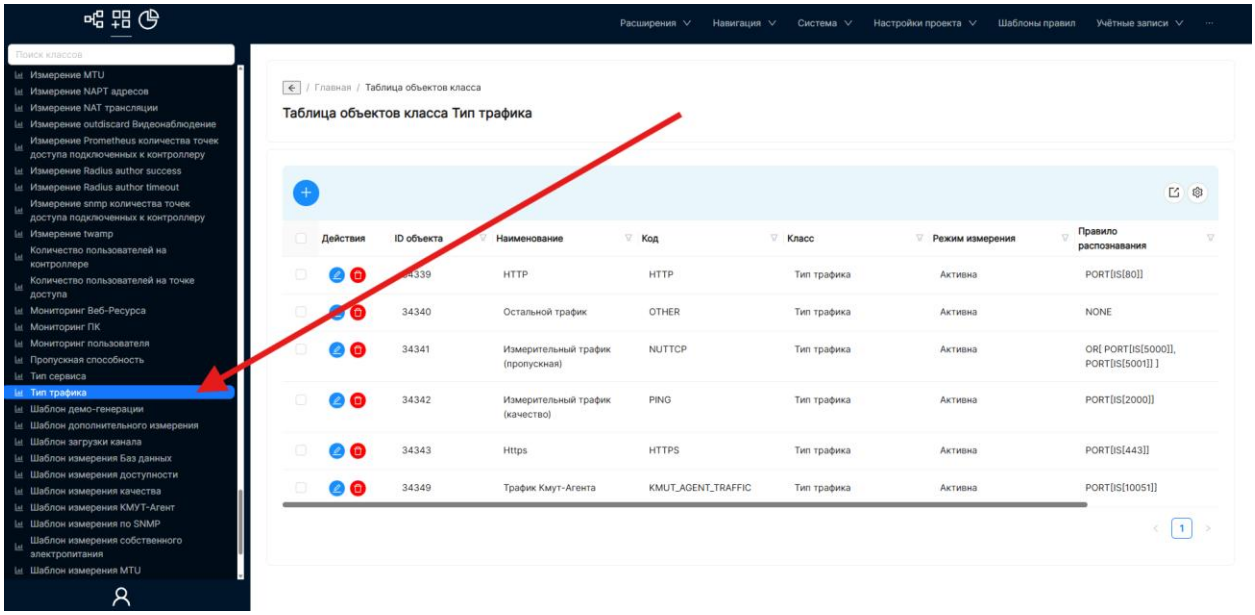


Рис. 27. Таблица «Тип трафика»

2. Изучить состав трафика и правила распознавания типов трафика.

3. Перейти к созданию типа трафика и изучить заполняемые атрибуты (рис. 28).

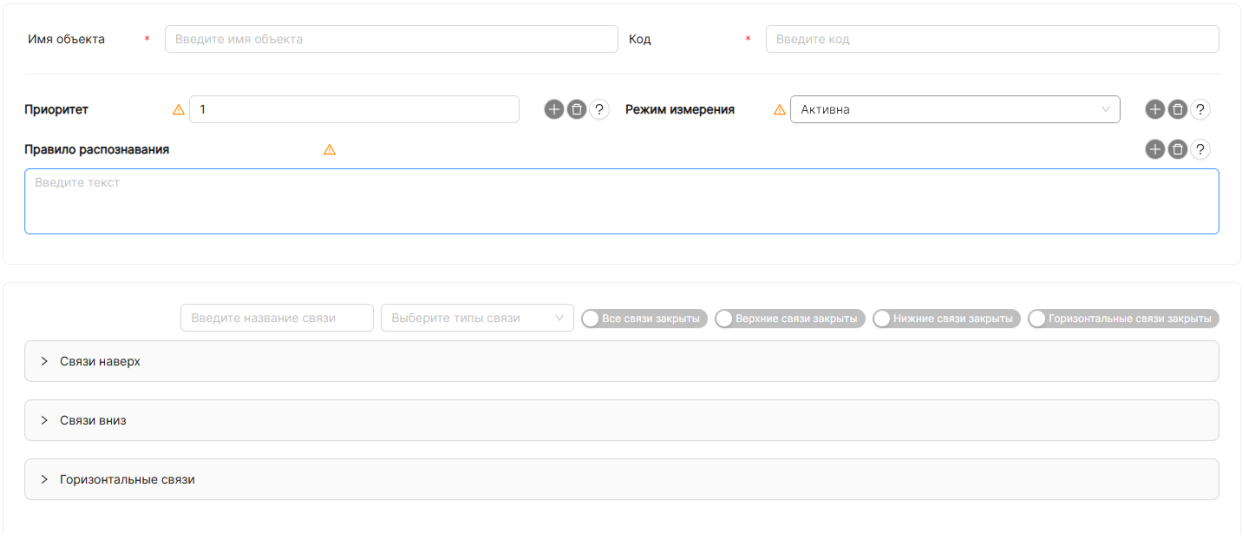


Рис. 28. Создание типа трафика в интерфейсе «Менеджер»

4. Изучить создание правил распознавания трафика (рис. 29).

Правило распознавания

Примеры:

1. NONE
2. PORT[IS[80]]
3. PORT[IS[443]]
4. OR[PORT[IS[5000]], PORT[IS[5001]]]

Рис. 29. Примеры правил распознавания

5. В разделе в разделе «Измерительная система» выбрать «Тип сервиса» (рис. 30).

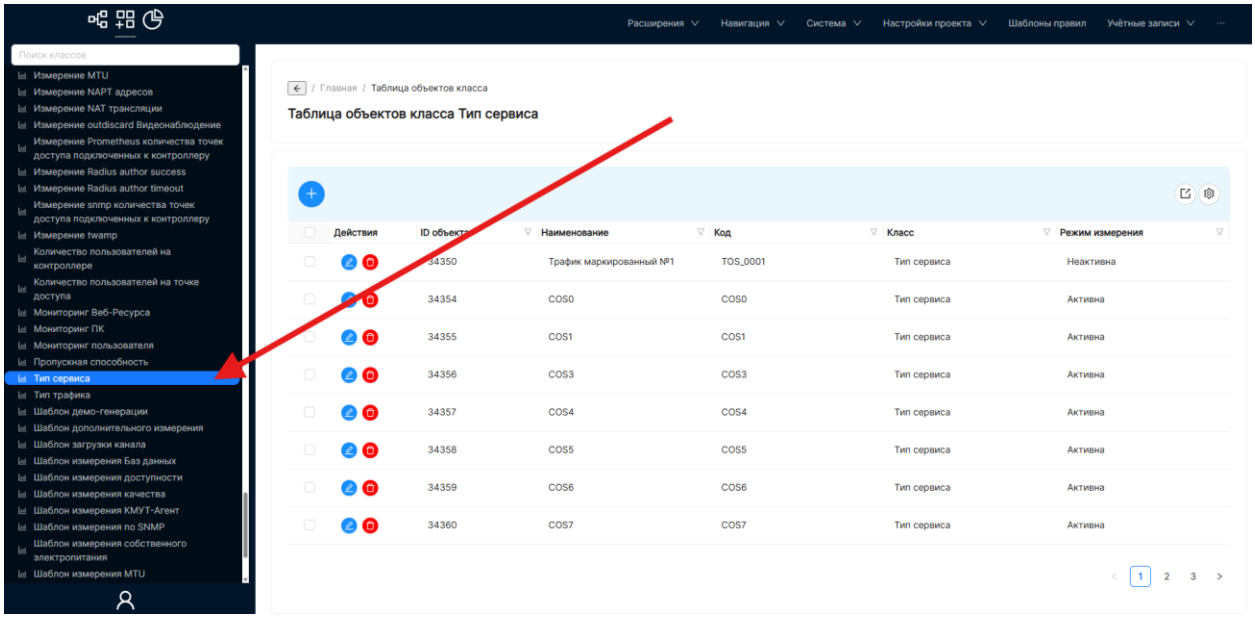


Рис. 30. Таблица «Тип сервиса»

5. Перейти к созданию типа сервиса и изучить заполняемые атрибуты (рис. 31).

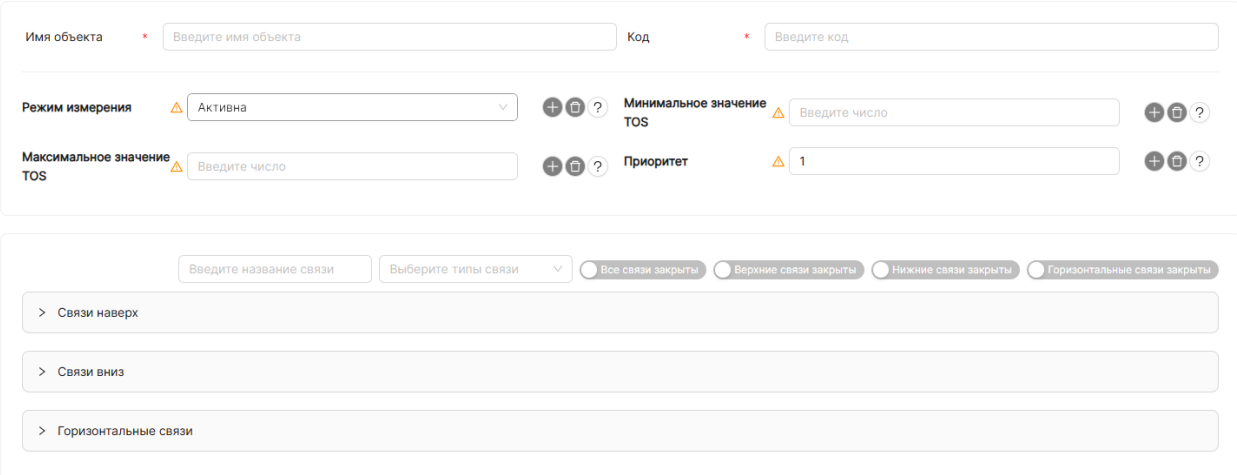


Рис. 31. Создание типа сервиса в интерфейсе «Менеджер»

6. Работа с генератором трафика. С сервера Системы КМУТ выполнить вход на Зонд агрегации и запустить с правами администратора утилиту **mausezahn** со следующими данными:

```
mtusiadmin@kmut-ml-mtusi:~$ ssh -p 8022 user@10.19.47.5
Authorized use only. All activity may be monitored and reported.
user@10.19.47.5's password:
Linux ZA-M11 6.6.63-rt46-kmut x86_64
-----------------------
   Welcome to probe!
-----------------------
To configure BGP please use "vtysh" command.
Last login: Fri Sep 26 07:51:27 2025 from 10.19.47.252
user@ZA-M11:~$ su -
Password:
root@ZA-M11:~# mausezahn eth0 -A rand -B 198.18.10.2 -t rtp id=11:11:11:11
```

7. Объяснить назначение данной команды.

8. Далее выполним анализ трафика. Не выключая предыдущую команду, с Зонда агрегации выполнить вход на Зонд CPE1-M5 и запустить с правами администратора утилиту **tcpdump** для снятия дампов трафика. Зафиксировать полученные результаты.

```
root@ZA-M11:~# ssh -p 8022 user@198.18.10.2
Authorized use only. All activity may be monitored and reported.
user@198.18.10.2's password:
Linux CPE1-M5 5.10.0-0.bpo.9-rt-amd64 x86_64
-----------------------
   Welcome to probe!
-----------------------
Last login: Fri Sep 26 07:09:33 2025 from 198.18.10.1
user@CPE1-M5:~$ su -
Password:
root@CPE1-M5:~# tcpdump -v -vv -n -nn -i eth0 | grep 30000
```

При успешных действиях на порт Зонда CPE1-M5 будет поступать сгенерированный трафик утилитой **mausezahn** по порту 30000:

```
64.5.117.128.30000 > 198.18.10.2.30000: [udp sum ok] UDP, length 172
83.196.140.128.30000 > 198.18.10.2.30000: [udp sum ok] UDP, length 172
194.169.121.0.30000 > 198.18.10.2.30000: [udp sum ok] UDP, length 172
130.247.128.0.30000 > 198.18.10.2.30000: [udp sum ok] UDP, length 172
```

Подробнее про анализ трафика можно ознакомиться в лабораторной работе 7.2.2.

Содержание отчета

1. Титульный лист;
2. Цель работы;
3. Теоретические сведения;
4. Описания выполнения работы (по шагам);
5. Полученные результаты (скриншоты);
6. Анализ и выводы.

Контрольные вопросы

1. Перечислите типы трафика.
2. Какие типы трафика чувствительны к задержке?
3. Какие типы трафика требуют большую пропускную способность?
4. Зачем нужен и как устроен QoS?
5. Какие очереди и обработки перегрузки существуют?
6. Где и для чего применяется TOS?
7. Зачем используют генераторы трафика?

7.1.4. Изучение схем включения Зондов мониторинга.

Ключевые слова

Схема включения Зонда КМУТ, роль «маршрутизатор», включение в разрыв, Т-схема.

Цель работы

Изучить схемы включения Зондов мониторинга. Изучить особенности реализации каждой схемы включения, выявить достоинства и недостатки каждой схемы включения.

Краткие теоретические сведения

Для выполнения услуг мониторинга и управления трафиком существуют несколько вариантов подключения Зонда КМУТ.

Рассмотрим эти схемы.

Роль «маршрутизатор»

Зонд КМУТ может выступать в роли маршрутизатора, осуществляю маршрутизацию между несколькими каналами связи, с помощью статической или динамической маршрутизации в соответствии с протоколом динамической маршрутизации BGP (рис. 32).

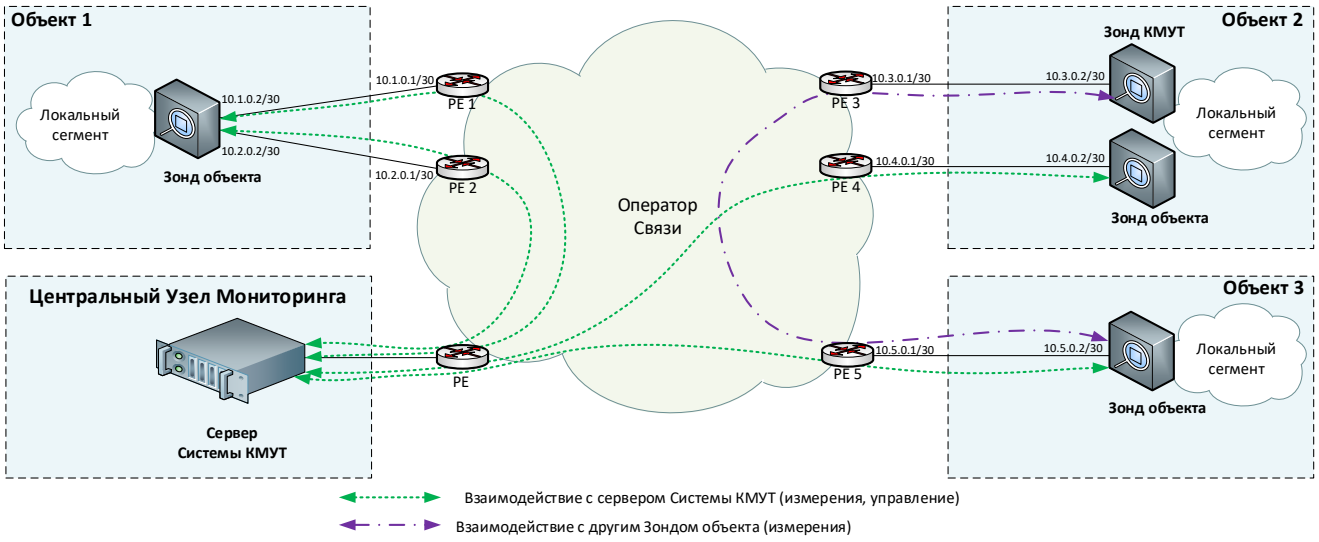


Рис. 32а. Подключение Зонда КМУТ в роли «маршрутизатор» на сети связи

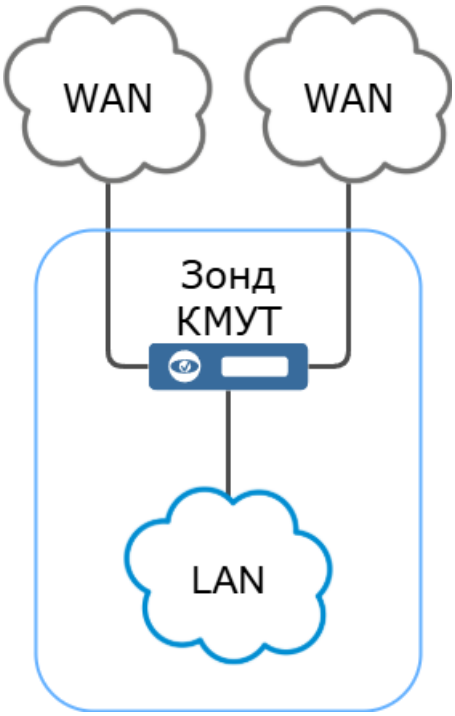


Рис. 32б. Схематичное изображение подключения Зонда КМУТ в роли «маршрутизатор»

Подключение Зонда КМУТ «в разрыв»

Зонд КМУТ устанавливается в разрыв в «прозрачном режиме» таким образом, чтобы весь телекоммуникационных трафик объекта заказчика проходил через Зонд КМУТ. При этом Зонд КМУТ не вносит никаких изменений в проходящий трафик.

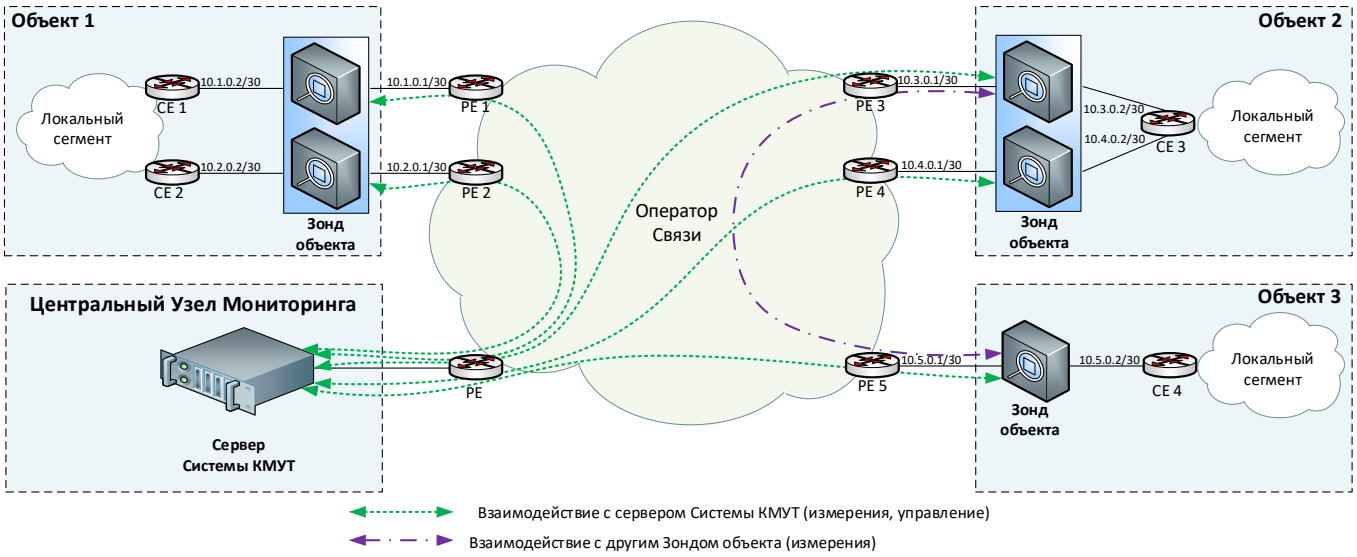


Рис. 33а. Подключение Зонда КМУТ «в разрыв» на сети связи

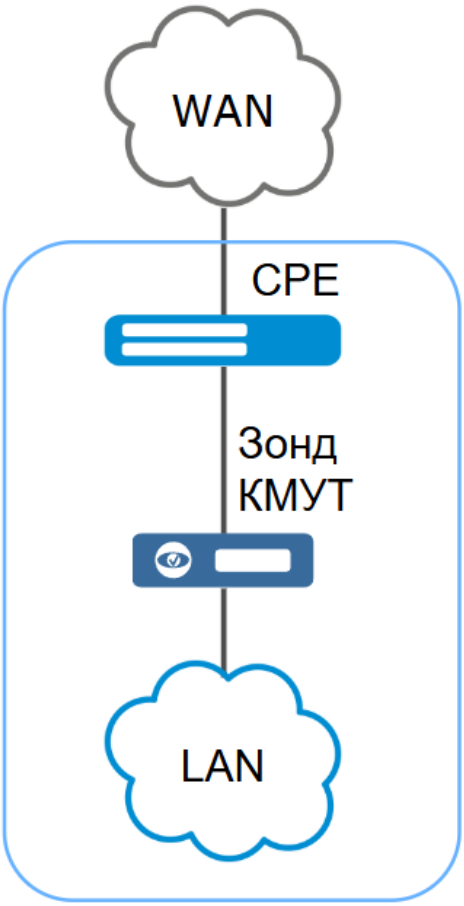


Рис. 33б. Схематичное изображение подключения Зонда КМУТ в разрыв канала связи

Подключение Зонда КМУТ в Т-образном режиме работы

В данном режиме Зонд КМУТ подключается одним сетевым интерфейсом к маршрутизатору или сетевому интерфейсу другого оборудования и обеспечивает получение информации от данного оборудования. Информация, полученная Зондом КМУТ, используется для проведения тестов, связанных с измерением качества канала связи и максимально доступной пропускной способности (рис. 34).

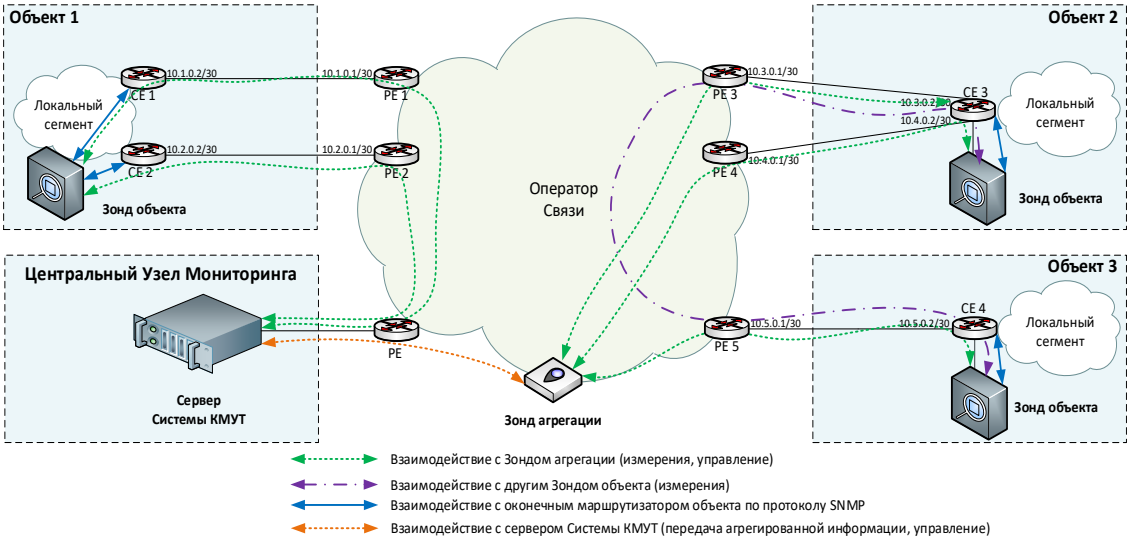


Рис. 34а. Подключение Зонда КМУТ в Т-образном режиме работы на сети связи

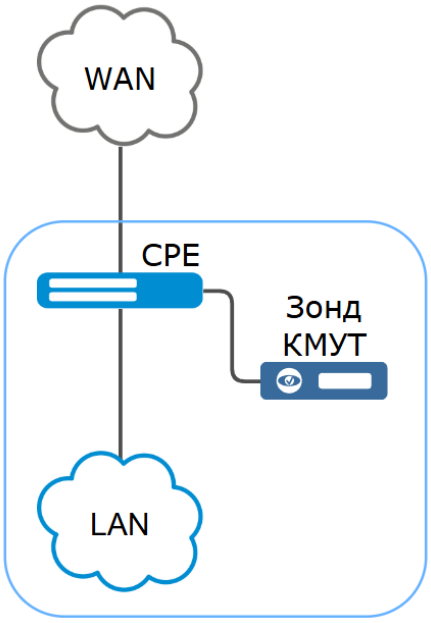


Рис. 34б. Схематичное изображение подключения Зонда КМУТ в Т-образном режиме работы

Задание

Спроектировать в графическом редакторе Draw.io три принципиальные схемы, иллюстрирующие различные методы интеграции устройства мониторинга (пробника) в сегмент сети между маршрутизатором и сервером. Каждая схема должна наглядно демонстрировать физические соединения, направление потоков данных и роль ключевых устройств:

Параллельное подключение (Т-схема);

Последовательное подключение (Включение в разрыв);

Подключение на основе маршрутизатора.

Алгоритм выполнения лабораторной работы

1. На АРМ откройте программу Draw.io;
2. Найдите нужные элементы в левой панели (библиотеке):

Элементы: Интернет, оборудование провайдера, зонд, оборудование клиента;

3. Нарисуйте: Сервер → Оборудование провайдера → Оборудование клиента;
4. Отобразите подключение зонда в оборудование провайдера в Т-образном подключении;
5. Нарисуйте: Сервер -> Оборудование оператора -> [ПРОБЕЛ] -> Оборудование клиента;
6. Нарисуйте: Оборудование провайдера -> Маршрутизатор -> Оборудование клиента -> Сервер;
7. Замените в данной цепочке маршрутизатор зондом который принимает в себя два канала от оборудования провайдера;
8. Сохраните файл.

Содержание отчета

1. Титульный лист.
2. Цель работы.
3. Теоретические сведения.
4. Описание выполнения работы (по шагам).
5. Полученные результаты.
6. Анализ и выводы.

Контрольные вопросы

1. Какие три основные функции (роли) может выполнять изделие «Зонд КМУТ» при его внедрении в сеть заказчика? Перечислите их и дайте краткое описание.
2. В чём заключается ключевое отличие в обработке трафика при установке Зонда КМУТ «в разрыв» от его работы в роли маршрутизатора?
3. Какой режим работы Зонда КМУТ — «в разрыв» или «Т-образный» — является полностью пассивным и не создаёт риска остановки обслуживания трафика в случае выхода самого зонда из строя? Обоснуйте свой ответ.

4. Какой протокол динамической маршрутизации поддерживает Зонд КМУТ при подключении в роли маршрутизатора?

5. Найти на сетевой схеме лаборатории применяемые схемы включения Зондов КМУТ (п.12).

7.1.5. Изучение отчетных форм по результатам мониторинга.

Ключевые слова

Отчеты, аналитика, статистика.

Цель работы

Изучить отчетные формы в Системе КМУТ. Выполнить аналитику действующих в Системе КМУТ услуг. Выполнить сопоставление данных на графиках и в отчетных материалах.

Краткие теоретические сведения

Отчёты в Системе КМУТ нужны для обеспечения достоверной и своевременной информации для принятия решений и оценки эффективности работы диагностируемой мониторингом инфраструктуры.

Отчёты позволяют:

- выявить тенденции в различных ИТ-активах организации;
- определить проблемные места;
- выявить уязвимости в работе оборудования на ранних стадиях, что даёт возможность принимать меры для их устранения и минимизации возможных потерь;
- прогнозировать будущее развитие инфраструктуры организации;
- повысить прозрачность функционирования оборудования внутри организации, что помогает лучше понять текущее состояние инфраструктуры;
- оптимизировать процессы. Можно выявить узкие места в работе оборудования, что позволяет оптимизировать их и сэкономить ресурсы.

Модуль «Отчёты» предназначен для построения отчетов по форме пользователя или в свободной форме.

Обычно отчёты в Системе мониторинга могут быть сгенерированы в распространенных форматах – XLSX, PDF, DOCX, ODT, CSV и пр. В Системе мониторинга Лаборатории Л400 доступна выгрузка отчётов в формате XLSX.

Модуль отчетов представлен таблицей заданий на построение отчетов, а также формой создания задания на построение. В таблице заданий в виде списка выводятся созданные

пользователями системы задания на построение отчетов, включая тип отчета, генерируемые форматы, интервал дат построения, текущий статус выполнения задания и инструменты скачивания готового отчета (рис. 35).

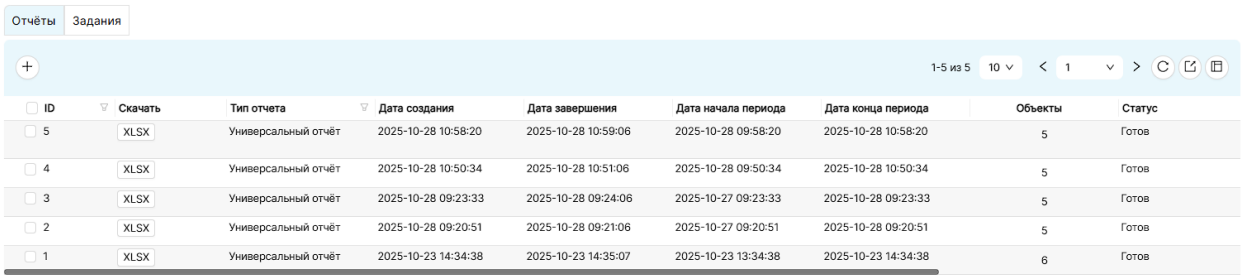


Рис. 35. Таблица заданий

Для видоизменения таблицы заданий необходимо нажать кнопку  . Появится окно редактирования таблицы (рис. 36).

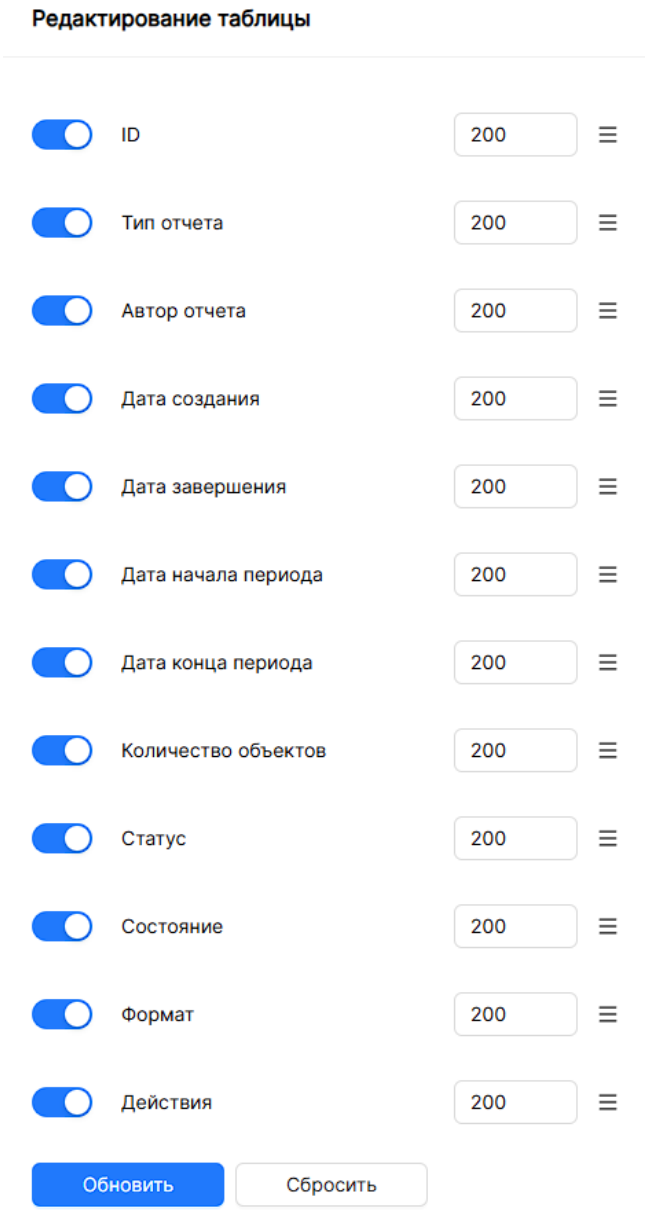


Рис. 36. Окно редактирования таблицы заданий

Задание

1. Изучить модуль построения отчетов.
2. Создать несколько отчетов для аналитики.
3. Проанализировать созданные отчеты.
4. Сопоставить данные, полученные в отчете с данными на графиках.
5. Сделать выводы по результатам работы.

Алгоритм выполнения лабораторной работы

Форма создания отчета состоит из трех шагов. На первом шаге выбираются основные параметры отчета, на втором производится конструирование полей для универсальных отчетов, на третьем – выбираются объекты, которые будут представлены в отчете.

Первый шаг формы отчета состоит из следующих полей (рис. 37):

1. Индикатор текущего шага;
2. Выбор периодичности построения;
3. Выбор регулярности. Для разовой периодичности построения регулярность не задается;
4. Выбор даты начала (только для периодических отчетов);
5. Тип отчета (из набора активированных в системе);
6. Доступные форматы отчета (зависят от типа отчета);
7. Период построения отчета (из набора предустановленных);
8. Дата начала периода построения (если предустановленный период не выбран);
9. Дата конца периода построения (если предустановленный период не выбран).

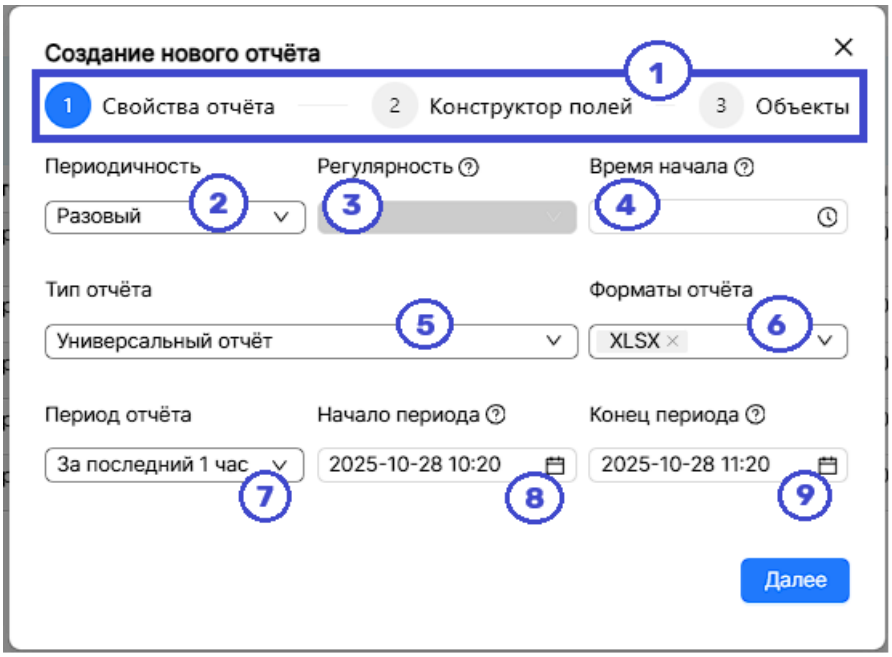


Рис. 37. Форма создания нового отчета

Второй шаг появляется только для одного типа отчета – «Универсального» и предназначен для конструирования его полей (рис. 38). В зависимости от выбранного класса для каждого объекта задается различный набор атрибутов. На рис. 38 показан набор атрибутов для выгрузки отчета по классу «Мониторинг CPE».

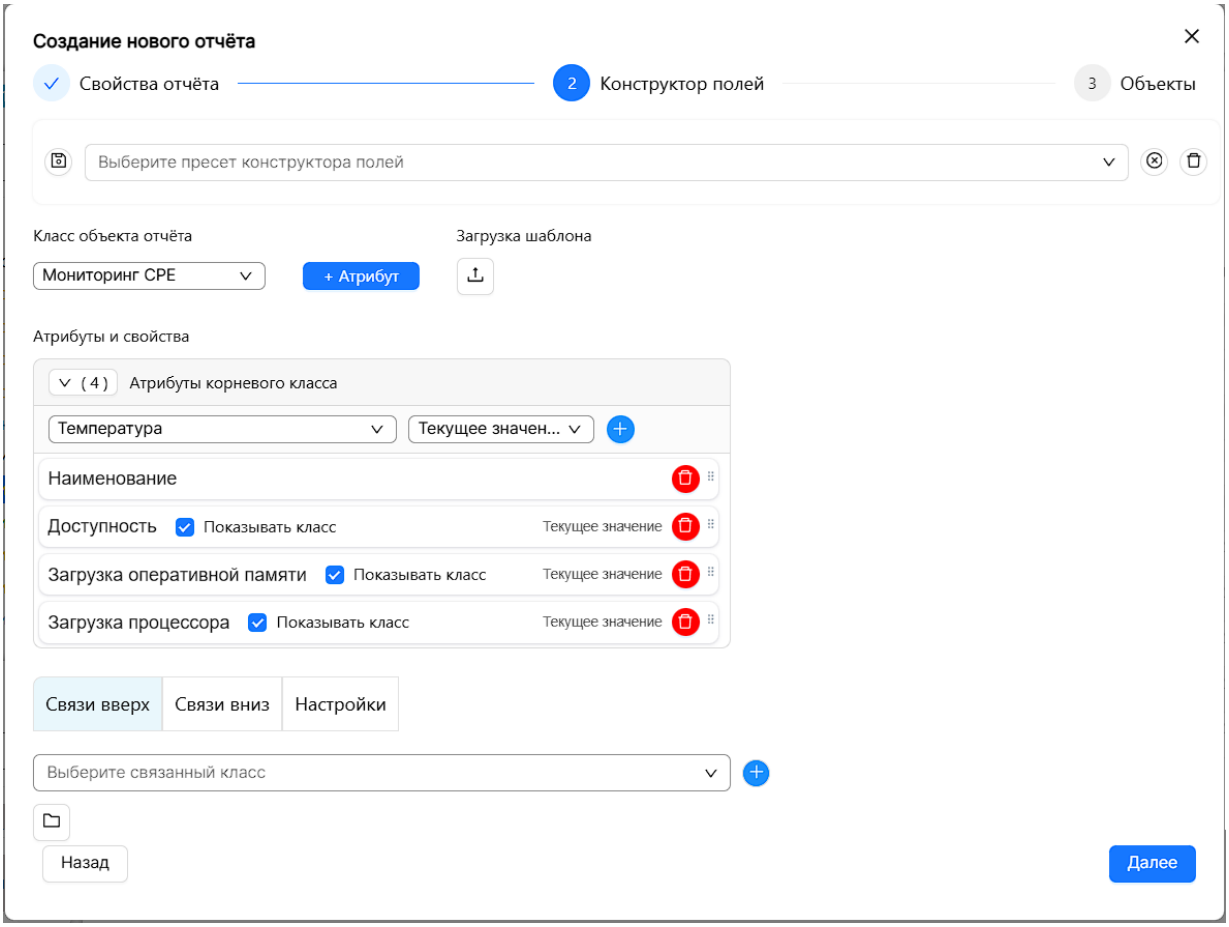


Рис. 38. Конструктор полей

Третий шаг предназначен для выбора объектов, которые будут отражены в отчете (рис. 39).

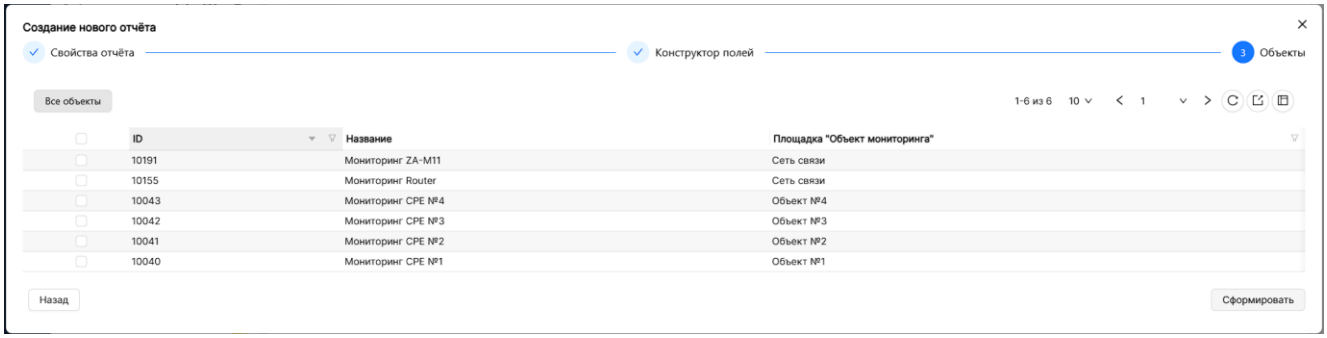


Рис. 39. Выбор объектов для отчета

Содержание отчета

1. Титульный лист.
2. Цель работы.
3. Теоретические сведения.
4. Описание выполнения работы (по шагам).
5. Полученные результаты.
6. Анализ и выводы.

Контрольные вопросы

1. Зачем в Системе КМУТ используются отчеты?
2. В каких форматах могут быть выгружены отчеты?
3. В каком виде представлен модуль построения отчетов?
4. Из каких этапов состоит форма построения отчета?
5. Какие поля используются в первом этапе построения отчета?
6. Можно ли выбрать интервал построения отчета?
7. Какие интервалы доступны для выбора?

7.1.6. Изучение механизма создания инцидентов.

Ключевые слова

Инциденты, триггеры, обработчики состояний.

Цель работы

Изучить Систему мониторинга КМУТ в части информировании о возможных инцидентах. Выполнить обзор настроек Системы КМУТ для настройки пороговых значений срабатывания триггеров инцидентов.

Важное замечание:
Во избежание рисков сбоя в работе Системы мониторинга КМУТ желательно не выполнять каких-либо действий по изменению функций, а только выполнить визуальное ознакомление с действующими настройками без внесения каких-либо изменений.

Краткие теоретические сведения

Система мониторинга КМУТ предназначена для мониторинга и анализа инцидентов, возникающих в процессе эксплуатации информационных систем.

Основными функциями системы являются:

- измерение параметров качества;
- мониторинг сети передачи данных;
- мониторинг состава трафика;
- мониторинг оборудования по протоколу SNMP.

Триггеры — это сигналы, которые срабатывают при определенных событиях.

Триггеры инцидентов настраиваются на основе определённых критериев. Администратор системы имеет возможность выбирать самостоятельно пороги и метрики, по которым будут создаваться инциденты. Настроенные пороговые значения позволяют своевременно реагировать на возникающие проблемы и минимизировать их последствия.

Триггеры могут быть настроены на различные параметры, включая загрузку процессора, использование памяти, температуру, потери пакетов, отклик сети и другие показатели.

К базовым метрикам относятся:

- CPU Utilization – Загрузка процессора
- Memory utilization – Использование памяти
- Temperature in/out – Температура (вход/выход)
- Power (статус БП) – Питание (статус блока питания)
- FAN Status (статус вентиляторов) – Статус вентиляторов
- Uptime – Время работы без перезагрузки
- ICMP availability – Доступность по ICMP
- ICMP response – Отклик ICMP
- ICMP loss – Потери ICMP
- Quality measurement – Измерение качества соединения
- Bandwidth capacity – Пропускная способность сети

В Системе КМУТ управление создания инцидентов осуществляется в подсистеме «Конструктор» в разделе «Автоматы».

1. На главной странице раздела пользователю предоставляется список из уже созданных обработчиков состояний, и классов, к которым привязаны эти обработчики (рис. 40).

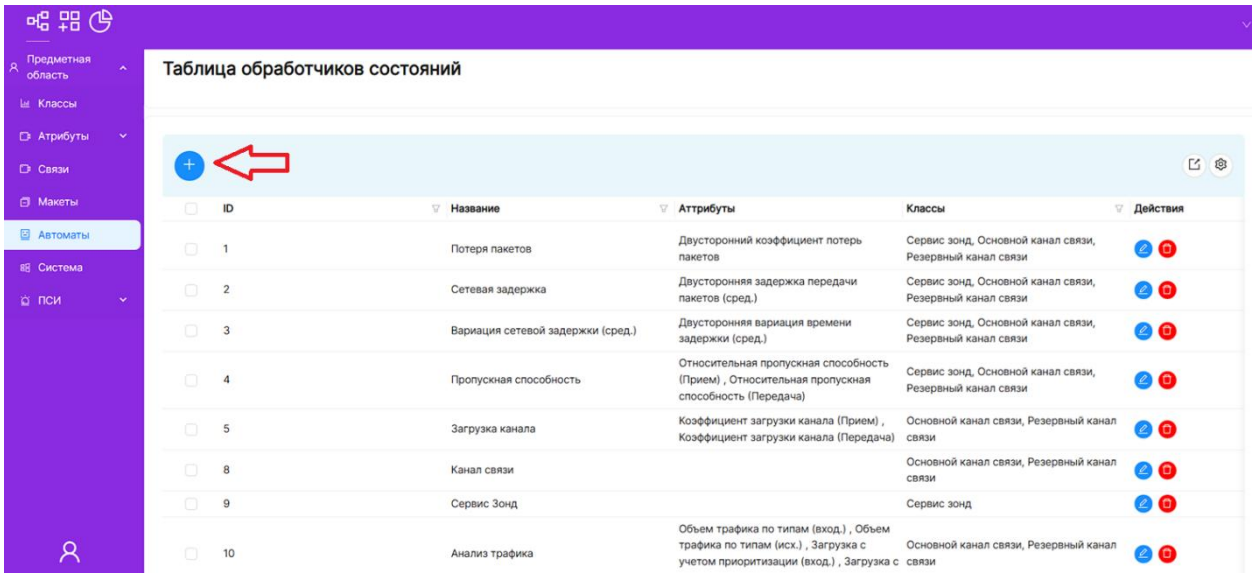


Рис. 40. Таблица обработчиков состояний

2. При нажатии на кнопку «Добавить» откроется модальное окно, в котором пользователь может создать новый обработчик состояний (задать имя, добавить классы для этого обработчика), а также указать атрибуты (триггер), по которому будет создаваться событие (рис. 41).

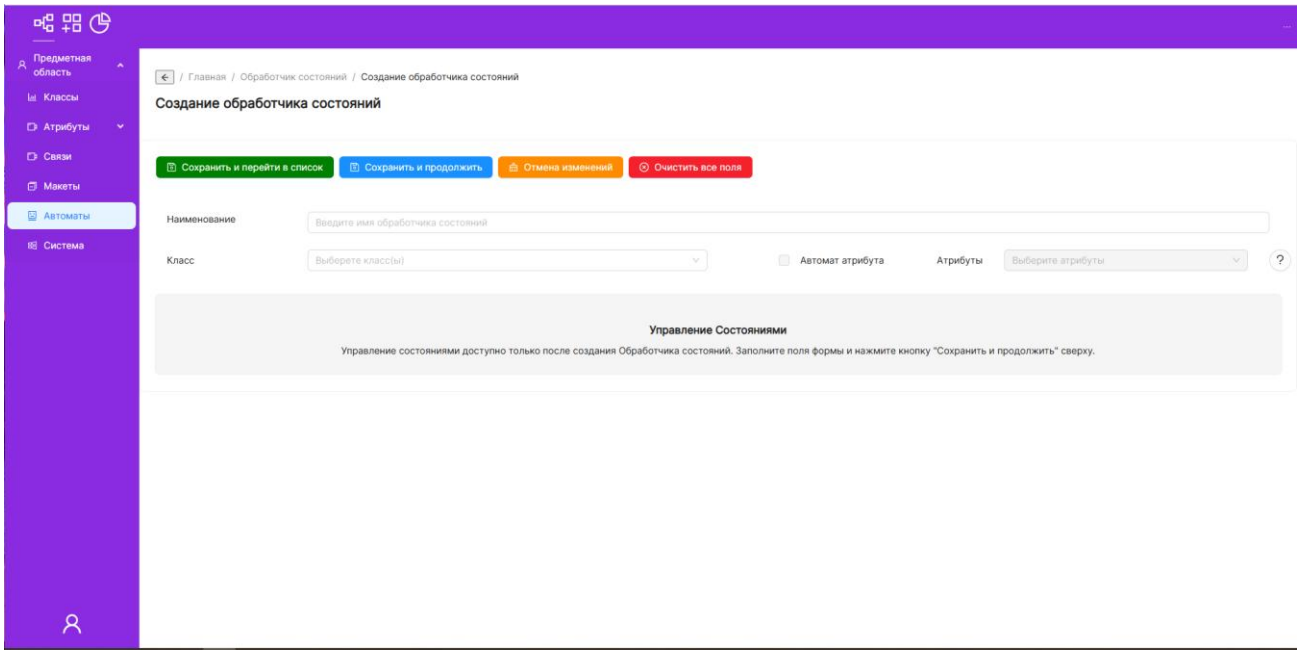


Рис. 41. Создание обработчика состояний

Примечание: после выбора класса (классов) необходимо установить галочку в чекбоксе «Автомат атрибута» для разблокировки окна «Выбора атрибутов». В данном окне отобразятся только те атрибуты, которые привязаны к выбранным классам.

После заполнения всей формы обязательно нажать «Сохранить и перейти в список» (рис. 42).

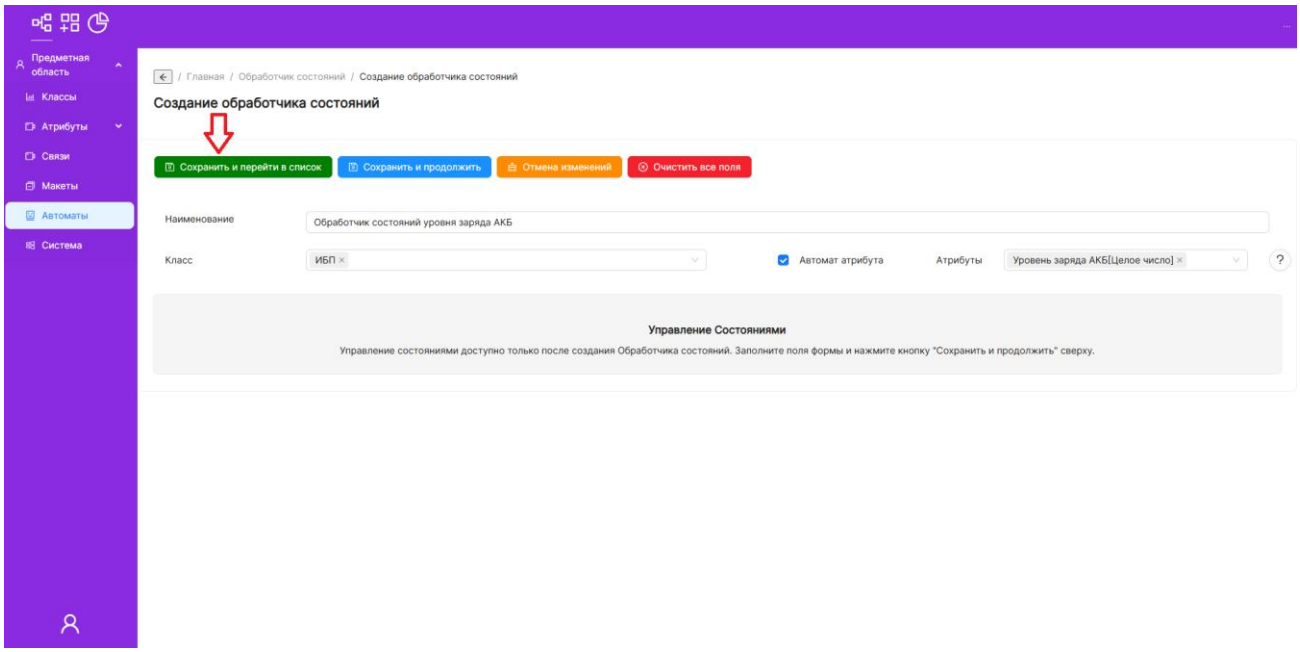


Рис. 42. Сохранение изменений

Получить уведомление об успешном создании обработчика (рис. 43).

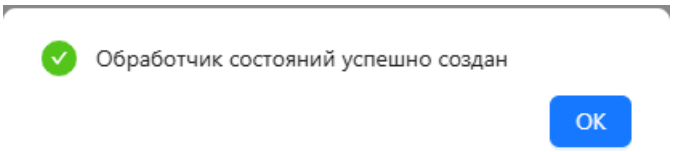


Рис. 43. Уведомление об успешном создании обработчика

Примечание: в случае, если выбранный атрибут (триггер) уже заведен в системе, вместо уведомления об успешном создании, будет ошибка с указанием атрибута (рис. 44).

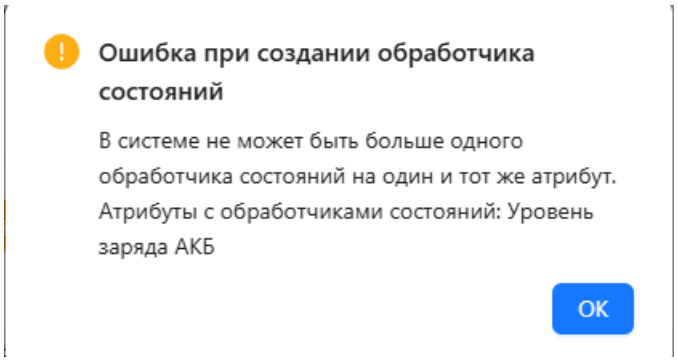


Рис. 44. Ошибка с указанием атрибута

3. После создания обработчика необходимо добавить состояния:

Добавить секцию→Добавить состояние.

Состояния в Системе КМУТ классифицируются по определенному принципу:

- Нет измерений (измерение на объекте только заведено и по нему еще не пришло ни одной положительной точки). Для данного типа состояния выбирается тип «начальное состояние» и самый высокий приоритет по умолчанию 99 (рис. 45).

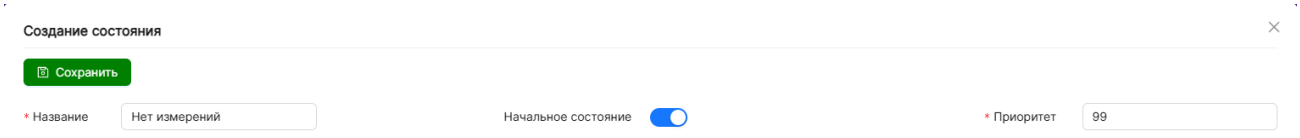


Рис. 45. Создание состояние «Нет измерений»

Далее у состояния указывается вид отображения, в котором пользователь выбирает цвета, в котором будет отображаться измерение (рис. 46).

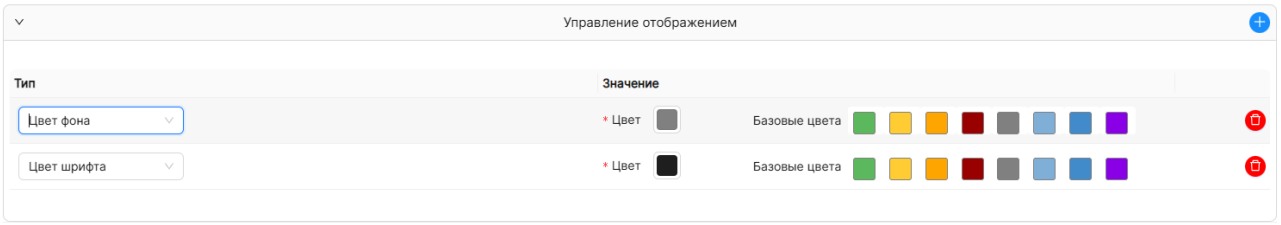


Рис. 46. Выбор цвета

Затем настраивается блок «Управление правилами» в котором пользователю предоставляется выбор из логических операторов И/ИЛИ (рис. 47).

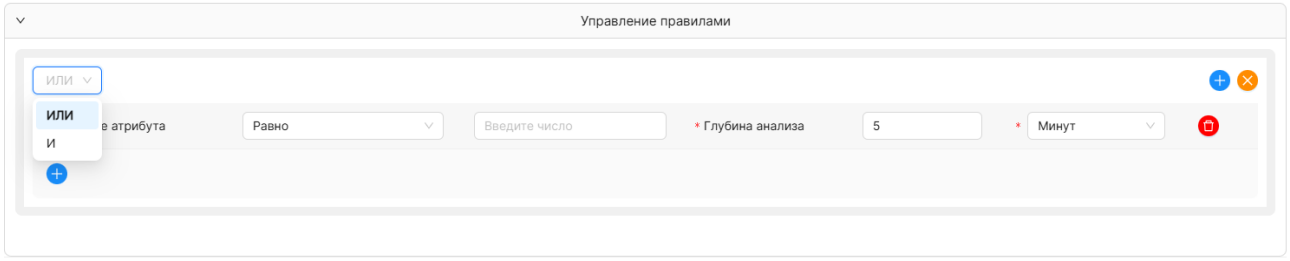


Рис. 47. Блок «Управление правилами»

Также указываются значения и логическая операция с ними, глубина анализа, которая может быть, как во временном интервале, так и в количестве значений, приходящих на график (точек) (рис. 48, 49).

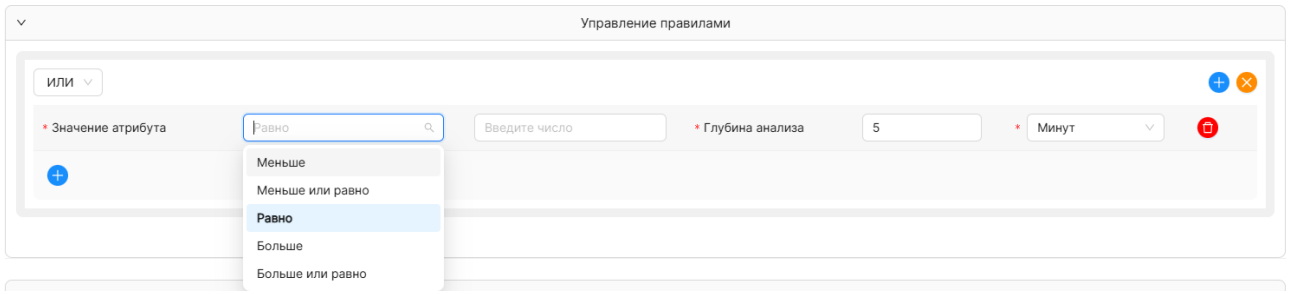


Рис. 48. Выбор значения атрибута

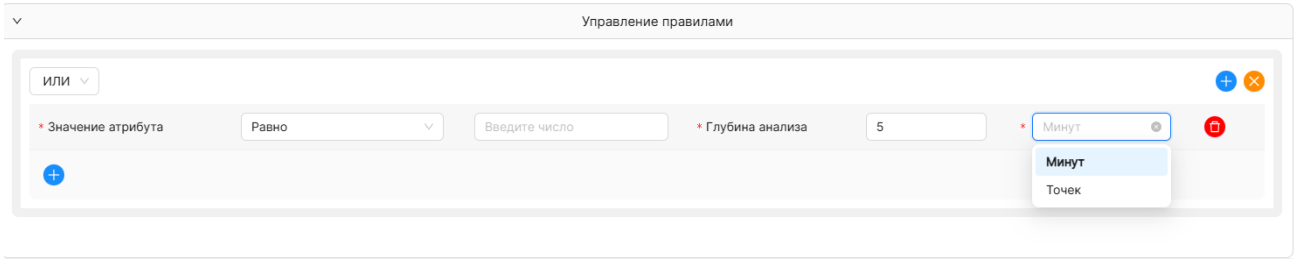


Рис. 49. Выбор глубины анализа

Блок управления эффектами. Данный блок отвечает за создание инцидентов на основе триггеров (порогов, правил), которые указаны в блоке «Управление правилами». Для создания инцидента выбирается тип эффекта: входит в состояние (что означает отработку одного из правил) тип действия: добавить сущность (создание инцидента) название таблицы: инцидент (рис. 50).

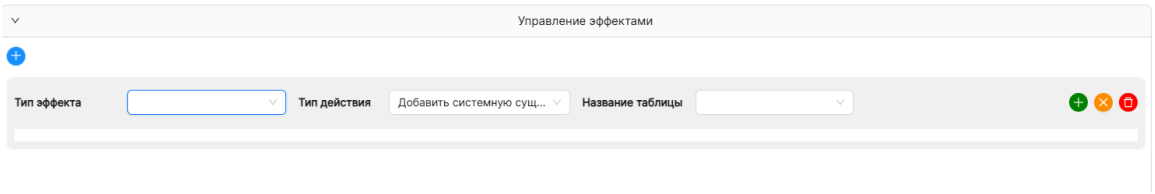


Рис. 50. Блок управления эффектами

Остальные состояния: Профилактика, Недоступность, Норма – настраиваются аналогичным образом. Приоритет идет от большего к меньшему.

Созданные инциденты отображаются в интерфейсе «Витрина» на странице «Инциденты».

Задание

1. Изучить функциональные возможности Системы КМУТ по выявлению и настройке инцидентов.
2. Создать новый триггер с пороговым значением
3. Провести тестирование триггера
4. Проверить создание инцидента в системе мониторинга.
5. Сделать выводы по результатам работы.

Алгоритм выполнения лабораторной работы

1. Войти в Систему мониторинга КМУТ и изучить интерфейс управления инцидентами.
2. Ознакомиться с существующими триггерами и их настройками.
3. Создать новый триггер с пороговым значением, соответствующим определённому инциденту.
4. Провести тестирование триггера, искусственно создавая условия для его срабатывания. (Например: временно отключить зонд и получить инцидент по недоступности).
5. Проанализировать созданные отчёты и статистику инцидентов.
6. Подготовить отчет по выполненной работе.

Содержание отчета

1. Титульный лист.
2. Цель работы.
3. Теоретические сведения.
4. Описание выполнения работы (по шагам).
5. Полученные результаты (настройки триггеров, скриншоты, отчёты).
6. Анализ и выводы.

Контрольные вопросы

1. Какие основные функции выполняет Система мониторинга КМУТ?

2. Что такое триггер инцидента и для чего он используется?
3. Как настроить пороговое значение срабатывания триггера?
4. Какие параметры можно использовать при настройке триггера?
5. Какие виды отчетности и аналитики предоставляет система?
6. Какие меры можно предпринять для уменьшения количества инцидентов?

7.1.7. Изучение команд управления Зондом мониторинга.

Ключевые слова

Зонд мониторинга, SSH, CLI, расписание выполнения измерений.

Цель работы

Изучить команды управления Зондом мониторинга, доступа по протоколу SSH, настройки параметров функционирования.

Краткие теоретические сведения

Зонды мониторинга Системы КМУТ устанавливаются на оконечных объектах мониторинга и предназначены для взаимодействия с измерительным сервером в процессе проведения измерений, а также могут использоваться для сбора данных с сетевых устройств объекта. Основными функциями Зондов являются:

– содействие при проведении измерений;
– мониторинг состояния собственного электропитания;
– дополнительные возможности проверки маршрутизации.

В Системе КМУТ частота проведения измерений задается в интерфейсе «Менеджер» Системы мониторинга КМУТ в разделе «Измерительная система» (рис. 51) и выборе объекта измерения.

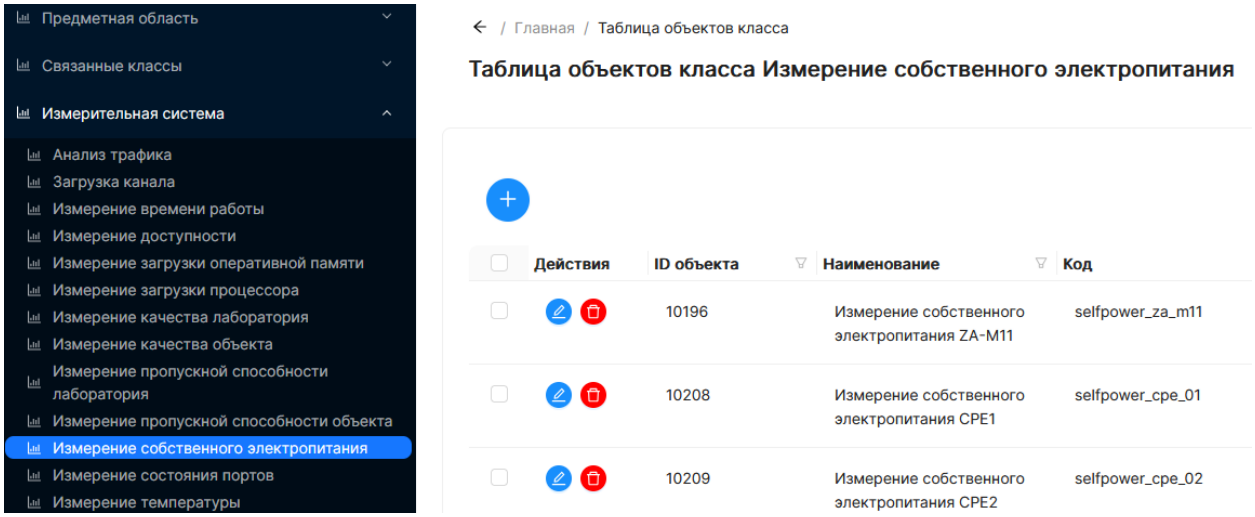


Рис. 51. Раздел «Измерительная система»

Пользователю доступны все возможные модели проведения измерений. Но, в отличие от остальных, расписание измерения электропитания хранится непосредственно на оконечном Зонде КМУТ.

Частоту измерений можно задать как для любого конкретного измерения, так и изменить в шаблоне, что автоматически

применится ко всем измерениям данного типа. Указать можно как временные интервалы, через которые будут совершаться измерения, так и дни недели, в которые надо проводить замеры. Пример задания расписания есть на рис. 52.

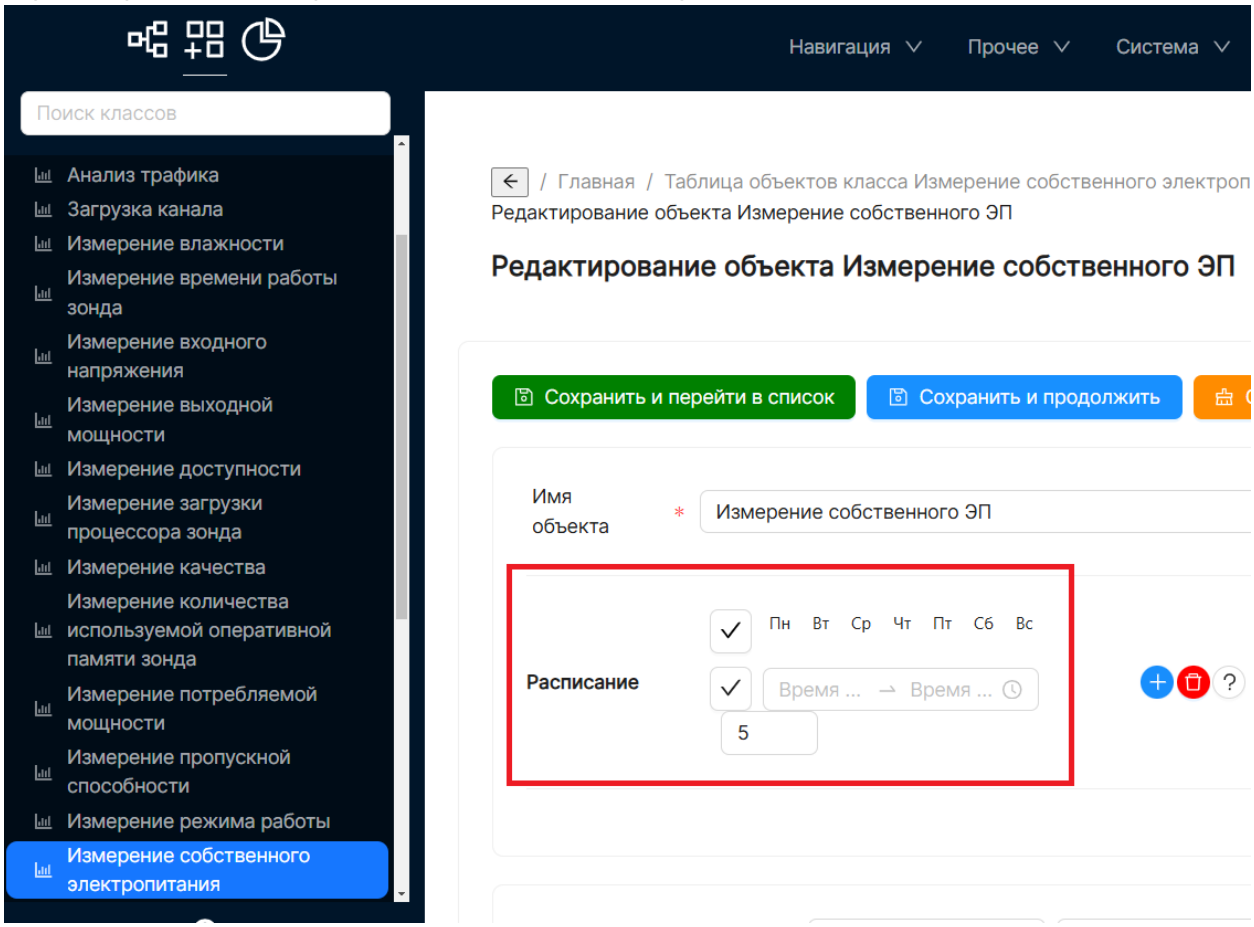


Рис. 52. Задание расписания в Системе мониторинга

Доступ на зонды осуществляется по протоколу SSH.

SSH (Secure Shell) — это протокол для безопасного удаленного доступа к серверам и управления ими через зашифрованное соединение.

Подключение к серверу через SSH

ssh username@server_ip # подключение под пользователем username

ssh -p 2222 username@ server_ip # если сервер слушает порт 2222

Зонды мониторинга Системы КМУТ по умолчанию прослушивают порт 8022.

После подключения вы попадаете в CLI.

CLI (Command Line Interface) — это текстовый интерфейс для взаимодействия с операционной системой. Вместо использования графического интерфейса (кнопок, меню и окон) вы вводите

команды в текстовом виде, чтобы выполнять различные задачи: управление файлами, настройка системы, запуск программ и многое другое.

Основные понятия

1. Терминал — программа, которая позволяет вводить команды и видеть их результат.

2. Команда — инструкция, которую вы вводите для выполнения определенной задачи.

3. Аргументы — дополнительные параметры, которые уточняют команду.

4. Опции (флаги) — специальные параметры, которые изменяют поведение команды (обычно начинаются с - или --).

Основные команды

Рассмотрим список базовых команд, которые используются для работы с Зондом КМУТ:

1. Навигация по файловой системе

- pwd (Print Working Directory) — показывает текущую директорию.
- ls (List) — список файлов и папок в текущей директории.
  - ls -l — подробный список.
  - ls -a — показать скрытые файлы.
- cd (Change Directory) — перейти в другую директорию.
  - cd .. — перейти на уровень выше.
  - cd ~ — перейти в домашнюю директорию.

2. Работа с файлами и папками

- mkdir (Make Directory) — создать папку.
  Пример: mkdir my_folder.
- touch — создать пустой файл.
  Пример: touch file.txt.
- cp (Copy) — скопировать файл или папку.
  Пример: cp file.txt backup/.
- mv (Move) — переместить или переименовать файл/папку.
  Пример: mv file.txt new_name.txt.
- rm (Remove) — удалить файл.
  Пример: rm file.txt.
  - rm -r — удалить папку и её содержимое.

3. Просмотр и редактирование файлов

- cat — вывести содержимое файла.

Пример: cat file.txt.

- vi— текстовый редактор для редактирования файлов. После открытия файла для перехода в режим редактирования необходимо нажать клавишу “i”, по завершении редактирования нажать “Esc”. Чтоб записать изменения, необходимо ввести “:w” , для выхода их файла без сохранения “:q”, чтоб выйти и сохранить изменения “:wq!”

- Пример: vi file.txt.

4. Поиск

- find — поиск файлов и папок.

- Пример: find . -name "file.txt".

- grep — поиск текста внутри файлов.

- Пример: grep "hello" file.txt.

5. Системные команды

- su - – переход в привилегированный режим.

- ps — список запущенных процессов.

- kill — завершить процесс.

- Пример: kill 1234 (где 1234 — ID процесса).

- top — мониторинг системы.

- date – показать текущие дату и время

- inventory – показать общие данные о версии ПО, ОС , типе процессора и текущий аптайм.

6. Сеть

- ping — проверить доступность хоста.

- Пример: ping 1.1.1.1

- traceroute – проверить доступность хоста с указанием промежуточных узлов

- Пример: traceroute 1.1.1.1

-ip a – проверить текущую конфигурацию портов

-ip r – проверить действующую таблицу маршрутизации

Все основные настройки Зонда мониторинга расположены в файле **/etc/kmut/main.conf.** На разных Зондах мониторинга структура данного файла может незначительно отличаться.

Структура файла следующая:

common[mycode] = <код объекта>

common[cso-url] = http://<IP адрес измерительного сервера>

common[proxy] = http://<IP адрес прокси-сервера>

time[timeserver]= <IP сервера синхронизации времени>

qual_bind_address = <IP адрес, с которого можно проводить диагностику (по умолчанию 0.0.0.0 - любой)>

qual_port = <Порт, использующийся для диагностики (по умолчанию 2000)>

ftp_proxy_dst_ip = IP устройства метрологической поверки

ftp_proxy_dst_port = Порт устройства метрологической поверки

ftp_proxy_dst_dataport = Порт для передачи данных при метрологической поверке

ftp_proxy_RemoteFtpCtrlPort = Удаленный порт управления

После внесения изменений в файл и его сохранения, необходимо выполнить команду в режиме суперпользователя, перезапускающую все основные службы:

```
/etc/init.d/kmut-ftp-proxy restart
```

При успешном перезапуске результат выполнения команды будет следующим (на примере Зонда мониторинга CPE1-M5):

```
user@CPE1-M5:~$ su -
Password:
root@CPE1-M5:~# /etc/init.d/kmut-ftp-proxy restart
Restarting kmut-ftp-proxy (via systemctl): kmut-ftp-proxy.service.
root@CPE1-M5:~#
```

Расписание измерения собственного электропитания, в отличие от остальных измерений, хранится непосредственно на самом Зонде мониторинга. При проблемах с измерением собственного электропитания необходимо проверить на Зонде мониторинга, что расписание получено, а настройки в файле main.conf верные

Расписание расположено в /var/cache/kmut/confs/ файл sched.file

Пример проверки:

```
# ls -la /var/cache/kmut/
                    drwxr-xr-x    2 root     0             1056 Dec  6 08:34 .
                    drwxr-xr-x    3 root     0                0 Jan  1  1970 ..
                    -rw-r--r--    1 root     0               10 Dec  6 08:33 kmut-getconf.file
                    -rw-r--r--    1 root     0               10 Dec  6 08:33 kmut-selfpower-
beats
                    -rw-r--r--    1 root     0                0 Dec  6 08:33 kmut-selfpower-
upt
                    -rw-r--r--    1 root     0                2 Dec  6 08:36 last-http
                    -rw-r--r--    1 root     0                2 Dec  6 08:36 last-ping
                    -rw-r--r--    1 root     0               21 Dec  6 08:36 last-selfpower
                    -rw-r--r--    1 root     0                2 Dec  6 08:36 last-snmp
                    -rw-r--r--    1 root     0               92 Dec  6 08:34 push-data-file
                    -rw-r--r--    1 root     0                1 Dec  6 08:36 res.file.tmp
                    -rw-r--r--    1 root     0                2 Dec  6 08:36 res.file.transit
                    -rw-r--r--    1 root     0              886 Oct 25 09:27 sched.file
                    -rw-r--r--    1 root     0               32 Oct 25 09:27 sched.file.md5
```

Для создания расписания необходимо выполнить следующие команды:

/usr/share/kmut/kmut-getconf -v --show-cmd --show-res --show-url -d -f -q – отправляет запрос данных по измерению на Сервер Системы КМУТ, дополнительные параметры --show-cmd --show-res --show-url покажут подробно, корректно ли работает запрос, к какому адресу обращается, и прочую информацию, помогающую понять текущие настроенные параметры измерений.

/usr/share/kmut/kmut-pooler-selfpower – производит измерение на текущий момент. При успешном результате команда не отобразит вывод, а через некоторое время точка измерения отобразится в соответствующем графике Системы мониторинга.

Для облегчения процесса навигации можно использовать следующие рекомендации:

1. Используйте Tab для автодополнения команд и путей.
2. Используйте Ctrl+C для прерывания текущей команды.
3. Используйте Ctrl+L для очистки экрана.
4. Изучите справку по командам с помощью man (например, man ls).

Задание

Изучить функциональные возможности Зонда КМУТ по управлению и мониторингу;

Изучить методы проверки и изменения расписаний измерений и провести тестовые настройки измерений;

Проверить корректировку настроек Системы мониторинга;

Сделать выводы по результатам работы.

Алгоритм выполнения лабораторной работы

1. На АРМ откройте браузер и перейдите по адресу https://10.19.47.252 (согласно п.4.1.);
2. Зайдите в модуль измерения собственного электропитания;
3. Проверьте и установите значение частоты измерения собственного электропитания;
4. Откройте терминал;
5. Зайдите по протоколу SSH на Зонд агрегации ZA-M11 и перейдите в режим администратора (суперпользователя);
6. Проверьте текущую дату, версию ПО, доступность измерительного сервера с Зонда КМУТ утилитой ping;
7. Выведите содержимое файла: cat /etc/kmut/main.conf;
8. Откройте файл для редактирования vi /etc/kmut/main.conf;
9. Проставьте корректные поля common[mycode], common[cso-url] и сохраните все изменения;
10. Перезапустите все службы командой /etc/init.d/kmut-ftp-proxy restart;
11. Запросите актуальное расписание командой /usr/share/kmut/kmut-getconf -v --show-cmd --show-res --show-url -d -f -q;
12. Проверьте наличие расписания на Зонде агрегации командой ls -al /var/cache/kmut/;
13. Дождитесь появления измерительных точек на графике.

Содержание отчета

1. Титульный лист
2. Цель работы
3. Теоретические сведения
4. Описание выполнения работы (по шагам)
5. Полученные результаты (настройки расписания измерений, Зонда КМУТ, скриншоты)
6. Анализ и выводы

Контрольные вопросы

1. Какие основные функции выполняет Зонд КМУТ?
2. Что такое CLI и для чего он используется?
3. Как настроить расписание проведения измерений?

4. Какие параметры нужно указать при настройке Зонда КМУТ в файле main.conf?

5. Как проверить связность между сервером и Зондом КМУТ (доступность)?

6. Как можно запросить актуальное расписание для Зонда КМУТ?

7.1.8. Изучение команд управления Зондом агрегации.

Ключевые слова

Зонд агрегации, SSH, CLI, расписание выполнения измерений.

Цель работы

Изучить команды управления Зондом агрегации, доступа по протоколу SSH, настройки параметров функционирования.

Краткие теоретические сведения

Зонд агрегации – сетевое оборудование, которое объединяет и управляет трафиком от нескольких конечных устройств и передает его дальше на центральный сервер управления.

Подключение к Зонду агрегации происходит через интерфейс командной строки (CLI), позволяющий управлять устройством с помощью команд. Пользователь заходит в систему и работает со строкой структурированных ключевых слов. Формат ключевых слов (команд) — это синтаксис. Язык программирования, используемый в CLI, называют языком оболочки. Операционная система Windows использует PowerShell, а Linux и macOS применяют Bash (Bourne Again Shell) и Zsh.

Подключение к CLI можно выполнить через стандартные порты VGA и клавиатуру, через консольный порт или с помощью протокола SSH.

Удаленное подключения и управление Зондом агрегации осуществляется с помощью сетевого протокола прикладного уровня – SSH (Secure SHell - защищенная оболочка), предназначенный для безопасного удаленного доступа к UNIX-системам. По умолчанию, используется 22-й порт, но в целях безопасности порт лучше сменить на отличный от порта по умолчанию. Для Зондов мониторинга и Зонда агрегации Системы КМУТ используется порт 8022. SSH команда состоит из трех отдельных частей:

ssh {user}@{host}

{user} – указание учетной записи, к которой хотите подключится.

{host} – означает устройство, к которому хотите подключиться (ip или доменное имя).

Базовые команды SSH:

- ls - Показать содержимое каталога (список названий файлов).
- cd - Сменить каталог (перейти в другой).
- mkdir - Создать новую папку (каталог).
- touch - Создать новый файл.
- rm - Удалить файл.
- cat - Показать содержимое файла.
- pwd - Показать текущий каталог (полный путь к этому каталогу).
- cp - Копировать файл/папку.
- mv - Переместить файл/папку.
- grep - Поиск конкретной фразы в файле.
- find - Поиск файлов и папок.
- vi \ nano - Текстовые редакторы.
- history - Показать 50 последних использованных команд.
- clear - Очистить окно терминала.

С Зонда агрегации выполняются измерения метрик до Зондов мониторинга, установленных на объекте, согласно расписанию и параметрам, заданных в шаблоне измерений. Измерения метрик происходит в автоматическом режиме с помощью скрипта пулера. Основными пулерами в системе являются:

- pooler-qual – служит для запуска измерения качества.
- pooler-dost – служит для запуска измерения доступности.
- pooler-band – служит для запуска измерения пропускной способности.
- pooler-util – служит для запуска измерений загрузки канала.
- pooler-snmp – служит для запуска измерений, снимаемых по протоколу SNMP.
- pooler-mtu – служит для запуска измерений MTU.
- pooler-selfpower – служит для запуска измерений электропитания.

Все пулеры можно посмотреть с помощью команды:

cat /usr/share/kmut/

Для запуска скрипта пулера служит команда:

php /usr/share/kmut/kmut-pooler-qual -v --show-cmd --show-res prid=400323 -f -q

где:

- --show-cmd – выводит используемую команду для измерения;
- --show-res – выводит полученный результат измерения;
- --prid – указывается id измерительной пробы;
- -q – не записывает данные в БД и промежуточный буфер (кэш), не создает точку на графике.

В качестве инструментов для проведения ручной диагностики, проверки измерений от Зонда агрегации до объекта служат обычно такие команды как:

- ping – утилита командной строки, которая нужна для проверки подключения к другому сетевому устройству на уровне IP;
- tcpdump – программа-сниффер (анализатор сетевого трафика), которая позволяет перехватывать и анализировать проходящий через утилиту сетевой трафик;
- snmpwalk – утилита используется для чтения всех значений из агента SNMP, указанного в hostname. Snmp-коммьюнити Зонда – public, используемая версия snmp – v2;
- kmut-qual – проприетарная утилита, предназначенная для проверки измерений качества (коэффициент потерь пакетов, вариация времени задержки, время задержки передачи пакетов);
- kmut-band – проприетарная утилита, предназначенная для проверки измерений пропускной способности.

Некоторые из этих утилит будут подробно рассмотрены в других лабораторных работах.

Задание

1. Сменить hostname Зонда агрегации ZA-M11;
2. Узнать какой порт используется для подключения по SSH и сменить его на порт отличный от выставленного;
3. Изучить проприетарные утилиты kmut-band и kmut-qual;
4. Проверить какие пулеры доступны на Зонде агрегации;
5. Научиться запускать пулер kmut-band;
6. Изучить утилиту snmpwalk;
7. Проверить прохождение меток TOS от Зонда до Зонда агрегации;
8. Сделать выводы по результатам работы.

Алгоритм выполнения лабораторной работы

1. Сменить hostname с помощью текстового редактора в файлах /etc/hostname, /etc/hosts и /etc/kmut/main.conf.
2. В файле /etc/ssh/sshd_config посмотреть и сменить порт подключения по SSH.
3. С помощью команды –help просмотреть какие ключи есть у утилит kmut-band и kmut-qual, протестировать данные утилиты.
4. Проверить какие пулеры есть в /usr/share/kmut/, попробовать запустить некоторые из них.
5. Запустить пулер kmut-band до любого Зонда и проанализировать его вывод и сравнить полученный вывод с ручной проверкой команды kmut-band.

Пример запуска на Зонде агрегации ZA-M11 утилиты измерения пропускной способности в ручном режиме от Зонда CH-M11 до Зонда агрегации ZA-M11 (на прием, время измерения 5 секунд, ширина канала 5 Мб/с):

```
root@ZA-M11:~# kmut-band -T5 -R5M 198.18.10.5
    3.0000 MB /   5.03 sec =    4.9988 Mbps 99 %TX 0 %RX 11 retrans 100
KB-cwnd 0.85 msRTT
```

Пример запуска на Зонде агрегации ZA-M11 утилиты измерения пропускной способности в ручном режиме от Зонда агрегации ZA-M11 до Зонда CH-M11 (на передачу, время измерения 5 секунд, ширина канала 5 Мб/с):

```
root@ZA-M11:~# kmut-band -T5 -R5M -r 198.18.10.5
    3.0000 MB /   5.03 sec =    4.9982 Mbps 59 %TX 0 %RX 0 retrans 77
KB-cwnd 0.94 msRTT
```

6. С помощью утилиты snmpwalk узнать загрузку оперативной памяти, CPU, время работы (uptime), Зонда или любого другого оборудования.

Пример запуска на Зонде агрегации ZA-M11 запроса времени работы коммутатора AS1 с помощью утилиты snmpwalk:

```
root@ZA-M11:~# snmpwalk -v2c -c private 10.19.47.21 1.3.6.1.2.1.1.3
iso.3.6.1.2.1.1.3.0 = Timeticks: (34985286) 4 days, 1:10:52.86
```

7. На Зонде запустить пинг с меткой TOS-байт = 32 (или любой другой) и на Зонде агрегации с помощью утилиты tcpdump проанализировать ICMP трафик от Зонда, видно ли отправляемую метку?
8. Подготовить отчет по выполненной работе.

Содержание отчета

1. Титульный лист
2. Цель работы
3. Теоретические сведения
4. Описания выполнения работы (по шагам)
5. Полученные результаты (скриншоты, выводы команд, результаты диагностики)
6. Анализ и выводы

Контрольные вопросы

1. Для чего нужен Зонд агрегации?
2. Что такое CLI и SSH и чем они различаются?
3. С помощью какой команды можно найти нужную строку в файле?
4. С помощью каких команд и инструментов можно узнать коэффициент потерь пакетов UDP трафика?
5. Какие основные ключи используются для команды snmpwalk, чтобы получить все значения по протоколу SNMP на устройстве?

7.1.9. Подключение и добавление Зонда мониторинга в Систему мониторинга КМУТ ML

Ключевые слова

Зонд мониторинга, Система мониторинга КМУТ ML.

Цель работы

Изучить подключение и добавление Зонда мониторинга в Систему мониторинга КМУТ ML.

Краткие теоретические сведения

Зонд мониторинга – это аппаратное или программное устройство/агент, которое устанавливается в ключевых точках сети для сбора, обработки и передачи данных о работе сетевых устройств, серверов, приложений и самом трафике.

Система мониторинга КМУТ ML – проприетарная система мониторинга с закрытым исходным кодом. Входит в реестр отечественного программного обеспечения. Основное назначение – измерение качества каналов связи, состояния оборудования, оповещение о возникающих инцидентах.

В комплекс Системы мониторинга входят:

Сервер КМУТ – ядро системы, которое включает в себя следующие компоненты: хранилище, веб-интерфейс, измерительная подсистема, система создания и оповещения об инцидентах, инвентаризация, отчетность.

Зонд агрегации – метрологически поверенное средство измерения. Используется для проведения измерений и передачи их на сервер КМУТ, необходим для снижения нагрузки на сервере КМУТ.

Зонд мониторинга – метрологически поверенное средство измерения, до него проводятся измерения каналов связи от сервера КМУТ или зонда агрегации.

КМУТ-агент – устанавливается на сервера, АРМ и т.п. для контроля локальных ресурсов и приложений.

Задание

1. Научиться создавать объект Зонд.
2. Узнать необходимые для заполнения атрибуты Зонда.

3. Научиться создавать интерфейс с необходимыми атрибутами и связью с Зондом.

4. Проверить правильность создания и подключения в интерфейсе «Витрина».

Алгоритм выполнения лабораторной работы

1. Перейти в интерфейс «Менеджер» и во вкладке «Связные классы» выбрать «Зонд» (рис. 53);

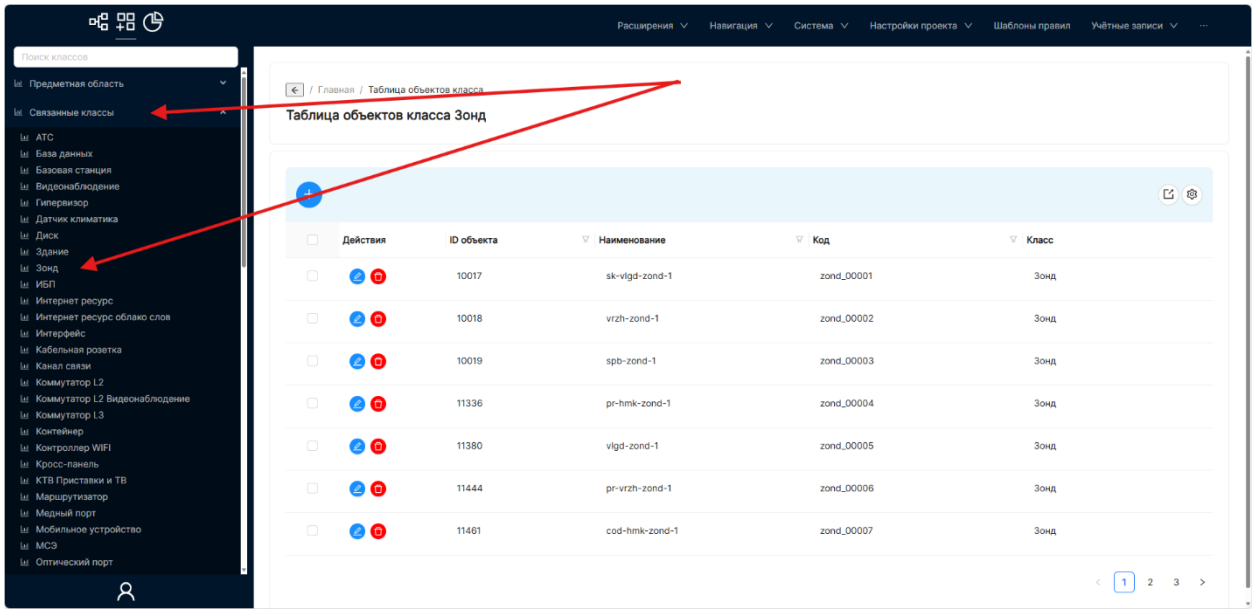


Рис. 53. Интерфейс «Менеджер»

2. Перейти к добавлению Зонда нажатием на кнопку добавления объекта (рис. 54);

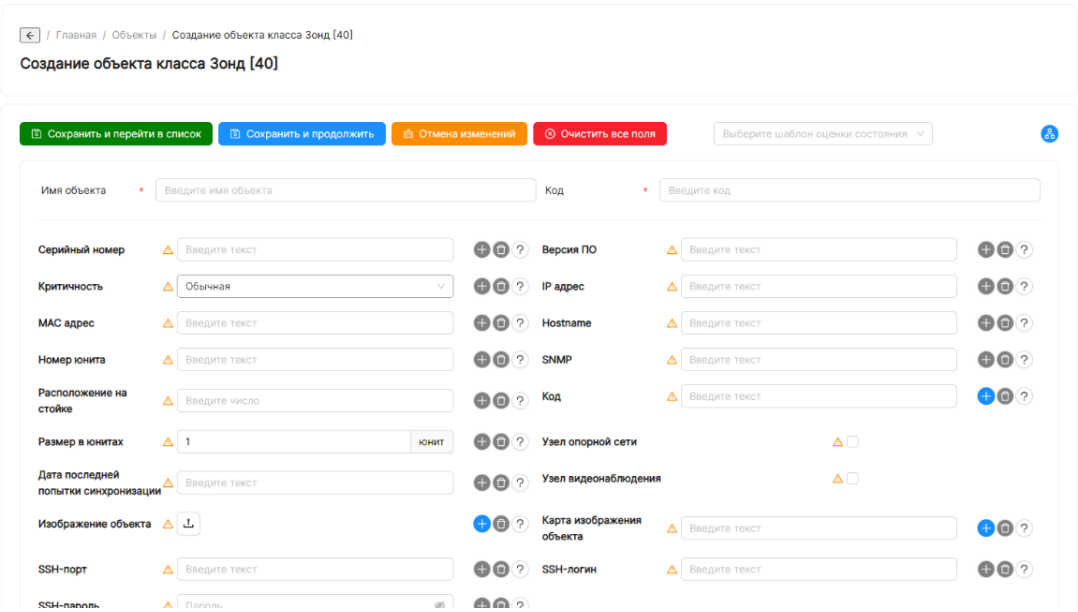


Рис. 54. Создание объекта класса Зонд

3. Заполнить поля «Имя объекта» и «Код объекта» и ввести атрибуты: Серийный номер, Версия ПО, MAC-адрес, Hostname и SNMP;

4. Нажать «Сохранить и продолжить»;

5. В группе «Связи вниз» у устройства создать объект «Интерфейс оборудования с сетевой картой»;

6. У интерфейса заполнить поля «Имя объекта», «Код», «IP адрес» и «Имя интерфейса» (рис. 55). Далее нажать кнопку «Сохранить изменения»;

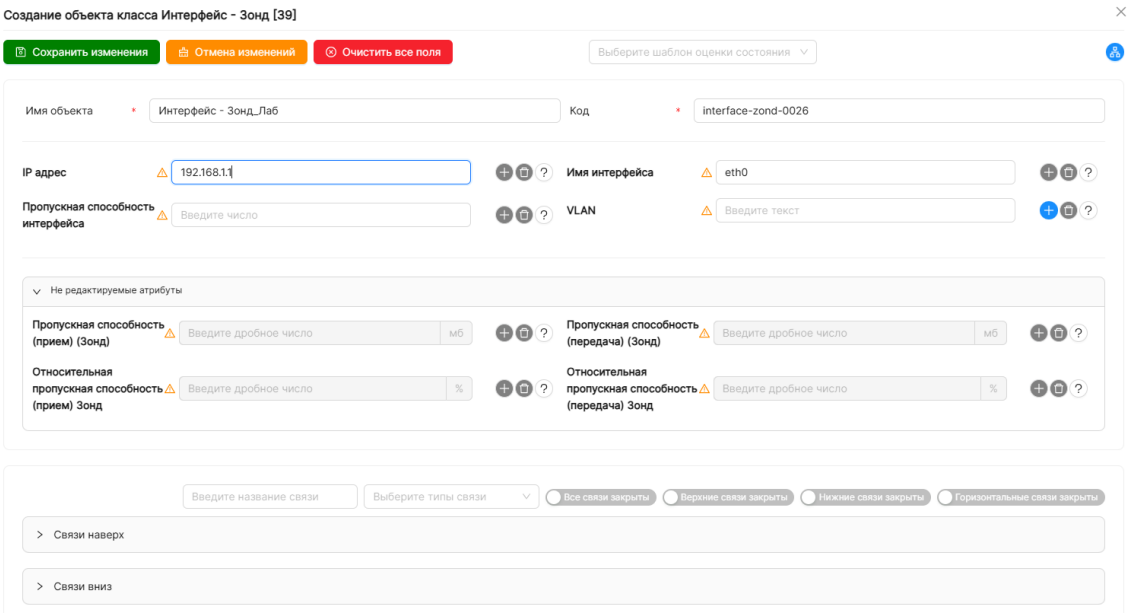


Рис. 55. Создание объекта класса Интерфейс-Зонд

7. Для проверки правильности введённых данных можно перейти в интерфейс «Витрина» во вкладку активы и найти созданный зонд.

Содержание отчета

1. Титульный лист;
2. Цель работы;
3. Теоретические сведения;
4. Описания выполнения работы (по шагам);
5. Полученные результаты (скриншоты);
6. Анализ и выводы.

Контрольные вопросы

1. В каком интерфейсе Системы мониторинга КМУТ ML создаётся Зонд?

2. Какие атрибуты необходимо заполнять при создании Зонда?

3. В какой связи добавляется Интерфейс к Зонду?

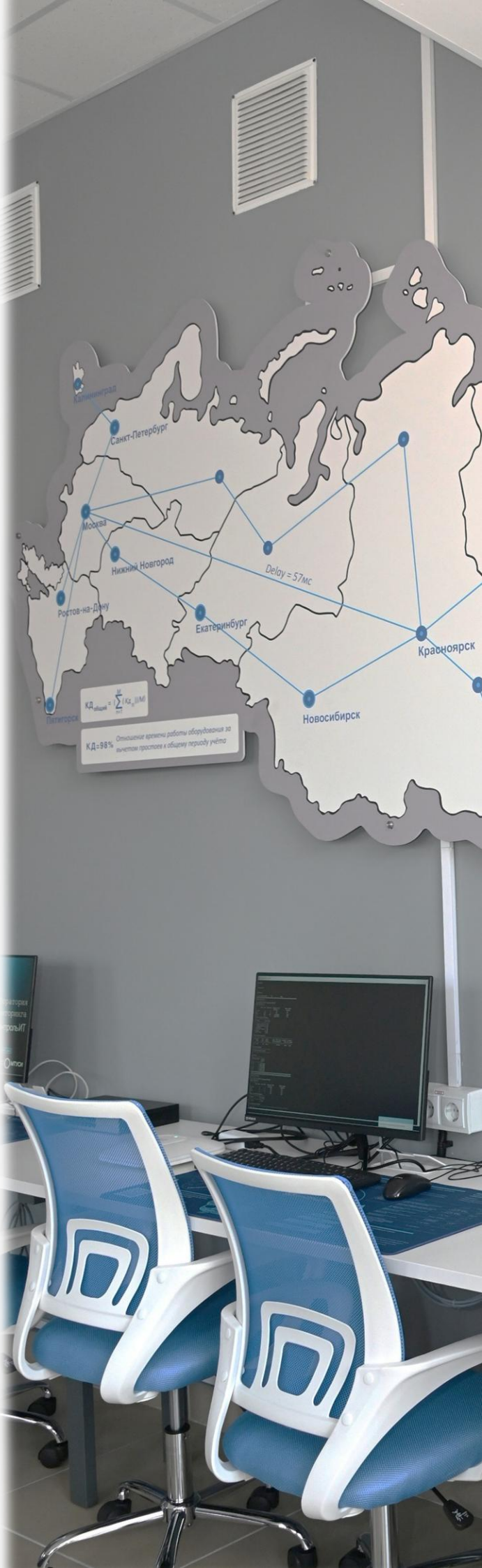
4. Какие атрибуты заполняются при создании Интерфейса Зонда?

5. Где можно проверить правильность создания и подключения Зонда?

7.2. Перечень лабораторных работ по модулю «Технологии коммутации, маршрутизации и диагностики в сетях связи. Сетевые технологии»

В результате выполнения лабораторных работ вы изучите:
основы работы с сетями связи через интерфейс командной строки,
процедуры сброса маршрутизаторов,
управление трафиком с помощью протокола STP,
настройку виртуальных локальных сетей (VLAN) и технологии Q-in-Q,
настройки статической маршрутизации, протокола DHCP и механизма NAT,
работу протокола туннелирования GRE,
управление сетевым трафиком с помощью протоколов BGP и OSPF.

Освоите:
анализ трафика с помощью анализатора сети,
сбор сетевых данных по протоколу SNMP.



7.2.1. Изучение диагностики сетей связи через интерфейс командной строки.

Ключевые слова

CLI, диагностика, troubleshooting, утилиты ping, traceroute, mtr.

Цель работы

Приобрести навыки работы по проведению диагностики сетей связи и уметь использовать специализированные утилиты для диагностики сетей связи через командную строку.

Краткие теоретические сведения

Интерфейс командной строки (CLI)

Помимо Систем мониторинга, для запуска утилит диагностики сетей связи можно также использовать интерфейс командной строки (CLI, Command Line Interface). Пользователь заходит в операционную систему и работает со строкой структурированных ключевых слов. Формат ключевых слов (команд) — это синтаксис. Язык программирования, используемый в CLI, называют языком оболочки.

Стоит отметить, что все команды CLI являются чувствительными к регистру. Прежде чем вводить команду необходимо убедиться, что отключены все функции, которые могут привести к изменению регистра текста.

Средства устранения неполадок (Troubleshooting Tools)

Устранение сетевых неполадок играет решающую роль в обеспечении надежной сети для бесперебойной и эффективной работы современных предприятий и организаций, а также поддержания работоспособности их бизнес-процессов. В современном мире, где надежная цифровая связь и безопасная передача данных являются жизненно важными факторами, даже незначительные сбои в работе сети могут иметь значительные последствия. Результатами таких сбоев может быть снижение производительности, простои, потеря финансов и удовлетворенности клиентов. Таким образом, для поддержания непрерывности работы и минимизации простоев необходимо знать определенные методики и иметь набор инструментов, который

поможет быстро обнаруживать и решать проблемы в сети, не требуя соответствующих навыков сетевого администратора. Большинство сетевых устройств предоставляют множество встроенных команд, которые помогут в мониторинге и устранении неполадок.

Команда ping

Данная команда помогает определять подключение между двумя устройствами в сетевой инфраструктуре.

Эхо-запрос и эхо-ответ, в совокупности называемые **эхо-протоколом**, представляют собой очень простое, но эффективное средство мониторинга сети. Компьютер, маршрутизатор или другое сетевое устройство посылает по составной сети ICMP-сообщение эхо-запроса, указывая в нем IP-адрес узла, достижимость которого нужно проверить. Узел, получивший эхо-запрос, формирует и отправляет эхо-ответ отправителю запроса. Так как эхо-запрос и эхо-ответ передаются по сети внутри IP-пакетов, их успешная доставка указывает на нормальное функционирование всей транспортной системы составной сети.

Формат эхо-запроса и эхо-ответа показан на рис. 56. Поле типа для эхо-ответа равно 0, для эхо-запроса — 8; поле кода всегда равно 0 и для запроса, и для ответа. В байтах 5 и 6 заголовка содержится идентификатор запроса, в байтах 7 и 8 — порядковый номер. В поле данных эхо-запроса может быть помещена произвольная информация, которая в соответствии с данным протоколом должна быть скопирована в поле данных эхо-ответа.

Рис. 56. Формат ICMP-сообщений типа эхо-запрос и эхо-ответ

Поля идентификатора запроса и порядкового номера используются одинаковым образом всеми сообщениями типа

запрос-ответ. Посылая запрос, приложение помещает в эти два поля информацию, предназначенную для последующего встраивания ее в соответствующий ответ. Сообщение-ответ копирует значения этих полей в свои поля того же назначения. Когда ответ возвращается в пункт отправки сообщения-запроса, то на основании идентификатора он может «найти и опознать» приложение, пославшее запрос. Порядковый номер используется приложением, чтобы связать полученный ответ с соответствующим запросом (учитывая, что одно приложение может выдать несколько однотипных запросов).

Утилита ping обычно посылает серию эхо-запросов к тестируемому узлу и предоставляет пользователю статистику об утерянных эхо-ответах и среднем времени реакции сети на запросы. Утилита ping выводит на экран сообщения следующего вида обо всех поступивших ответах.

Пример:

```
# ping mtuci.ru
Pinging mtuci.ru [185.71.67.49] with 64 bytes of data:
Reply from 185.71.67.49: bytes=64 time=256ms TTL= 123
Reply from 185.71.67.49: bytes=64 time=310ms TTL= 123
Reply from 185.71.67.49: bytes=64 time=260ms TTL= 123
Reply from 185.71.67.49: bytes=64 time=146ms TTL= 123
```

Из сообщения видно, что в ответ на тестирующие запросы, посланные узлу mtuci.ru, получено 4 эхо-ответа. Длина каждого сообщения — 64 байта. В следующей колонке помещены значения времени оборота (RTT, round-trip time), то есть времени от момента отправки запроса до получения ответа на запрос. Как видим, сеть работает достаточно нестабильно — время в последней строке отличается от времени во второй более чем в два раза. На экран выводится также оставшееся время жизни поступивших пакетов.

Команда traceroute

Рассмотрим использование ICMP-сообщений об ошибке недостижимости узла в популярной утилите **traceroute**, предназначенной для мониторинга сети. Когда маршрутизатор не может передать или доставить IP-пакет, он отсылает узлу, отправившему этот пакет, сообщение о недостижимости узла назначения.

ICMP-сообщения об ошибках лежат в основе работы популярной утилиты **traceroute** для ОС Unix, имеющей в ОС Windows название **tracert**. Утилита позволяет проследить маршрут до удаленного хоста, определить среднее время оборота (RTT), IP-адрес и в некоторых случаях доменное имя каждого промежуточного маршрутизатора. Эта информация помогает найти маршрутизатор, на котором оборвался путь пакета к удаленному хосту.

Утилита traceroute осуществляет трассировку маршрута, посылая серию обычных IP-пакетов в конечную точку изучаемого маршрута. Идея метода состоит в следующем. Значение времени жизни (TTL) первого отправляемого пакета устанавливается равным 1. Когда протокол IP первого маршрутизатора принимает этот пакет, он в соответствии со своим алгоритмом уменьшает значение TTL на 1 и получает 0. Маршрутизатор отбрасывает пакет с нулевым временем жизни и возвращает узлу-источнику ICMP-сообщение об ошибке истечения времени дейтаграммы (значение поля типа равно 11) вместе с заголовком IP и первыми 8 байтами потерянного пакета. Получив ICMP-сообщение о причине недоставки пакета, утилита traceroute запоминает адрес первого маршрутизатора (который извлекает из заголовка IP-пакета, несущего ICMP-сообщение).

Затем traceroute посылает следующий IP-пакет, но теперь со значением TTL, равным 2. Этот пакет благополучно проходит первый маршрутизатор, но «умирает» на втором, о чем немедленно отправляется аналогичное ICMP-сообщение об ошибке истечения времени дейтаграммы. Утилита traceroute запоминает адрес второго маршрутизатора, и т. д. Такие действия выполняются с каждым маршрутизатором вдоль маршрута вплоть до узла назначения или неисправного маршрутизатора. Рассмотрена работа утилиты traceroute весьма схематично, но и этого достаточно, чтобы оценить изящество идеи, лежащей в основе ее работы. Остальные ICMP-сообщения об ошибках имеют такой же формат и отличаются друг от друга только значениями полей типа и кода.

Далее приведен пример экранной формы, выведенной утилитой traceroute (ОС Linux) при трассировке хоста 8.8.8.8.

Пример:

```
 # traceroute 8.8.8.8
 traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
  1  172.16.1.3 (172.16.1.3)  0.558 ms  0.770 ms  0.930 ms
  2  172.16.13.114 (172.16.13.114)  46.576 ms  46.476 ms  46.193 ms
  3  172.16.13.89 (172.16.13.89)  46.796 ms  46.886 ms  46.802 ms
  4   mag9-cr03-be77.100.msk.mts-internet.net (195.34.36.225)  56.064 ms  56.195 ms
55.635 ms
  5  72.14.223.74 (72.14.223.74)  55.784 ms  56.159 ms 72.14.223.72 (72.14.223.72)
55.250 ms
  6  * * *
  7  192.178.241.148 (192.178.241.148)  22.662 ms 72.14.233.96 (72.14.233.96)  31.771
ms 108.170.225.20 (108.170.225.20)  31.201 ms
  8  192.178.241.234 (192.178.241.234)  31.049 ms * 172.253.66.116 (172.253.66.116)
53.337 ms
  9 108.170.235.64 (108.170.235.64) 43.449 ms 172.253.66.108 (172.253.66.108) 58.028
ms 142.251.238.70 (142.251.238.70)  43.274 ms
 10  142.251.237.144 (142.251.237.144)  53.803 ms 142.250.235.68 (142.250.235.68)
53.655 ms 74.125.253.109 (74.125.253.109)  53.868 ms
 11  216.239.62.107 (216.239.62.107)  53.790 ms * *
 12  * * *
 13  * * *
 14  * * *
 15  * * *
 16  * * *
 17  * * *
 18  * * *
 19  * * *
 20  dns.google (8.8.8.8)  54.121 ms * *
```

Последовательность строк соответствует последовательности маршрутизаторов, образующих маршрут к заданному узлу. Первое число в строке — число хопов (транзитных участков) до соответствующего маршрутизатора. Утилита traceroute тестирует каждый маршрутизатор трижды, поэтому следующие три числа в строке — это значения *RTT*, вычисленные путем посылки трех пакетов, время жизни которых истекло на этом маршрутизаторе. Если ответ от какого-либо маршрутизатора не приходит за заданное время, то вместо времени на экране печатается звездочка (*). Далее идут доменное имя (если оно имеется) и IP-адрес маршрутизатора. Видно, что почти все интерфейсы маршрутизаторов поставщиков услуг Интернета зарегистрированы в службе DNS, а первые два, относящиеся к локальным маршрутизаторам, — нет.

Время, указанное в каждой строке, не отражает время прохождения пакетов между двумя соседними маршрутизаторами — это время, за которое пакет проделывает путь от источника до

соответствующего маршрутизатора и обратно. Так как ситуация в Интернете с загрузкой маршрутизаторов постоянно меняется, время достижимости маршрутизаторов не всегда нарастает монотонно, а может изменяться достаточно произвольным образом.

Команда mtr

Команда mtr — это простой, кроссплатформенный инструмент сетевой диагностики с помощью командной строки, который объединяет функциональность широко используемых программ traceroute и ping в один инструмент. Подобным образом, как traceroute, команда mtr выводит информацию о маршруте, показывает список маршрутизаторов, через которые проходит пакет. Команда mtr показывает больше информации, чем traceroute: она определяет путь к удаленному компьютеру при выводе процента отклика, а также времени отклика всех сетевых переходов в интернет-маршруте между локальной системой и удаленными машинами.

После запуска mtr проверяет сетевое соединение между локальной системой и указанным вами удаленным хостом. Сначала mtr устанавливает адрес каждого сетевого перехода (мосты, маршрутизаторы, шлюзы и т.д.) между хостами, затем он пингует (отправляет запросы ICMP ECHO) каждому из них, чтобы определить качество отклика на каждом сетевом устройстве.

Команда mtr имеет несколько опций.

Для отображения IP-адреса вместо имён хостов используется флаг -n:

mtr -n mtuci.ru

Для ограничения числа пингов конкретным значением и выхода из mtr после этих пингов можно использовать флаг -c. Если наблюдать за столбцом Snt, то вы увидите, что как только указанное количество пингов достигнуто, текущее обновление останавливается и программа завершает проверку.

mtr -c 5 mtuci.ru

Можно установить инструмент сетевой диагностики mtr в режиме отчета с использованием флага -r, что является полезным вариантом для создания статистики качества сети. Возможно использовать эту опцию вместе с опцией -c, чтобы указать

количество пингов. Флаг -w обеспечивает более широкий режим отчета для подробного вывода.

Пример:

```
# mtr -rw -c 5 mtuci.ru
Start: 2025-08-15T09:53:53+0000
HOST: Zond1                                    Loss%   Snt   Last   Avg  Best  Wrst StDev
1.|-- 172.16.1.3                                0.0%     5    0.8   0.8   0.7   0.9   0.1
2.|-- 172.16.13.114                             0.0%     5   42.2  36.2  25.2  42.2   7.0
3.|-- 172.16.13.93                              0.0%     5   49.7  38.6  25.1  51.0  11.3
4.|-- mag9-cr03-be77.100.msk.mts-internet.net    0.0%     5   46.4  44.5  26.8  75.7  19.0
5.|-- mag9-cr02-be13.77.msk.mts-internet.net     0.0%     5   50.5  41.7  31.6  50.5   8.2
6.|-- as20764.asbr.router                        0.0%     5   50.5  47.9  42.1  52.4   4.1
7.|-- ???                                      100.0     5    0.0   0.0   0.0   0.0   0.0
8.|-- STV.msk-m9-cr5.rascom.as20764.net          0.0%     5   57.4  45.6  36.2  57.4   9.7
9.|-- ???                                      100.0     5    0.0   0.0   0.0   0.0   0.0
10.|-- 185.71.67.49                              0.0%     5   49.0  40.7  28.1  49.0   8.0
```

Задание

Выполнить команды по проведению диагностики сетей связи через командную строку и уметь правильно понимать результаты выполнения команд.

Сделать выводы по результатам работы.

Алгоритм выполнения лабораторной работы

1. Из терминальной программы АРМ выполнить вход по протоколу SSH на Зонд агрегации;

Пример выполнения команды (с сервера, ЦСИ):

```
mtusiadmin@kmut-ml-mtusi:~$ ssh -p 8022 user@10.19.47.5
Authorized use only. All activity may be monitored and
reported.
user@10.19.47.5's password:
Linux ZA-M11 6.6.63-rt46-kmut x86_64
----------------------
   Welcome to probe!
----------------------
Last login: Thu Sep  4 12:19:54 2025 from 10.19.47.232
user@ZA-M11:~$
```

2. Выполнить команды ping, traceroute, mtr до двух сетевых устройств, находящихся в подсетях Зонда агрегации (например, до Зондов мониторинга CPE1-M5 или CPE2-M5);

3. Выполнить команды ping, traceroute, mtr до двух российских ресурсов;

Пример выполнения команд:

```
user@ZA-M11:~$ ping 10.10.1.2
PING 10.10.1.2 (10.10.1.2) 56(84) bytes of data.
64 bytes from 10.10.1.2: icmp_seq=1 ttl=63 time=287 ms
--- 10.10.1.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 286.674/286.674/286.674/0.000 ms
user@ZA-M11:~$ traceroute 10.10.1.2
traceroute to 10.10.1.2 (10.10.1.2), 30 hops max, 60 byte packets
 1  198.18.10.2 (198.18.10.2)  281.570 ms  274.625 ms  240.225 ms
 2  10.10.1.2 (10.10.1.2)  259.001 ms  258.939 ms  177.182 ms
user@ZA-M11:~$ mtr 10.10.1.2
```

4. Выполнить команды ping, traceroute, mtr до двух зарубежных ресурсов;

Пример выполнения команд:

```
user@ZA-M11:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=105 time=17.9 ms
^C
--- 8.8.8.8 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 17.920/17.920/17.920/0.000 ms
user@ZA-M11:~$ traceroute 8.8.8.8
user@ZA-M11:~$ mtr 8.8.8.8
```

5. При наличии соответствующих утилит на АРМ выполнить аналогичные команды пункта 2, 3, 4 от имени АРМ;
6. Зафиксировать полученные значения;
7. Сделать анализы и выводы по результатам полученных значений.

Содержание отчета

1. Титульный лист;
2. Цель работы;
3. Теоретические сведения;
4. Описание выполнения работы (по шагам);
5. Полученные результаты (выводы команд, скриншоты);
6. Анализ и выводы.

Контрольные вопросы

1. Зачем нужны инструменты диагностики сетей связи?
2. Поясните назначение команды ping?
3. Поясните назначение команды traceroute?
4. Поясните назначение команды mtr?

7.2.2. Изучение функций анализатора трафика локальной сети.

Ключевые слова

Анализ трафика, сниффер, tcpdump, iftop.

Цель работы

Изучить функции анализатора трафика (сниффера) для анализа трафика в локальной сети, изучение трафика по типам и протоколам.

Краткие теоретические сведения

Описание анализатора трафика

Sniffer (от англ. to sniff – нюхать) – это сетевой анализатор трафика, программа или программно-аппаратное устройство, предназначенное для перехвата и последующего анализа, либо только анализа сетевого трафика, предназначенного для других узлов.

Перехват трафика может осуществляться:

- обычным «прослушиванием» сетевого интерфейса (метод эффективен при использовании в сегменте концентраторов (хабов) вместо коммутаторов (свичей), в противном случае метод малоэффективен, поскольку на сниффер попадают лишь отдельные фреймы);
- подключением сниффера в разрыв канала связи;
- ответвлением (программным или аппаратным) трафика и направлением его копии на сниффер;
- через анализ побочных электромагнитных излучений и восстановление таким образом прослушиваемого трафика;
- через атаку на канальном (2-й) или сетевом (3-й) уровне, приводящую к перенаправлению трафика жертвы или всего трафика сегмента на сниффер с последующим возвращением трафика в надлежащий адрес.

Снифферы применяются как в благих, так и в деструктивных целях.

Анализ прошедшего через сниффер трафика позволяет:

- отслеживать сетевую активность приложений;
- отлаживать протоколы сетевых приложений;
- локализовать неисправность или ошибку конфигурации;

- обнаружить паразитный, вирусный и закольцованный трафик, наличие которого увеличивает нагрузку сетевого оборудования и каналов связи;
- выявить в сети вредоносное и несанкционированное ПО, например, сетевые сканеры, флудеры, троянские программы, клиенты пиринговых сетей и другие;
- перехватить любой незашифрованный (а порой и зашифрованный) пользовательский трафик с целью узнавания паролей и другой информации.

На сегодняшний момент существует достаточно большое количество реализаций снифферов.

Команда tcpdump

tcpdump — это программа-сниффер, которая позволяет перехватывать и анализировать проходящий через утилиту сетевой трафик. Утилита имеет богатый набор опций и различных фильтров, благодаря чему ее можно применять для различных целей. tcpdump — это полностью консольная утилита без графического интерфейса, а значит, ее можно запускать на сетевом оборудовании, где отсутствует поддержка графического интерфейса (GUI).

Утилита tcpdump позволяет проверять заголовки пакетов TCP/IP и выводить одну строку для каждого из пакетов. Она будет делать это до тех пор, пока не нажать на клавиатуре Ctrl + C.

Рассмотрим одну строку из примера вывода:

```
 20:58:26.765637 IP 10.0.0.50.80 > 10.0.0.1.53181: Flags [F.], seq 1, ack 2, win 453, options [nop,nop,TS val 3822939 ecr 249100129], length 0
```

Каждая строка включает:

- Метка времени Unix (20: 58: 26.765637);
- протокол (IP);
- имя или IP-адрес исходного хоста и номер порта (10.0.0.50.80);
- имя хоста или IP-адрес назначения и номер порта (10.0.0.1.53181);
- Флаги TCP (Flags [F.]). Указывают на состояние соединения и могут содержать более одного значения:
  - S — SYN. Первый шаг в установлении соединения;
  - F — FIN. Прекращение соединения;
  - A — ACK. Пакет подтверждения принят успешно;

- P — PUSH. Указывает получателю обрабатывать пакеты вместо их буферизации;
- R — RST. Связь прервалась;

- Порядковый номер данных в пакете. (seq 1);
- Номер подтверждения. (ack 2);
- Размер окна (win 453). Количество байтов, доступных в приемном буфере. Далее следуют параметры TCP;
- Длина полезной нагрузки данных (length 0).

Важно заметить, что утилита tcpdump (как и все анализаторы пакетов) при своей работе может генерировать огромные массивы информации и сильно загружать работу сети, вплоть до отказов в её работе. Поэтому при анализе трафика следует применять рациональный подход — в зависимости от ситуации и условий задачи (или проблемы) использовать фильтры, это является очень эффективной частью функционала утилиты tcpdump.

Наиболее часто используемые ключи при запуске tcpdump приведены таблице:

Ключ	Описание
-a	Преобразовывает сетевые и широковещательные адреса в доменные имена.
-e	Отображает данные канального уровня (MAC-адрес, протокол, длина пакета). Помимо IP-адресов будут отображаться MAC-адреса компьютеров.
-F файл	Использовать фильтр, находящийся в файле. Если вы используете этот параметр, фильтр из командной строки будет игнорироваться.
-i	Указывает на то, какой сетевой интерфейс будет использоваться для захвата пакетов. По умолчанию — eth0, для выбора всех интерфейсов — any. Если отсутствует локальная сеть, то можно воспользоваться интерфейсом обратной связи lo.
-l	Использовать стандартный потоковый вывод tcpdump (stdout), например для записи в файл: shell# tcpdump -l | tee out.log //отобразит работу tcpdump и сохранит результат в файле out.log
-N	Не добавляет доменное расширение к именам узлов. Например, tcpdump отобразит ‘net’ вместо ‘net.library.org’
-n	Отображает IP-адрес вместо имени хоста.
-nn	Отображает номер порта вместо используемого им протокола.
-p	Не переводит интерфейс в режим приема всех пакетов (promiscuous mode).
-q	Выводит минимум информации. Обычно это имя протокола, откуда и куда шел пакет, порты и количество переданных данных.
-r	Этот параметр позволяет tcpdump прочесть трафик из файла, если он был предварительно сохранен параметром -w.
-S	Позволяет не обрабатывать абсолютные порядковые номера (initial sequence number — ISN) в относительные.
-s число	Количество байтов пакета, которые будет обрабатывать tcpdump. При установке большого числа отображаемых байтов информация может не уместиться на экране и её будет трудно изучать. В зависимости от того, какие цели вы преследуете, и следует выбирать значение этого параметра. По умолчанию tcpdump захватывает первые 68 байт (для SunOS минимум 96 байт), однако если вы хотите увидеть содержимое всего пакета, используйте значение в 1514 байт (максимально допустимый размер кадра в сети Ethernet).
-t	Не отображает метку времени в каждой строке.
-T тип	Интерпретация пакетов заданного типа. Поддерживаются типы aodv, cnfp, rpc, rtp, rtcp, snmp, tftp, vat, wb.
-tt	Отображает неформатированную метку времени в каждой строке.
-tttt	Показывает время вместе с датой.
-v	Вывод подробной информации (TTL; ID; общая длина заголовка, а также его параметры; производит проверку контрольных сумм IP и ICMP-заголовков)
-vv	Вывод ещё более полной информации, в основном касается NFS и SMB.
-vvv	Вывод максимально подробной информации.
-w файл	Сохраняет данные tcpdump в двоичном формате. Преимущества использования данного способа по сравнению с обычным перенаправлением в файл является высокая скорость записи и возможность чтения подобных данных другими программами, например snort, но этот файл нельзя прочитать человеку. Возможен вывод двоичных данных на консоль, для этого необходимо использовать -w —

-x	Делает распечатку пакета в шестнадцатеричной системе, полезно для более детального анализа пакета. Количество отображаемых данных зависит от параметра -s
-xx	То же, что и предыдущий параметр **-x**, но включает в себя заголовок канального уровня
-X	Выводит пакет в ASCII- и hex-формате. Полезно в случае анализа инцидента, связанного со взломом, так как позволяет просмотреть какая текстовая информация передавалась во время соединения.
-XX	То же, что и предыдущий параметр **-X**, но включает заголовок канального уровня.
-c число	tcpdump завершит работу после получения указанного числа пакетов.
-U	Собранные пакеты будут сразу складываться в файл, а иначе копиться в памяти до тех пор, пока она не закончится.

Фильтры tcpdump

Фильтры разделяются на следующие классификации

Тип:

host— адрес узла сети;

port – порт, на котором нужно ловить пакеты;

portrange – диапазон портов;

net – сеть.

Примеры:

```
# tcpdump net 192.168.0.0/24
```

захват всего трафика, в котором в качестве источника или получателя стоят ip адреса из сети 192.168.0.0/24.

```
# tcpdump port 80
```

Будет захватываться весь трафик на 80-м порту.

Направление трафика по отношению к объекту мониторинга:

src – отправитель;

dst — получатель.

Например, команда src host 172.31.25.200 – захват трафика, у которого отправитель ip адрес 172.31.25.200.

Протокол:

ether – базовая сетевая технология Ethernet, как правило указывает на то, что в фильтре используется аппаратный MAC адрес;

ip – протокол IPv4;

ip6 – протокол IPv6;

arp – протокол ARP;

tcp – протокол TCP;

udp – протокол UDP.

Если протокол не указан, то будет захвачен трафик по все протоколам. Например, команда udp port 5060 означает захват трафика по протоколу udp порт 5060.

Составные фильтры

Для того, чтобы более гибко фильтровать трафик можно использовать логические операции

«И» – and (&&)

«ИЛИ» – or (||)

«НЕ» – not (!) – инверсия значения

При этом приоритет этих операций следующий:

наивысшим приоритетом обладает операция инверсии;

потом логическое «И»;

наименьшим приоритетом обладает операция «ИЛИ».

Приоритет операций можно менять с помощью круглых скобок.

(net 172.16.0.0/24 or host 172.31.0.5) and tcp port 80– захват трафика протокола TCP и использующего порт 80 принадлежащего сети 172.16.0.0/24 или хосту 172.31.0.5, как любому хосту по отдельности, так и вместе.

(net 172.16.0.0/24 || host 172.31.0.5) && not tcp port 80 – захват любого трафика кроме трафика протокола TCP и использующего порт 80 принадлежащего сети 172.16.0.0/24 или хосту 172.31.0.5 как любому хосту по отдельности, так и вместе.

Команда iftop

Утилита iftop предоставляет информацию об активных сетевых соединениях, скорость сетевой закачки/отдачи, слушает трафик на указанном интерфейсе и отображает таблицу текущего использования по парам хостов.

Примеры:

Вывести весь трафик интерфейсе eth0:

```
# iftop -i eth0
```

Увидеть прохождение трафика в байтах (по умолчанию iftop выводит статистику в битах):

```
# iftop -i eth0 -B
```

Увидеть трафик между локальным интерфейсом и хостом 8.8.8.8:

```
# iftop -i eth0 -f “dst 8.8.8.8”
```

Увидеть трафик, идущий по SSH (можно использовать номер порта или же название протокола):

```
# iftop -i eth0 -f “dst port 22”
```

Увидеть трафик по протоколу HTTP:

```
# iftop -i eth0 -f “dst port http”
```

Увидеть трафик протокола DNS:

```
# iftop -i eth0 -f “dst port domain”
```

Увидеть трафик ICMP:

```
# iftop -i eth0 -f “icmp”
```

Отображать трафик по SSH с хоста XXXXXXX:

```
# iftop -i eth0 -f “port ssh and host XXXXXXX”
```

Отображать весь трафик, кроме широковещательных запросов:

```
# iftop -i eth0 -f “not ether host ff:ff:ff:ff:ff:ff”
```

Задание

Изучить функционирование команд tcpdump и iftop.

Использовать некоторые опции и флаги при запуске команд.

Сделать выводы по результатам работы.

Алгоритм выполнения лабораторной работы

1. По протоколу SSH выполнить вход на Зонд агрегации с правами администратора;

Пример:

```
mtusiadmin@kmut-ml-mtusi:~$ ssh -p 8022 user@10.19.47.5
Authorized use only. All activity may be monitored and
reported.
user@10.19.47.5's password:
Linux ZA-M11 6.6.63-rt46-kmut x86_64
----------------------
   Welcome to probe!
----------------------
To configure BGP please use "vtysh" command.
Last login: Mon Jan 26 14:16:36 2026 from 10.19.47.232
user@ZA-M11:~$ su -
Password:
root@ZA-M11:~#
```

2. Выполнить команду tcpdump на любых двух сетевых интерфейсах;
3. Собрать не менее 100 пакетов;
4. Выполнить команду tcpdump на одном из сетевых интерфейсов с любым флагом и фильтром. Описать полученную команду;
5. Зафиксировать полученные значения;
6. Выполнить команду iftop на одном из сетевых интерфейсов с любым флагом и фильтром. Описать полученную команду;

7. Зафиксировать полученные значения;
8. Сделать анализы и выводы по результатам полученных значений.

Утилиты tcpdump и iftop установлены на следующем оборудовании Лаборатории:
ZA-M11,
CPE1-M5,
CPE2-M5,
CH-M11.

Содержание отчета

1. Титульный лист;
2. Цель работы;
3. Теоретические сведения;
4. Описание выполнения работы (по шагам);
5. Полученные результаты (выводы команд, скриншоты);
6. Анализ и выводы.

Контрольные вопросы

1. Что такое сниффер?
2. В каких случаях сниффер можно использовать в благородных целях?
3. В каких случаях сниффер можно использовать в деструктивных целях?
4. Какая разница между утилитами tcpdump и iftop?
5. Может ли утилита tcpdump анализировать другие протоколы помимо TCP?

7.2.3. Изучение работы статической маршрутизации.

Ключевые слова

Статический маршрут, маршрут по умолчанию.

Цель работы

Изучить работу статической маршрутизации, настроек статической маршрутизации на сетевом оборудовании, настроек сетевого маршрута по умолчанию.

Краткие теоретические сведения

Протоколы маршрутизации обеспечивают поиск и фиксацию маршрутов продвижения пакетов данных через сеть связи TCP/IP.

Различают протоколы, выполняющие статическую и адаптивную (динамическую) маршрутизацию.

При статической маршрутизации все записи в таблице имеют неизменяемый, статический статус, что подразумевает бесконечный срок их жизни. Записи о маршрутах составляются и вводятся в память каждого маршрутизатора вручную администратором сети. При изменении состояния сети администратору необходимо срочно отразить эти изменения в соответствующих таблицах маршрутизации, иначе может произойти их рассогласование и сеть связи будет работать некорректно.

Статические маршруты рекомендуется использовать в небольших сетях, для которых задан только один путь к внешней сети. Они также обеспечивают безопасность в больших сетях с определённым типом трафика или в каналах к другим сетям, для которых требуются расширенные функции контроля.

Статическая маршрутизация имеет три основных назначения:

1) обеспечение упрощённого обслуживания таблиц маршрутизации в небольших сетях, где не планируется расширение;

2) маршрутизация к тупиковым сетям и от них (тупиковая сеть представляет собой сеть, доступ к которой осуществляется через один маршрут, и маршрутизатор имеет только одно соседнее устройство);

3) использование маршрута по умолчанию для предоставления пути к любой сети, не имеющего более точного совпадения с другим маршрутом в таблице маршрутизации.

Маршруты по умолчанию используются для отправки трафика в любой пункт назначения за пределами следующего маршрутизатора.

Таблица маршрутизации и маршрут по умолчанию

Для подробного вывода таблицы маршрутизации используют команду **routel** (от пользователя root).

```
root@ZA-M11:~# routel
Dst             Gateway         Prefsrc         Protocol Scope   Dev             Table
default          10.19.47.1                                        eth0
10.10.4.0/30     198.18.20.10                      bgp             eth2
10.19.47.0/24                     10.19.47.5       kernel   link    eth0
198.18.10.0/28                    198.18.10.1      kernel   link    eth1
198.18.20.0/28                    198.18.20.1      kernel   link    eth2
10.19.47.5                       10.19.47.5       kernel   host    eth0            local
10.19.47.255                     10.19.47.5       kernel   link    eth0            local
127.0.0.0/8                      127.0.0.1        kernel   host    lo              local
127.0.0.1                        127.0.0.1        kernel   host    lo              local
127.255.255.255                  127.0.0.1        kernel   link    lo              local
198.18.10.1                      198.18.10.1      kernel   host    eth1            local
198.18.10.15                     198.18.10.1      kernel   link    eth1            local
198.18.20.1                      198.18.20.1      kernel   host    eth2            local
198.18.20.15                     198.18.20.1      kernel   link    eth2            local
```

Чаще всего для просмотра таблицы маршрутизации в ОС Linux используют команду **ip r**.

```
user@ZA-M11:~$ ip r
default via 10.19.47.1 dev eth0 onlink
10.10.4.0/30 nhid 10 via 198.18.20.10 dev eth2 proto bgp metric 20
10.19.47.0/24 dev eth0 proto kernel scope link src 10.19.47.5
198.18.10.0/28 dev eth1 proto kernel scope link src 198.18.10.1
198.18.20.0/28 dev eth2 proto kernel scope link src 198.18.20.1
```

Вывод похож на результат предыдущей команды, но выглядит не совсем привычно, это потому, что вывод команды можно использовать в качестве аргумента для ip route add или ip route del. Это очень удобно. Как вы видите, в качестве шлюза по умолчанию везде используется 10.19.47.1. Рассмотрим подробнее что означает вывод этой команды:

- **default** - в данной строке означает маршрут по умолчанию. Здесь должен быть IP адрес цели или маска подсети;
- **via 198.18.20.10** - указывает через какой шлюз мы можем добраться до этой цели, у нас это 198.18.20.10;
- **dev eth0** - сетевой интерфейс, с помощью которого будет доступен этот шлюз;

- **proto static** - означает, что маршрут был установлен администратором, значение kernel означает, что он был установлен ядром;
- **metric** — это приоритет маршрута, чем меньше значение - тем выше приоритет.

Можно настраивать таблицу маршрутизации с помощью команды ip. Например, чтобы изменить маршрут по умолчанию через 192.168.1.1 достаточно выполнить:

```
# ip route add default via 192.168.1.1
```

Также можно добавить маршрут для любого IP адреса, например, для 243.143.5.25:

```
#  sudo ip route add 243.143.5.25 via 192.168.1.1
```

Для удаления маршрута вместо слова **add** используется слово **del**.

На сетевых устройствах Cisco для вывода таблицы маршрутизации используется команда **show ip route**.

```
CPE3#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     198.18.10.0/28 is subnetted, 1 subnets
C       198.18.10.0 is directly connected, FastEthernet0/0
     10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C       10.10.3.0/30 is directly connected, FastEthernet0/1
B       10.19.47.0/24 [20/0] via 198.18.10.1, 4w3d
```

Задание

Изучить таблицы маршрутизации на сетевых устройствах.

Сделать выводы по результатам работы.

Алгоритм выполнения лабораторной работы

1. Посмотреть таблицы маршрутизации на трех сетевых элементах (АРМ, сервер, зонд, маршрутизатор);
2. Описать параметры таблицы маршрутизации, данные шлюзов по умолчанию и сетевые интерфейсы, использующиеся при составлении маршрута;

3. Зафиксировать полученные значения;
4. Сделать анализы и выводы по результатам полученных значений.

Содержание отчета

1. Титульный лист;
2. Цель работы;
3. Теоретические сведения;
4. Описание выполнения работы (по шагам);
5. Полученные результаты (выводы команд, скриншоты);
6. Анализ и выводы.

Контрольные вопросы

1. Что такое маршрутизация?
2. Какие бывают виды маршрутизации?
3. Чем статическая маршрутизация отличается от динамической?
4. Что такое маршрут по умолчанию?
5. Какой командой можно посмотреть таблицу маршрутизации в ОС Linux?

7.2.4. Изучение протокола DHCP и механизма NAT.

Ключевые слова

Протокол DHCP, механизм NAT.

Цель работы

Изучить принципы работы протокола DHCP и механизма NAT трансляции.

Краткие теоретические сведения

Протокол DHCP

DHCP (Dynamic Host Configuration Protocol) — протокол прикладного уровня модели TCP/IP, служит для назначения IP-адреса клиенту. IP-адрес можно назначать вручную каждому клиенту, то есть сетевому устройству в локальной сети. Но в больших сетях это очень трудозатратно, к тому же, чем больше локальная сеть, тем выше возрастает вероятность ошибки при настройке. Поэтому для автоматизации назначения IP адреса был создан протокол DHCP.

Рассмотрим процесс получения IP-адреса (рис. 57):

- Когда клиент загружается (или хочет присоединиться к сети), он начинает четырехэтапный процесс для получения аренды. Он запускает процесс с широковещательным (**broadcast**) сообщением **DHCPDISCOVER** со своим собственным MAC-адресом для обнаружения доступных серверов DHCP. Поскольку у клиента нет способа узнать подсеть, к которой он принадлежит, у сообщения **DHCPDISCOVER** адрес назначения IP адреса - **255.255.255.255**. А поскольку у клиента еще нет настроенного адреса IP, то исходный IP-адрес - **0.0.0.0**.
- Сообщение **DHCPDISCOVER** находит серверы DHCP в сети. Поскольку клиент не имеет IP информации при загрузке, он использует широковещательные адреса 2 и 3 уровня для связи с сервером.
- Когда DHCP-сервер получает сообщение **DHCPDISCOVER**, он резервирует доступный IP-адрес для аренды клиенту. Сервер также создает запись ARP, состоящую из MAC-адреса клиента и арендованного IP-адреса DHCP сервер отправляет связанное сообщение **DHCPOFFER** запрашивающему клиенту, как одноадресная передача (**unicast**), используя MAC-

адрес сервера в качестве исходного адреса и MAC-адрес клиента в качестве адреса доставки.

- Когда клиент получает **DHCPOFFER** с сервера, он отправляет обратно сообщение **DHCPREQUEST**. Это сообщение используется как для получения, так и для продления аренды. Когда используется для получения аренды, **DHCPREQUEST** служит в качестве уведомления о принятии выбранных сервером параметров, которые он предложил, и отклонении предложения от других серверов. Многие корпоративные сети используют несколько DHCP серверов, и сообщение **DHCPREQUEST** отправляется в виде широковещательной передачи, чтобы информировать все серверы о принятом предложении.
- При получении сообщения **DHCPREQUEST** сервер проверяет информацию об аренде с помощью ICMP-запроса на этот адрес, чтобы убедиться, что он уже не используется и создает новую **ARP** запись для аренды клиента, а затем отвечает одноадресным DHCPACK-сообщением. Это сообщение является дубликатом **DHCPOFFER**, за исключением изменения поля типа сообщения. Когда клиент получает сообщение **DHCPACK**, он регистрирует информацию и выполняет поиск ARP для назначенного адреса. Если ответа на ARP нет, клиент знает, что адрес IP действителен и начинает использовать его как свой собственный.

Рис. 57. Процесс получения IP-адреса

Когда DHCP-сервер выделяет IP из области, он оставляет запись о том, что этот адрес зарезервирован за клиентом с указанием срока действия IP. Этот срок действия называется срок аренды (lease time). Срок аренды по умолчанию выставлен на 24

часа, но может доходить до нескольких дней, недель или даже месяцев. Период задается в настройках самого DHCP сервера.

Предоставление IP-адреса в аренду, а не на постоянной основе необходимо по нескольким причинам. Во-первых, это разумное использование IP-адресов — отключенные или вышедшие из строя клиенты не резервируют за собой адрес. Во-вторых, это гарантия того, что новые клиенты при необходимости смогут получить уникальный адрес.

Технология NAT

NAT — это технология, позволяющая перенаправлять трафик между локальной и глобальной сетями. Благодаря ей устройства могут не только взаимодействовать друг с другом в домашней сети, но и выходить в интернет.

Для использования данной технологии используют маршрутизаторы (рис. 58). Например, при открытии страницы в браузере, NAT перенаправляет трафик с локального IP-адреса смартфона или компьютера в локальной сети на основной IP, выданный провайдером, и обратно. С точки зрения сети это выглядит так, будто все запросы в сеть Интернет отправляет именно маршрутизатор, а не несколько смартфонов и ноутбуков, подключённых к нему.

Рис. 58. Пример работы технологии NAT

Преимущества технологии NAT:

- Экономия IP-адресов. Количество свободных IP-адресов в мире ограничено, а оборудование, способное выдержать столько независимых линий, стоит очень дорого. Чтобы не выдавать каждому устройству, подключённому к Глобальной сети, свой собственный адрес, проще предоставить клиенту один основной, к которому через маршрутизатор можно подключить множество гаджетов.
- Безопасность. NAT скрывает локальные IP-адреса от внешней сети, обеспечивая дополнительный уровень конфиденциальности. Современные маршрутизаторы также позволяют вручную настраивать NAT, поэтому можно запретить некоторым устройствам выходить в интернет или посещать определённые сайты.
- Удобство. NAT позволяет устройствам обмениваться данными в локальной сети без необходимости подключения к интернету, одновременно обеспечивая доступ к ресурсам Глобальной сети.

Также рассмотрим PAT (Port Address Translation), также известен как NAT с перегрузкой (NAT overloading) — технология трансляции адресов с использованием портов. Данная технология решает проблему доставки возвратных пакетов. Так как количество белых IP ограничено нам необходимо экономить эти адреса. Помня об этом, была создана технология PAT. Она позволяет локальным хостам использовать частные IP-адреса и установить один зарегистрированный адрес на маршрутизатор доступа. В технологии преобразования адресов PAT используется особенность работы протокола TCP: с точки зрения сервера абсолютно все равно, осуществляются соединения с тремя разными хостами с разными адресами или соединения устанавливаются с одним хостом на один IP-адрес, но с разными портами. Следовательно, чтобы подключить к Интернету множество хостов небольшого офиса с помощью, одного только зарегистрированного публичного IP адреса, служба PAT транслирует частные адреса локальных хостов в один имеющийся зарегистрированный. Чтобы правильно пересылать пакеты обратной коммуникации локальным хостам, маршрутизатор хранит у себя таблицу IP адресов и номеров портов для протоколов TCP и UDP.

Задание

Изучить работу протокола DHCP на учебных АРМ.

Выполнить настройку протокола DHCP и механизма NAT. Для этого собрать и исследовать схему лабораторной работы в эмуляторе.

Алгоритм выполнения лабораторной работы

1. В среде PNetLab собрать схему связи, которая должна получиться по результатам лабораторной работы (рис. 59);

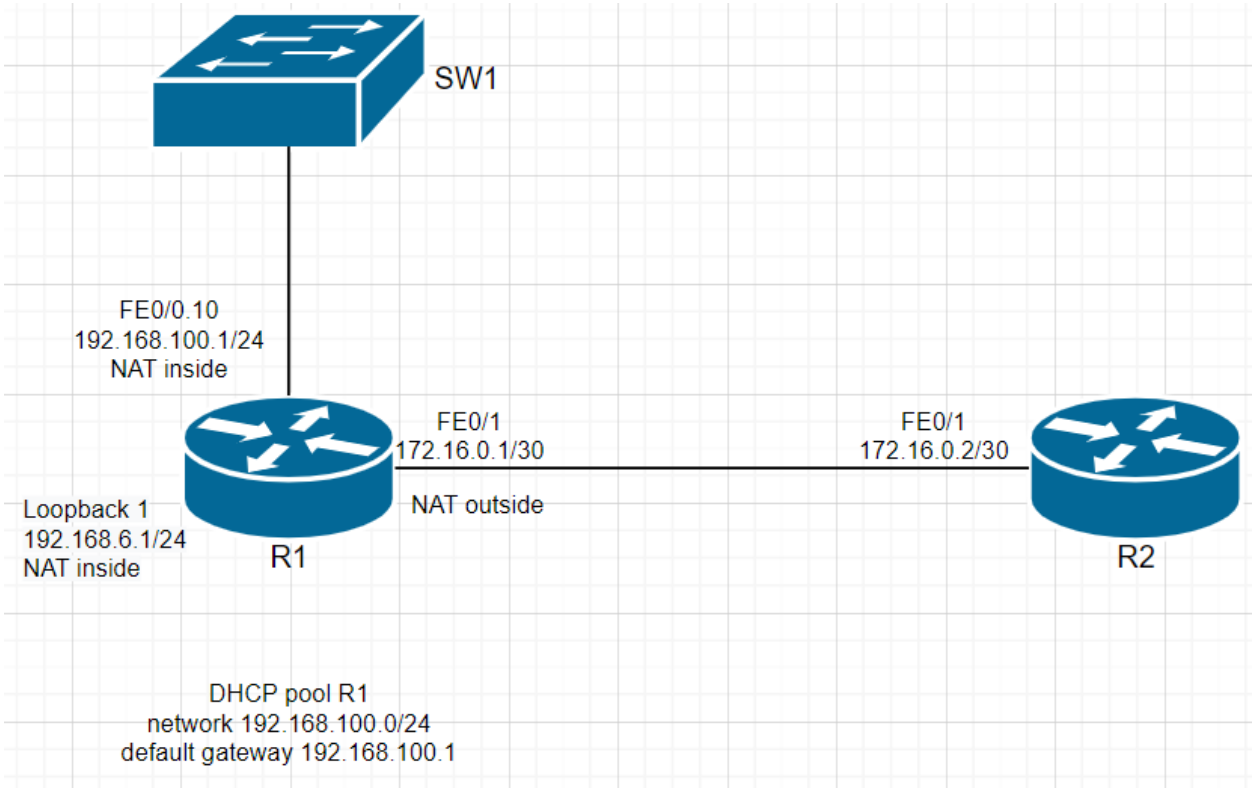


Рис. 59. Схема связи

2. Выполнить следующие настройки на оборудовании:

Настройка саб интерфейса для пользователей на R1 (рис. 60).

conf t

int fa 0/0

no shutdown

int fa 0/0.10 – создаем sub интерфейс

encapsulation dot1q 10 – делаем тег 10 sub интерфейсу (vlan 10)

ip address 192.168.100.1 255.255.255.0

```
R1(config)#interface fa
R1(config)#interface fastEthernet 0/0
R1(config-if)#no shu
R1(config-if)#no shutdown
R1(config-if)#
*Nov 29 15:12:34.887: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up
R1(config-if)#
R1(config-if)#exi
R1(config)#int fa 0/0.10
R1(config-subif)#ip ad
R1(config-subif)#ip address 192.168.100.1 255.255.255.0

% Configuring IP routing on a LAN subinterface is only allowed if that
subinterface is already configured as part of an IEEE 802.10, IEEE 802.1Q,
or ISL vLAN.

R1(config-subif)#enca
R1(config-subif)#encapsulation d
R1(config-subif)#encapsulation dot1Q 10
R1(config-subif)#ip address 192.168.100.1 255.255.255.0
R1(config-subif)#
*Nov 29 15:13:34.975: %LINK-3-UPDOWN: Interface FastEthernet0/0, changed state to up
R1(config-subif)#
```

Рис. 60. Настройка саб интерфейса для пользователей на R1

Настройка DHCP пула для пользователей (рис. 61)

ip dhcp pool LAN-USERS – создаем dhcp пул с названием LAN-USERS

network 192.168.100.0 255.255.255.0 – Указываем сеть в которой будут выдаваться IP адреса

default-router 192.168.100.1 – этот IP будет выдаваться хостам как default gateway

exit

ip dhcp excluded-address 192.168.100.1 192.168.100.10 – исключаем из выдачи dhcp сервара первые 10 IP адресов

```
R1(config)#ip dhcp pool LAN-USERS
R1(dhcp-config)#net
R1(dhcp-config)#netw
R1(dhcp-config)#network 192.168.100.0 255.255.255.0
R1(dhcp-config)#def
R1(dhcp-config)#default-router 192.168.100.1
R1(dhcp-config)#exi
R1(config)#ip dhcp
R1(config)#ip dhcp-e
R1(config)#ip dhcp-ex
R1(config)#ip dhcp ex
R1(config)#ip dhcp excluded-address 192.168.100.1 192.168.100.10
```

Рис. 61. Настройка DHCP пула для пользователей

Настройка PAT трансляции (рис. 62)

ip access-list extended USERS – создаем access list, IP адреса из него будут попадать под PAT

permit ip 192.168.100.0 0.0.0.255 any

permit ip 192.168.6.0 0.0.0.255 any

ip nat inside source list USERS interface fastethernet0/1 overload – этой командой указываем что IP из access листа USERS относятся к зоне inside, а IP адрес на fa0/1 к зоне outside. Overload специализирует NAT трансляцию как PAT.

```
R1(config)#ip nat inside source list USERS interface FastEthernet0/1 overload

R1(config)#ip access-list extended USERS
R1(config-ext-nacl)# permit ip 192.168.100.0 0.0.0.255 any
R1(config-ext-nacl)#
```

Рис. 62. Настройка DHCP пула для пользователей

int fa 0/1

ip nat outside – указываем на интерфейсе что он относится к зоне outside

int fa 0/0.10

ip nat inside – указываем на интерфейсе что он относится к зоне inside

```
R1(config)#interface fa
R1(config)#interface fastEthernet 0/1
R1(config-if)#ip nat
R1(config-if)#ip nat out
R1(config-if)#ip nat outside
R1(config-if)#exi
R1(config)#int fa 0/0.10
R1(config-subif)#ip nat
R1(config-subif)#ip nat ins
R1(config-subif)#ip nat inside
R1(config-subif)#
R1(config-subif)#
```

Рис. 63. Результат ввода команд

interface loopback 1

ip nat inside – указываем на интерфейсе что он относится к зоне inside

```
R1(config)#interface lo
R1(config)#interface loopback 1
R1(config-if)#ip na
R1(config-if)#ip nat inside
```

Рис. 64. Результат ввода команд

ip route 0.0.0.0 0.0.0.0 172.16.0.2 – указываем дефолтный маршрут в сторону R2

3. Пробуем сделать пинг любого IP с наших inside интерфейсов чтобы создались трансляции.

Проверяем трансляции с помощью **show ip nat translations**

Статистику можно посмотреть командой **show ip nat statistics** (рис. 65)

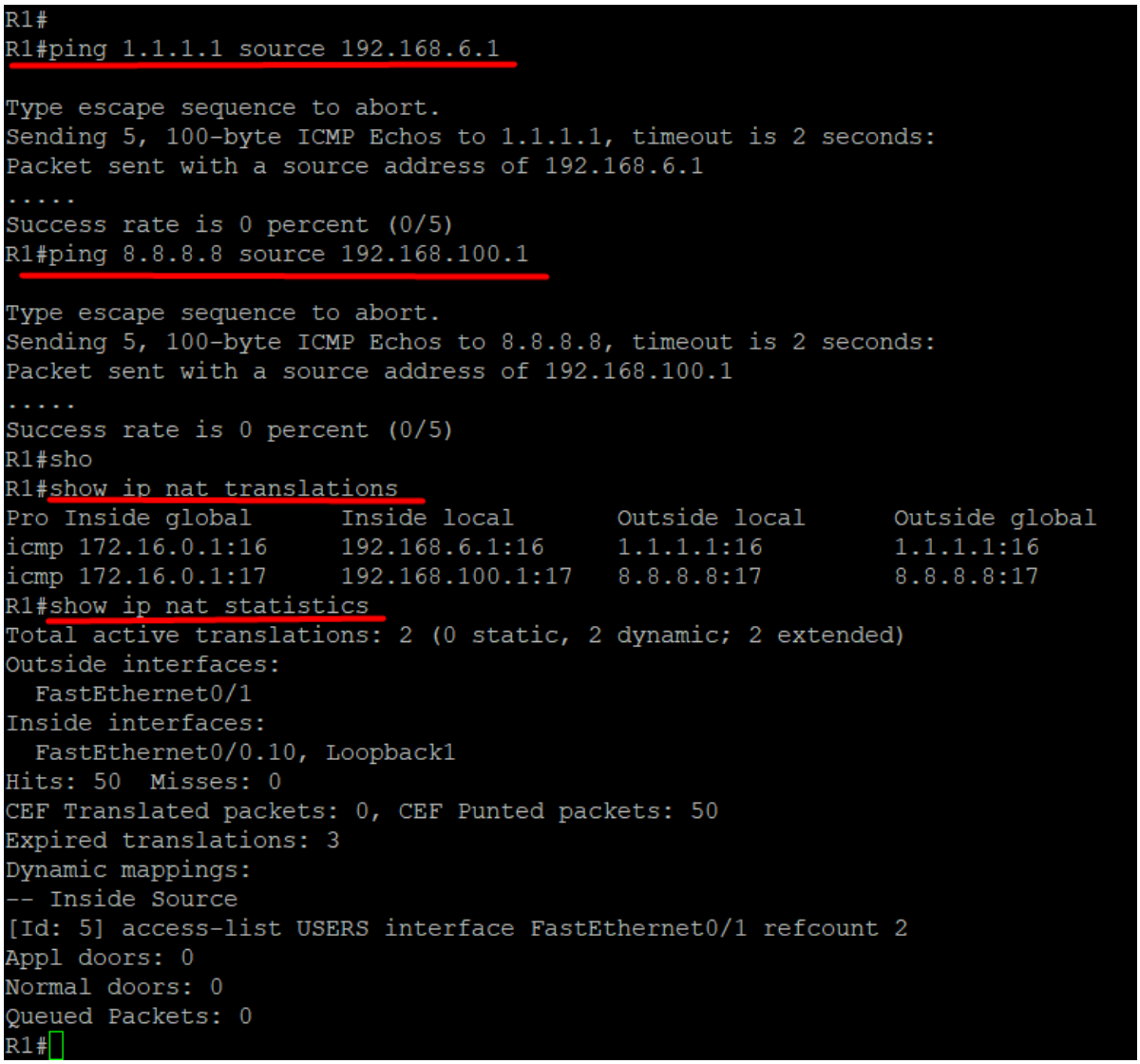


Рис. 65. Просмотри статистики

Содержание отчета

1. Титульный лист;
2. Цель работы;
3. Теоретические сведения;
4. Описание выполнения работы (по шагам);
5. Полученные результаты (выводы команд, скриншоты);
6. Анализ и выводы.

Контрольные вопросы

1. В каких случаях использую протокол DHCP?
2. Какое оборудования используется для работы технологии NAT?
3. Назовите отличия технологий NAT и PAT?

7.2.5. Изучение процесса сброса маршрутизатора до значений по умолчанию.

Ключевые слова

Reset, сброс настроек маршрутизатора, настройка конфигурации.

Цель работы

Изучить процесс возвращения конфигурации сетевого устройства до значений по умолчанию. Выполнить заливку конфигурации для функционирования сетевого устройства.

Краткие теоретические сведения

Сброс настроек сетевого устройства может потребоваться пользователю в различных ситуациях, когда ИТ-система не функционирует в штатном режиме. При этом обнуление настроек может быть как плановым, так и экстренным. В первом случае обычно речь идет про первое включение сетевого устройства в системе:

- Не выполнен обмен данными – несмотря на корректное проводное подключение ко всем нужным элементам, пользователь не получает доступа к подключенному оборудованию. Также это актуально и в ситуациях, когда маршрутизатор используют для выхода в интернет, но не получают доступа;
- Устройство ранее было использовано в другой локальной сети – как при обновлении оборудования ИТ-структуры, так и при первичном проектировании системы могут применять старые агрегаты. Например, в случае релокации коммерческого предприятия или с целью сократить бюджет на формирование серверной. Особенно важно сделать сброс настроек, если роутер был подключен к другому интернет-провайдеру.

В обоих случаях требуется сбросить настройки сетевого устройства до заводских, чтобы упростить выбор индивидуальных параметров системы, а также обеспечить безопасность данных.

Также может потребоваться и экстренная процедура сброса. К примеру, если пользователю необходимо скорректировать параметры маршрутизатора, но прежние настройки не сохранились. Еще один вариант – внезапный сбой программных настроек

вследствие внешних факторов, например, перебоев с напряжением в сети.

Иногда пользователь сталкивается с ситуацией, когда программная настройка невозможна из-за недостатка исходных данных от поставщика услуг связи. К примеру, отсутствия пароля доступа к программной настройке оборудования. Такую проблему также может решить обнуление настроек до заводских параметров и базовый вход в программный интерфейс.

Сброс настроек сетевого устройства до значений по умолчанию (заводских настроек) можно выполнить двумя способами: через веб-интерфейс сетевого устройства или с помощью кнопки Reset или F на корпусе устройства.

Задание

Изучить процесс сброса и восстановления конфигурации на Зонде КМУТ М7.

Алгоритм выполнения лабораторной работы

1. Ознакомиться с п.3.2; 7.1.7 и п.7.1.9 данного Методического указания;
2. Выполнить вход по протоколу SSH на Зонд КМУТ М7;
3. Выполнить сохранение настроек Зонда, а именно содержимого файла **/etc/kmut/main.conf**;
4. Выполнить сброс настроек Зонда нажатием на кнопку F не менее 3 секунд;
5. Подключить Зонд к АРМ и просмотреть содержимое файла **/etc/kmut/main.conf;**
6. Выполнить повторную настройку Зонда согласно сохраненным настройкам.

Содержание отчета

1. Титульный лист;
2. Цель работы;
3. Теоретические сведения;
4. Описание выполнения работы (по шагам);
5. Полученные результаты (выводы команд, скриншоты);
6. Выводы по результатам выполнения работы.

Контрольные вопросы

1. Зачем выполняется сброс настроек сетевого оборудования?

2. Каким образом можно осуществить сброс настроек сетевого оборудования?

7.2.6. Изучение управления трафиком по протоколам семейства STP.

Ключевые слова

Алгоритмы связующего дерева, протокол STP, протокол RSTP, протокол MSTP.

Цель работы

Ознакомиться с основными особенностями протоколов связующего дерева STP, RSTP, MSTP.

Краткие теоретические сведения

Протоколы семейства Spanning Tree Protocol (STP)

В настоящее время для повышения надежности и производительности каналов связи существует ряд протоколов и функций. Наиболее распространены методы создания резервных связей между коммутаторами на основе двух технологий:

– резервирование соединений с помощью протоколов семейства Spanning Tree;

– балансировка нагрузки, обеспечивающая параллельную передачу данных по всем альтернативным соединениям с помощью механизма агрегирования портов.

Перейдем к рассмотрению протокола связующего дерева. Spanning Tree Protocol (STP) является протоколом 2-го уровня модели OSI и позволяет строить древовидные свободные от петель конфигурации связей между коммутаторами локальной сети. Помимо этого, STP обеспечивает возможность автоматического резервирования альтернативных каналов связи между коммутаторами на случай выхода из строя активных каналов.

В настоящее время существуют следующие версии протоколов связующего дерева:

– IEEE 802.1D Spanning Tree Protocol (STP);

– IEEE 802.1w Rapid Spanning Tree Protocol (RSTP);

– IEEE 802.1s Multiple Spanning Tree Protocol (MSTP).

Фактически протокол STP является специфической упрощенной версией протокола маршрутизации. Упрощение — в том, что кадры направляются по активному маршруту независимо от их адреса назначения, в то время как в протоколах маршрутизации активный маршрут выбирается для каждого адреса индивидуально.

Сегодня протокол STP широко применяется в наиболее массовых устройствах современных локальных сетей — коммутаторах. Версия протокола STP, получившая название RSTP (Rapid STP, то есть быстрый протокол покрывающего дерева), затрачивает на поиск новой топологии несколько секунд.

Протокол STP

Протокол связующего дерева Spanning Tree Protocol (STP) является протоколом 2 уровня модели OSI, который позволяет строить древовидные, свободные от петель, конфигурации связей между коммутаторами локальной сети. Т.е. протокол приводит сеть к виду – «корень» и растущие из него «ветви». Один из свитчей становится «корнем» (root bridge), затем все остальные рассчитывают «стоимость» (cost) достижения корня из всех своих портов, имеющих такую возможность, и отключают все неоптимальные линки. Таким образом, разрывая «петли». Если в дальнейшем в сети произойдёт сбой, и «корень» вдруг окажется недостижим через работающий порт, включится лучший из ранее заблокированных и связь будет восстановлена.

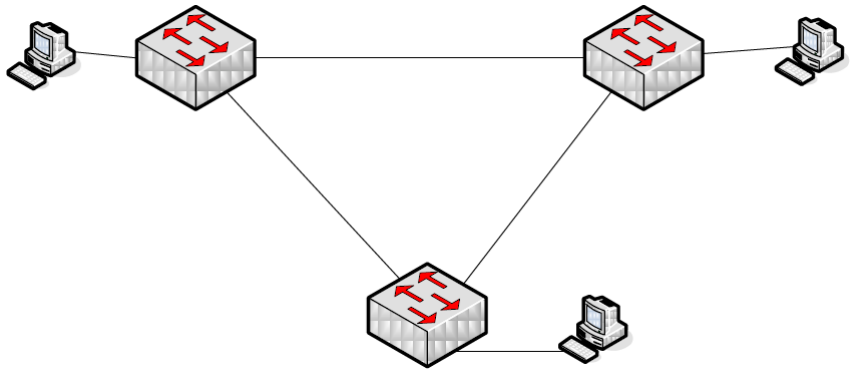


Рис. 66. Схема до применения протокола STP

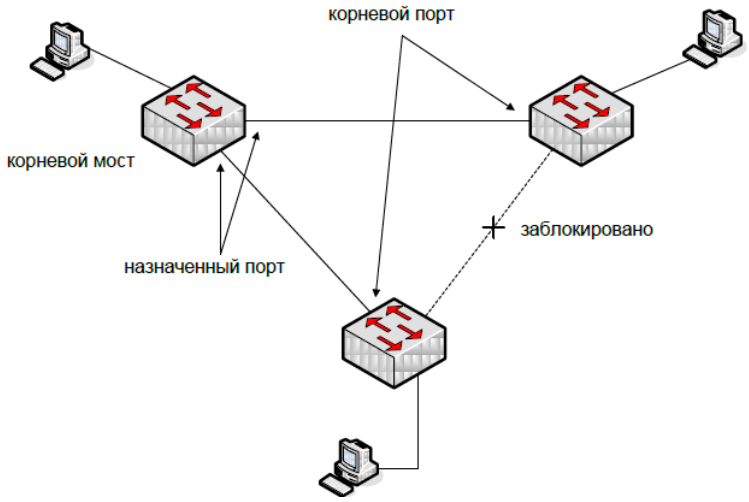


Рис. 67. Схема после применения протокола STP

Рассмотрим эти процессы подробнее (рис. 66 и 67).

Процесс вычисления связующего дерева начинается с выбора корневого моста (root bridge), от которого будет строиться дерево. В качестве корня дерева выбирается коммутатор с наименьшим значением идентификатора моста (Bridge ID). Идентификатор моста – это 8-байтное поле, которое состоит из 2-ч частей: приоритета моста (2 байта), назначаемого администратором сети и MAC-адреса блока управления коммутатора (6 байт). При сравнении идентификаторов двух коммутаторов сначала сравниваются значения приоритета. Если они одинаковы, то корневой мост определяется по наименьшему MAC-адресу.

Когда процесс выбора корневого моста завершен, оставшиеся коммутаторы определяют «стоимость» (cost) от себя до корня дерева.

Стоимость пути рассчитывается как суммарное условное время на передачу данных от порта данного коммутатора до порта корневого моста. Сравнив стоимости всех возможных маршрутов до корня, каждый коммутатор выбирает среди них один с наименьшим значением «стоимости». Порт, соединяющий коммутатор с этим маршрутом, становится корневым портом. В случае если минимальные стоимости пути нескольких маршрутов окажутся одинаковыми, корневым портом станет порт, имеющий наименьшее значение идентификатора порта.

Далее определяются назначенные порты (Designated Port). Каждый сегмент в коммутируемой сети имеет один назначенный порт. Этот порт функционирует как единственный порт моста, т.е. принимает кадры от сегмента и передает их в направлении корневого моста через корневой порт данного коммутатора. Коммутатор, содержащий назначенный порт для данного сегмента, называется назначенным мостом (Designated Bridge) этого сегмента. Назначенный порт сегмента определяется путем сравнения значений стоимости пути всех маршрутов от данного сегмента до корневого моста. Им становится порт, имеющий наименьшее значение стоимости, среди всех портов, подключенных к данному сегменту. Если минимальные значения стоимости пути окажутся одинаковыми у двух или нескольких портов, то для выбора назначенного порта сегмента STP принимает решение на основе

последовательного сравнения идентификаторов мостов и идентификаторов портов.

У корневого моста все порты являются назначенными, а их расстояние до корня полагается равным нулю. Корневого порта у корневого моста нет.

После выбора корневых и назначенных портов все остальные порты коммутаторов сети переводятся в состояние Blocking («Блокировка»), то есть такое, при котором они принимают и передают только кадры BPDU. При таком выборе активных портов в сети исключаются петли, и оставшиеся связи образуют связующее дерево.

Все эти вычисления требуют периодического обмена информацией между коммутаторами связующего дерева, что достигается при помощи специальных кадров, называемых блоками данных протокола моста – BPDU (Bridge Protocol Data Unit).

Существует три типа кадров BPDU:

- Configuration BPDU (CBPDU) – конфигурационный кадр BPDU, который используется для вычисления связующего дерева;
- Topology Change Notification (TCN) BPDU – уведомление об изменении топологии сети;
- Topology Change Notification Acknowledgement (TCA) – подтверждение о получении уведомления об изменении топологии сети.

Коммутаторы обмениваются BPDU через равные интервалы времени (по умолчанию 2 секунды), что позволяет им отслеживать состояние топологии сети.

В процессе построения топологии сети каждый порт коммутатора проходит несколько стадий (состояний портов):

- Blocking («Блокировка») – при инициализации коммутатора все порты (за исключением отключенных) автоматически переводятся в состояние «Заблокирован». В этом случае порт принимает и обрабатывает только кадры BPDU. Все остальные кадры отбрасываются;
- Listening («Прослушивание») – в этом состоянии порт продолжает принимать, обрабатывать и ретранслировать только кадры BPDU. Из этого состояния порт может перейти в состояние «Заблокирован», если получит BPDU с лучшими параметрами, чем его собственные (стоимость пути, идентификатор моста или порта).

В противном случае, при истечении периода, установленного таймером задержки смены состояний (Forward Delay), порт перейдет в следующее состояние «Обучение»;

- Learning («Обучение») – порт начинает принимать все кадры и на основе MAC-адресов источника строить таблицу коммутации. Порт в этом состоянии все еще не продвигает кадры. Порт продолжает участвовать в алгоритме STP и при поступлении BPDU с лучшими параметрами переходит в состояние «Заблокирован». В противном случае, при истечении периода, установленного таймером задержки смены состояний, порт перейдет в следующее состояние «Продвижение»;
- Forwarding («Продвижение») – в этом состоянии порт может обрабатывать кадры данных в соответствии с построенной таблицей коммутации. Также продолжают приниматься, передаваться и обрабатываться кадры BPDU;
- Disable («Отключен») – в это состояние порт переводит администратор. Отключенный порт не участвует ни в работе протокола STP, ни в продвижении кадров данных. Порт можно также вручную включить, и первоначально он перейдет в состояние «Заблокирован».

Протокол RSTP

Протокол Rapid Spanning Tree Protocol (RSTP) является развитием протокола STP. Он был разработан для преодоления отдельных ограничений протокола STP, связанных с его производительностью. Протокол RSTP значительно ускоряет время сходимости коммутируемой сети за счет мгновенного перехода корневых и назначенных портов в состояние продвижения.

Протокол MSTP

Протокол Multiple Spanning Tree Protocol (MSTP), являющийся расширением протокола RSTP, преодолевает это ограничение. В дополнение к обеспечению быстрой сходимости сети он позволяет настраивать отдельное связующее дерево для любой VLAN или группы VLAN, создавая множество маршрутов передачи трафика и позволяя осуществлять балансировку нагрузки.

Логическая структура MSTP

Протокол MSTP делит коммутируемую сеть на регионы MST (Multiple Spanning Tree (MST) Region), каждый из которых может содержать множество копий связующих деревьев (Multiple Spanning Tree Instance, MSTI) с независимой друг от друга топологией. Другими словами, регион MST, представляющий собой набор физически подключенных друг к другу коммутаторов, делит данную физическую топологию на множество логических (рис 68).

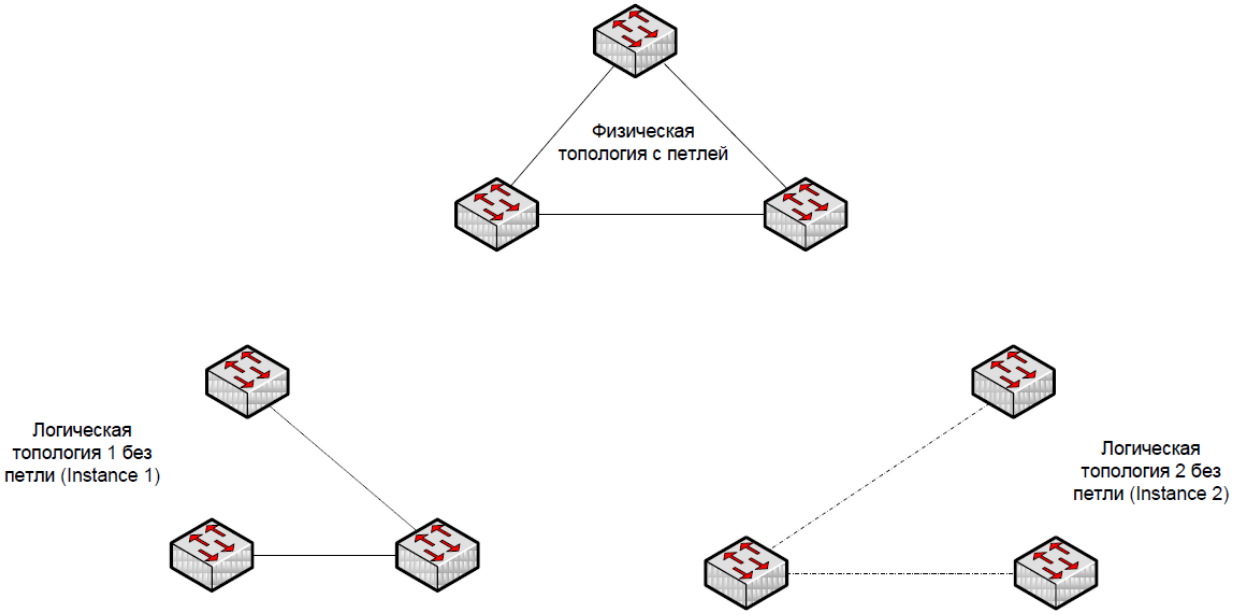


Рис. 68. Физическая и логическая топология региона MST

Для того, чтобы два и более коммутатора принадлежали одному региону MST, они должны обладать одинаковой конфигурацией MST. Конфигурация MST включает такие параметры, как номер ревизии MSTP (MSTP revision level number), имя региона (Region name), карту привязки VLAN к копии связующего дерева (VLAN-to-instance mapping).

Роли портов MSTP:

- Корневой порт (Root Port) – это порт, который обладает минимальной стоимостью пути от коммутатора до корневого моста.
- Назначенный порт (Designated Port) – это порт, обладающий наименьшей стоимостью пути от подключенного сегмента сети до корневого моста.
- Альтернативный/резервный порт (Alternate/Backup Port) – это порт, который обеспечивает подключение, если происходит потеря соединения с какими-либо коммутаторами или сегментами сети.

➢ Мастер-порт (Master Port) – это порт, который обеспечивает подключение региона к корневому мосту, находящемуся за пределами данного региона.

➢ Пограничный порт (Boundary Port) – это порт, который подключает MST-регион к другому региону или SST-мосту.

Задание

Изучить процесс сброса и восстановления конфигурации на Зонде КМУТ М7.

Алгоритм выполнения лабораторной работы

Работу выполнять в виртуальной среде эмулирования PNetLab.

При достаточной подготовке возможно выполнить работу на оборудовании лаборатории, используя коммутаторы AS1, AS3, AS4, AS5 и подключенные к ним Зонды КМУТ М7.

1. Собрать схему связи согласно рис. 69. Не соединять порты коммутатора с образованием петли во время настройки.

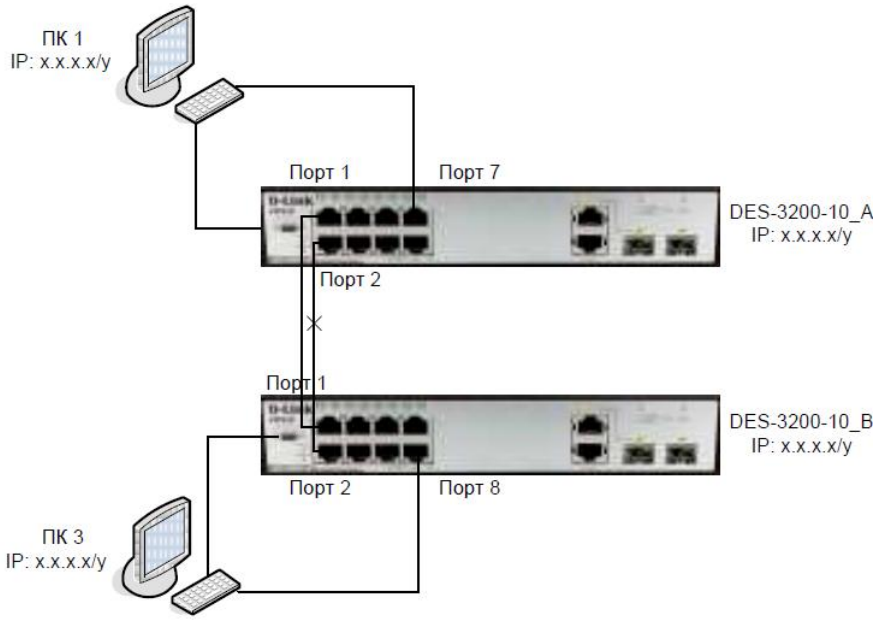


Рис. 69. Схема соединения коммутаторов для настройки STP

2. Настройте IP-адрес интерфейса управления коммутатора командой config ipif System ipaddress x.x.x.x/y, где y – инверсная маска подсети.

3. Включите протокол связующего дерева командой enable stp.

4. Проверьте текущую конфигурацию протокола связующего дерева командой show stp.

5. Протокол RSTP используется по умолчанию после активизации протокола связующего дерева. Если нет, включите его командой config stp version rstp

6. Установите на коммутаторе наименьшее значение приоритета, чтобы он мог быть выбран корневым мостом (приоритет по умолчанию = 32768) командой config stp priority 4096 instance_id 0.

7. Назначьте порты 3-8 граничными портами командой config stp ports 3-8 edge true.

8. Активизируйте протокол связующего дерева на портах config stp ports 1-8 state enable.

9. Соединить порты коммутатора вторым кабелем.

10. Выполните продолжительный ping от ПК1 до ПК3 и наоборот командой ping x.x.x.x -t

11. Поменяйте RSTP на STP командой config stp version stp.

12. Выполните продолжительный ping от ПК1 до ПК3 и наоборот командой ping x.x.x.x –t

Содержание отчета

1. Титульный лист;
2. Цель работы;
3. Теоретические сведения;
4. Описание выполнения работы (по шагам);
5. Полученные результаты (выводы команд, скриншоты);
6. Выводы по результатам выполнения работы.

Контрольные вопросы

1. Какое предназначение протокола STP? На каком уровне модели OSI работает протокол?
2. Назовите основные отличия протоколов STP, RSTP, MSTP?
3. Принцип действия протокола STP?
4. Что такое «петля»?
5. Что такое корневой мост, корневой порт?
6. Что такое назначенный мост, назначенный порт?
7. Что такое граничный порт?
8. Какие бывают роли портов MSTP?
9. Какие бывают состояния портов MSTP?
10. Что такое BPDU?
11. Расскажите о логической структуре MSTP?

7.2.7. Изучение виртуальных локальных сетей.

Ключевые слова

Виртуальная локальная сеть, VLAN, архитектура Router-on-a-Stick.

Цель работы

Изучить технологию VLAN и сегментации сети. Изучить настройки сетевого оборудования с такой технологией. Ознакомиться с архитектурой Router-on-a-Stick.

Краткие теоретические сведения

Определение VLAN

Важным свойством коммутатора локальной сети является способность контролировать передачу кадров между сегментами сети. По различным причинам (соблюдение прав доступа, политика безопасности и т.д.) некоторые кадры не следует передавать по адресу назначения.

Ограничения такого типа можно реализовать с помощью пользовательских фильтров (см. ранее). Но пользовательский фильтр может запретить коммутатору передачу кадров только по конкретным адресам, а широковещательный трафик он в соответствии с алгоритмом работы обязан передать всем сегментам сети. Поэтому сети, созданные на основе коммутаторов, иногда и называют плоскими — из-за отсутствия барьеров на пути широковещательного трафика. Технология виртуальных локальных сетей позволяет преодолеть указанное ограничение.

Виртуальной локальной сетью (Virtual Local Area Network, VLAN) называется группа узлов сети, трафик которой, в том числе широковещательный, на канальном уровне полностью изолирован от трафика других узлов сети.

Это означает, что передача кадров между разными виртуальными сетями на основании адреса канального уровня невозможна независимо от типа адреса (уникального, группового или широковещательного). В то же время внутри виртуальной сети кадры передаются по технологии коммутации, то есть только на тот порт, который связан с адресом назначения кадра.

Виртуальные локальные сети могут перекрываться, если один или несколько компьютеров входят в состав более чем одной

виртуальной сети. На рис. 70 сервер электронной почты входит в состав виртуальных сетей 3 и 4. Это означает, что его кадры передаются коммутаторами всем компьютерам, входящим в эти сети. Если же какой-то компьютер входит в состав только виртуальной сети 3, то его кадры до сети 4 доходить не будут, но он сможет взаимодействовать с компьютерами сети 4 через общий почтовый сервер. Такая схема защищает виртуальные сети друг от друга не полностью — например, широковещательный шторм, возникший на сервере электронной почты, затопит и сеть 3, и сеть 4. В подобных случаях принято считать, что виртуальная сеть образует домен широковещательного трафика по аналогии с доменом коллизий, образуемым повторителями сетей Ethernet.

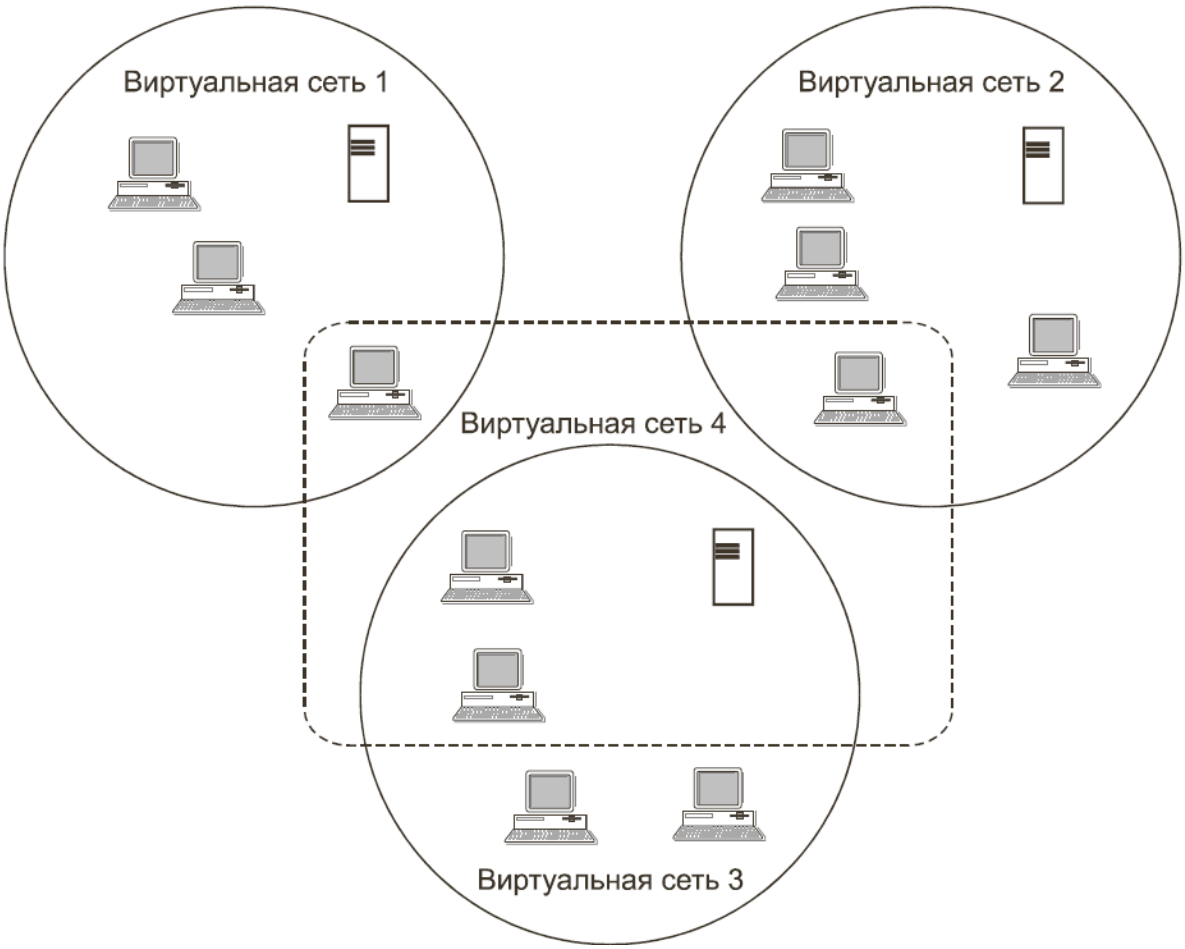


Рис. 70. Виртуальные локальные сети

Достоинства использования VLAN

➢ Гибкое разделение устройств на группы

Как правило, одному VLAN соответствует одна подсеть. Компьютеры, находящиеся в разных VLAN, будут изолированы друг от друга. Также можно объединить в одну виртуальную сеть компьютеры, подключенные к разным коммутаторам.

➢ Уменьшение широковещательного трафика в сети

Каждый VLAN представляет отдельный широковещательный домен. Широковещательный трафик не будет транслироваться между разными VLAN. Если на разных коммутаторах настроить один и тот же VLAN, то порты разных коммутаторов будут образовывать один широковещательный домен.

➢ Увеличение безопасности и управляемости сети

В сети, разбитой на виртуальные подсети, удобно применять политики и правила безопасности для каждого VLAN. Политика будет применена к целой подсети, а не к отдельному устройству.

➢ Уменьшение количества оборудования и сетевого кабеля

Для создания новой виртуальной локальной сети не требуется покупка коммутатора и прокладка сетевого кабеля. Однако вы должны использовать более дорогие управляемые коммутаторы с поддержкой VLAN.

Конфигурирование VLAN

Существуют различные подходы к конфигурированию виртуальных локальных сетей, построенных на нескольких коммутаторах. Наиболее распространенным является подход, основанный на понятиях линии доступа и транка.

Линия доступа связывает порт коммутатора (называемый в этом случае портом доступа, access) с конечным узлом (компьютером, мобильным устройством и т. п.), принадлежащим некоторой виртуальной локальной сети. Предполагается, что конечный узел работает с непомеченными кадрами, то есть структура VLAN для него прозрачна.

Транк — это линия связи, которая соединяет между собой порты двух коммутаторов; в общем случае через транк передается трафик нескольких виртуальных сетей.

Коммутаторы, поддерживающие технику VLAN, без специального конфигурирования по умолчанию работают как стандартные коммутаторы, обеспечивая соединения всех со всеми. В сети, образованной такими коммутаторами, все конечные узлы по умолчанию относятся к условной сети VLAN1 с идентификатором VID, равным 1. Все порты этой сети, к которым подключены конечные узлы, по определению являются портами доступа. Сеть VLAN1 можно отнести к виртуальным локальным сетям лишь *условно*, так как по

ней передаются непомеченные кадры. Условная сеть VLAN также называется сетью VLAN, предлагаемой по умолчанию (default VLAN), или естественной (native VLAN).

Чтобы образовать в исходной сети виртуальную локальную сеть, нужно в первую очередь выбрать для нее значение идентификатора VID, отличное от 1, а затем, используя команды конфигурирования коммутатора, приписать к этой сети те порты, к которым присоединены включаемые в нее компьютеры. Порт доступа может быть приписан только к одной виртуальной локальной сети.

Порты доступа получают от конечных узлов сети непомеченные кадры, маркируя их тегом VLAN, содержащим то значение VID, которое назначено этому порту. При передаче же помеченных кадров конечному узлу порт доступа удаляет тег виртуальной локальной сети. Для более наглядного описания вернемся к рассмотренному ранее примеру сети. На рис. 71 показано, как решается задача избирательного доступа к серверам на основе техники VLAN.

Будем считать, что поставлена задача обеспечить доступ компьютеров C1 и C3 к серверам S1 и S3, в то время как компьютеры C2 и C4 должны иметь доступ только к серверам S2 и S4. Чтобы решить эту задачу, можно организовать две виртуальные локальные сети, VLAN2 и VLAN3 (напомним, что сеть VLAN1 уже существует по умолчанию — это наша исходная сеть), приписав один набор компьютеров и серверов к VLAN2, а другой — к VLAN3. Первым шагом в конфигурировании VLAN2 и VLAN3 является их активизация в каждом из коммутаторов сети.

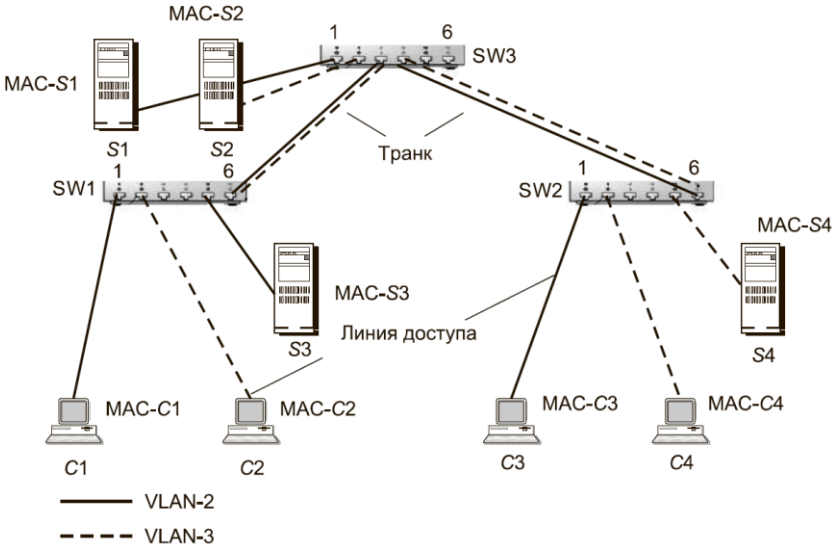


Рис. 71. Разбиение сети на две виртуальные локальные сети

Затем к этим сетям VLAN можно приписать определенные порты, работающие в режиме доступа. Для приписывания конечных узлов к определенной виртуальной локальной сети соответствующие порты объявляются портами доступа этой сети путем назначения им соответствующего идентификатора VID. Например, порт 1 коммутатора SW1 должен быть объявлен портом доступа VLAN2 путем назначения ему идентификатора VID2, то же самое должно быть проделано с портом 5 коммутатора SW1, портом 1 коммутатора SW2 и портом 1 коммутатора SW3. Порты доступа сети VLAN3 должны получить идентификатор VID3.

В нашей сети нужно также организовать транки — те линии связи, которые соединяют между собой порты коммутаторов. Порты, подключенные к транкам, не добавляют и не удаляют теги, а просто передают кадры в неизменном виде. В нашем примере такими портами должны быть порты 6 коммутаторов SW1 и SW2, а также порты 3 и 4 коммутатора SW3. Порты в нашем примере должны поддерживать сети VLAN2 и VLAN3 (и VLAN1, если в сети есть узлы, явно не приписанные ни к одной виртуальной локальной сети). Транк может быть сконфигурирован как в «неразборчивом» режиме, когда он передает кадры с любым номером VLAN, так и в избирательном режиме, когда он передает кадры только определенных номеров VLAN.

Коммутаторы, поддерживающие технологию VLAN, осуществляют дополнительную фильтрацию трафика. В том случае, если таблица продвижения коммутатора говорит о том, что пришедший кадр нужно передать на некоторый порт, перед передачей коммутатор проверяет, соответствует ли значение VID в теге VLAN кадра той виртуальной локальной сети, которая приписана к этому порту. В случае соответствия кадр передается, несоответствия — отбрасывается. Непомеченные кадры обрабатываются аналогичным образом, но с использованием условной сети VLAN1. MAC-адреса изучаются коммутаторами сети отдельно по каждой виртуальной локальной сети.

Архитектура Router-on-a-Stick

Router-on-a-stick (роутер на палочке) - это термин, часто используемый для описания архитектуры, состоящей из маршрутизатора и коммутатора, которые соединены с

использованием одного канала Ethernet, настроенного как 802.1Q транк (пример показан на рис. 72). Стандарт 802.1Q используется для тегирования трафика, для передачи информации о принадлежности к VLAN. В этой схеме на коммутаторе настроено несколько VLAN и маршрутизатор выполняет всю маршрутизацию между различными сетями или VLAN (Inter-VLAN routing).

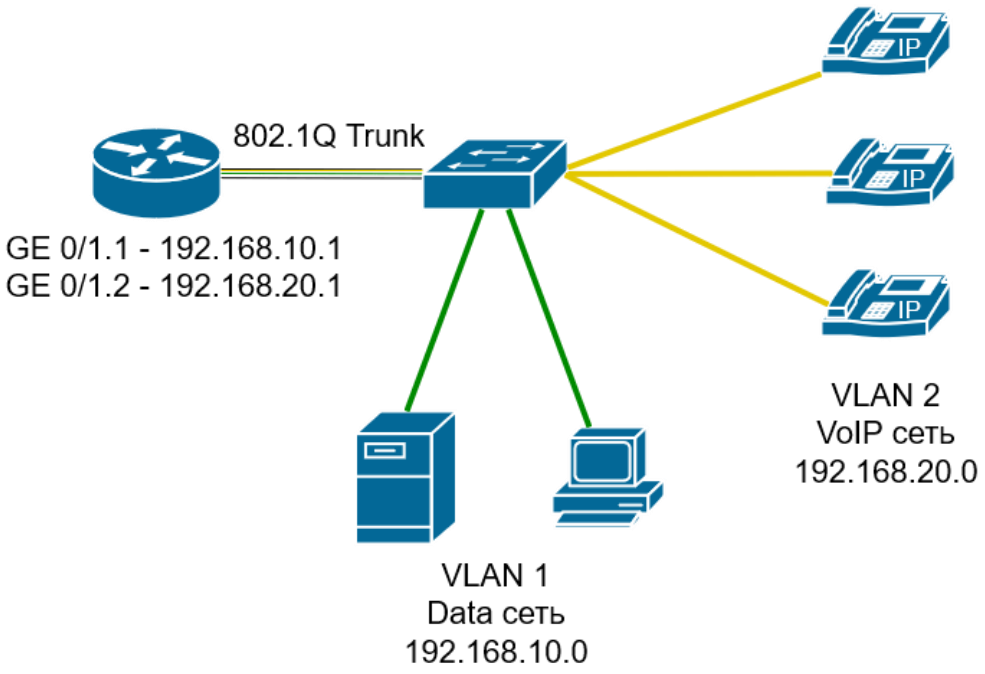


Рис. 72. Архитектура Router-on-a-Stick

Задание

Для изучения технологии VLAN построить локальную сеть, состоящую из двух сегментов, соединенных через центральный коммутатор (структура изображена на рис.72).

Алгоритм выполнения лабораторной работы

Работу выполнять в виртуальной среде эмулирования PNetLab.

При достаточной подготовке возможно выполнить работу на оборудовании лаборатории, используя коммутаторы AS1, AS3, AS4, AS5. В качестве оконечных устройств использовать подключенные к коммутаторам Зонды КМУТ М7.

1. Собрать схему связи (рис.72);
2. Настроить коммутатор. Для его настройки первым шагом является создание необходимых двух VLAN на коммутаторе и настройка их с IP-адресом. Поскольку все коммутаторы содержат VLAN1 (VLAN по умолчанию), нам нужно только создать VLAN2.

Пример настройки:

```
Switch# configure terminal
Switch(config)# vlan2
Switch(config-vlan)# name voice
Switch(config-vlan)# exit
Switch(config)# interface vlan1
Switch(config-if)# ip address 192.168.10.2 255.255.255.0
Switch(config-if)# exit
Switch(config)# interface vlan2
Switch(config-if)# ip address 192.168.20.2 255.255.255.0
Switch(config-if)# exit
```

Далее, нам нужно создать транк порт, который будет соединятся с маршрутизатором. Для этой цели мы выберем порт GigabitEthernet 0/1.

```
Switch# configure terminal
Switch(config)# interface gigabitethernet 0/1
Switch(config-if)# switchport trunk encapsulation dot1q
Switch(config-if)# switchport mode trunk
Switch(config-if)# spanning-tree portfast trunk
```

При помощи данных команд мы определили, что транк будет использовать инкапсуляцию 802.1Q, установили порт в режим транка и включили функцию portfast trunk spanning-tree, чтобы гарантировать, что порт будет пересылать пакеты немедленно при подключении к устройству, например, маршрутизатору. Внимание: команда spanning-tree portfast trunk не должна использоваться на портах, которые подключаются к другому коммутатору, чтобы избежать петель в сети.

3. Настроить маршрутизатор для обеспечения связью с коммутатором и позволить всему трафику VLAN проходить и маршрутизироваться по мере необходимости.

Создание транка на порте маршрутизатора не сильно отличается от процесса, описанного выше - хотя мы транк на одном физическом интерфейсе, мы должны создать подинтерфейс (**sub-interface**) для каждого VLAN.

Чтобы сформировать транк с нашим коммутатором, необходимо создать один подинтерфейс для каждого VLAN, сконфигурированного на нашем коммутаторе. После создания подинтерфейса мы назначаем ему IP-адрес и устанавливаем тип

инкапсуляции 802.1Q и указываем номер VLAN, к которому принадлежит подинтерфейс.

Например, команда encapsulation dot1q 2 определяет инкапсуляцию 802.1Q и устанавливает подинтерфейс на VLAN 2.

Параметр native который мы использовали для подинтерфейса gigabitethernet0/1.1, сообщает маршрутизатору, что нативный vlan - это VLAN 1. Это параметр по умолчанию на каждом коммутаторе Cisco и поэтому должен совпадать с маршрутизатором.

Пример настройки маршрутизатора:

```
Router# configure terminal
Router(config)# interface gigabitethernet0/1
Router(config-if)# no ip address
Router(config-if)# duplex auto
Router(config-if)# speed auto
Router(config-if)# interface gigabitethernet0/1.1
Router(config-subif)# encapsulation dot1q 1 native
Router(config-subif)# ip address 192.168.10.1 255.255.255.0
Router(config-subif)# interface gigabitethernet0/1.2
Router(config-subif)# encapsulation dot1q 2
Router(config-subif)# ip address 192.168.20.1 255.255.255.0
```

4. Для проверки правильности выполнения команд можно использовать на маршрутизаторе команду **show vlans**, где будут отображены созданные нами подинтерфейсы. Зафиксировать полученный результат.

5. Также при помощи команды **show ip route** в таблице маршрутизации мы должны увидеть наши подинтерфейсы. Зафиксировать полученный результат.

6. Настроить адресацию на оконечных устройствах.

7. Запустить пинг между оконечными устройствами в разных VLAN. Зафиксировать полученный результат.

Содержание отчета

1. Титульный лист;
2. Цель работы;
3. Теоретические сведения;
4. Описание выполнения работы (по шагам);
5. Полученные результаты (выводы команд, скриншоты);
6. Выводы по результатам выполнения работы.

Контрольные вопросы

1. Что такое VLAN и для чего предназначен?
2. Зачем нужно использовать VLAN, если можно просто разбить сеть на подсети на сетевом уровне?
3. Какой номер VLAN установлен на коммутаторах по умолчанию?
4. Что такое архитектура Router-on-a-Stick?

7.2.8. Изучение технологии Q-in-Q.

Ключевые слова

Технология Q-in-Q, Double VLAN.

Цель работы

Изучить технологию Q-in-Q на примере сети связи. Изучить настройки сетевого оборудования с такой технологией.

Краткие теоретические сведения

Определение Q-in-Q

Технология (функция) Q-in-Q, также известная как Double VLAN, 802.1Q Tunneling, VLAN Tunneling, соответствует стандарту IEEE 802.1ad, который является расширением стандарта IEEE 802.1Q (технология VLAN). Она позволяет добавлять в маркированные кадры Ethernet второй тег IEEE 802.1Q.

Инкапсуляция кадра Ethernet вторым тегом происходит следующим образом: тег, содержащий идентификатор VLAN сети провайдера SP-VLAN ID (от Service Provider VLAN ID — идентификатор сервиса провайдера VLAN, внешний тег) вставляется перед внутренним тегом, содержащим клиентский идентификатор VLAN – C-VLAN ID (Customer VLAN ID — идентификатор сервиса клиента VLAN, внутренний тег). Передача кадров в сети провайдера осуществляется только на основе внешнего тега SP-VLAN ID, внутренний тег пользовательской сети C-VLAN ID при этом скрыт.

Функции Q-in-Q

Существует две реализации функции Q-in-Q:

Port-based Q-in-Q и Selective Q-in-Q.

Функция Port-based Q-in-Q по умолчанию присваивает любому кадру, поступившему на порт доступа граничного коммутатора провайдера идентификатор SP-VLAN, равный идентификатору порта. Порт маркирует кадр независимо от того, является он маркированным или немаркированным. При поступлении маркированного кадра в него добавляется второй тег с идентификатором, равным SP-VLAN. Если на порт пришёл немаркированный кадр, в него добавляется только тег с SP-VLAN порта.

Функция Selective Q-in-Q является более гибкой по сравнению с Port-based Q-in-Q.

Она позволяет:

- маркировать кадры внешними тегами с различными идентификаторами SP-VLAN в зависимости от значений внутренних идентификаторов C-VLAN;
- задавать приоритеты обработки кадров внешних SP-VLAN на основе значений приоритетов внутренних пользовательских C-VLAN;
- добавлять к немаркированным пользовательским кадрам помимо внешнего тега SP-VLAN внутренний тег C-VLAN.

Все порты граничного коммутатора, на котором используется функция Port-based Q-in-Q или Selective Q-in-Q, должны быть настроены как порты доступа (UNI) или Uplink-порты (NNI):

- UNI (User-to-Network Interface) — эта роль назначается портам, через которые будет осуществляться взаимодействие граничного коммутатора провайдера с клиентскими сетями;
- NNI (Network-to-Network Interface) — эта роль назначается портам, которые подключаются к внутренней сети провайдера или другим граничным коммутаторам.

Функция Selective Q-in-Q позволяет добавлять в кадры различные внешние теги VLAN, основываясь на значениях внутренних тегов. Для этого на портах UNI граничного коммутатора необходимо задать правила соответствия идентификаторов C-VLAN идентификаторам SP-VLAN (vlan translation).

Область применения технологии Q-in-Q

Основной областью применения технологии Q-in-Q являются сети передачи данных операторского класса. В первую очередь, применение данной технологии предназначено для прозрачного пропуска трафика клиентов, использующих в своих сетях VLAN через сеть оператора.

Преимущества технологии Q-in-Q

Преимуществом применения технологии Q-in-Q для такой услуги является отсутствие необходимости согласования тегов VLAN между клиентом и оператором, что в свою очередь предоставляет клиенту практически безграничные возможности по

расширению своей сети передачи данных, в частности не ограничивает клиента в количестве VLAN.

Selective Q-in-Q предоставляет возможность избирательного добавления Outer VLAN (внешней VLAN) тега в зависимости от тега Inner VLAN (внутренней VLAN). Основной областью применения Selective Q-in-Q является организация в сети передачи данных схемы предоставления услуг операторского класса 'VLAN на сервис'.

Пример настройки классического варианта Q-in-Q

На схеме (рис. 73) CE1 и CE2 - коммутаторы клиента, а P, PE1 и PE2 - коммутаторы оператора. Предположим, что необходимо организовать передачу тегированных и нетегированных кадров между двумя офисами клиента через сеть передачи данных оператора. В данном примере нам нет необходимости выяснять используемые в сети клиента теги VLAN - все кадры со стороны сети клиента будут дополнительно маркироваться тегом Outer VLAN.

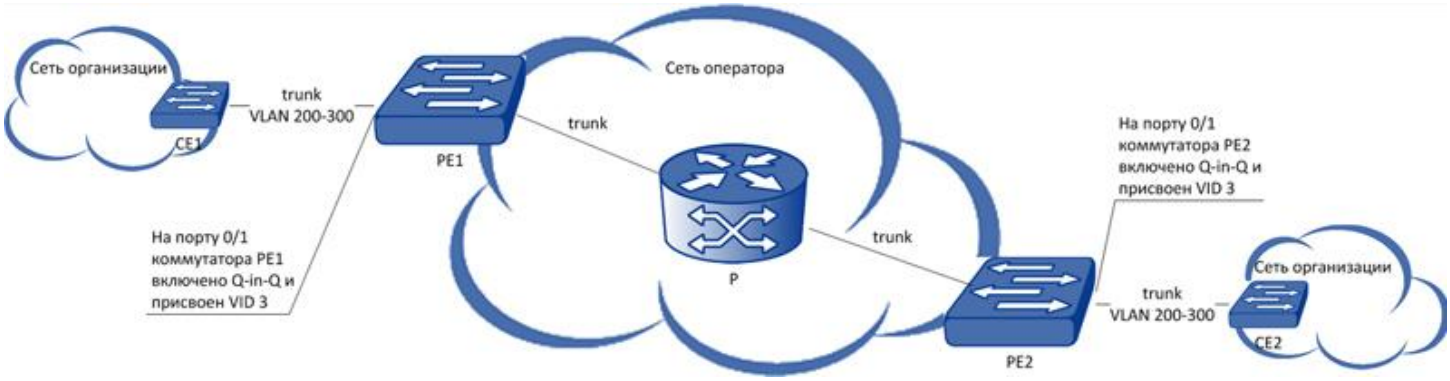


Рис. 73. Схема связи с использованием Q-in-Q

Задание

Для изучения технологии Q-in-Q построить локальную сеть, состоящую из двух клиентов. Каждый из них использует свою нумерацию VLAN. Необходимо обеспечить передачу маркированного трафика через сеть провайдера. Выполнить задание использовать функции Port-Based Q-in-Q VLAN, которая позволяет добавить в передаваемый в сети провайдера трафик уникальный идентификатор VLAN, присвоенный клиенту.

Алгоритм выполнения лабораторной работы

Работу выполнять в виртуальной среде эмулирования PNetLab.

При достаточной подготовке возможно выполнить работу на оборудовании лаборатории, используя коммутаторы КД1, КД2, КД3. В качестве оконечных устройств использовать подключенные к коммутаторам Зонды КМУТ М7.

1. Собрать схему связи (рис.49);

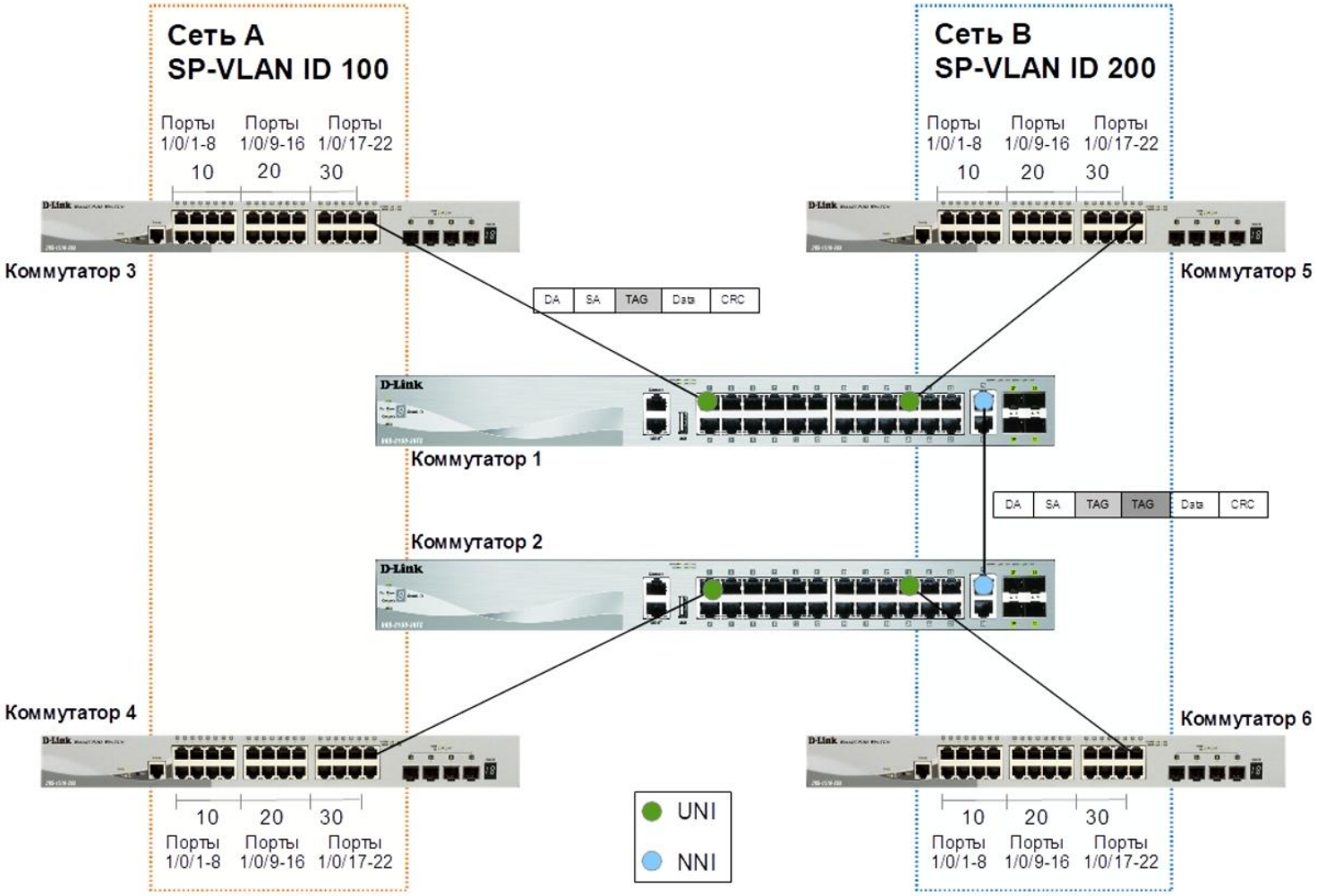


Рис. 49. Схема связи

2. Настроить коммутатор 1 и 2.

- Создать SP-VLAN 100 и 200:

```
Switch# configure terminal
Switch(config)# vlan 100
Switch(config-vlan)# exit
Switch(config)# vlan 200
Switch(config-vlan)# exit
```

- Настроить UNI-порты в SP-VLAN:

```
Switch(config)# interface ethernet 1/0/1
Switch(config-if)# switchport mode dot1q-tunnel
Switch(config-if)# switchport access vlan 100
Switch(config-if)# no vlan mapping miss drop
Switch(config-if)# exit
```

```
Switch(config)# interface range ethernet 1/0/19
Switch(config-if)# switchport mode dot1q-tunnel
Switch(config-if)# switchport access vlan 200
Switch(config-if)# no vlan mapping miss drop
Switch(config-if)# exit
```

- Настроить NNI-порты в SP-VLAN:

```
Switch(config)# interface range ethernet 1/0/25
Switch(config-if)# switchport mode trunk
Switch(config-if)# switchport trunk allowed vlan 100
Switch(config-if)# switchport trunk allowed vlan add 200
Switch(config-if)# dot1q tunneling ethertype 0x88a8
Switch(config-if)# end
```

3. Настроить коммутатор 3, 4, 5 и 6.

- Создать новые VLAN и добавить в них маркированные и немаркированные порты:

```
Switch# configure terminal
Switch(config)# vlan 10,20,30
Switch(config-vlan)#exit
Switch(config)# interface range ethernet 1/0/1-8
Switch(config-if-range)# switchport mode access
Switch(config-if-range)# switchport access vlan 10
Switch(config-if-range)# exit
Switch(config)# interface range ethernet 1/0/9-16
Switch(config-if-range)# switchport mode access
Switch(config-if-range)# switchport access vlan 20
Switch(config-if-range)# exit
Switch(config)# interface range ethernet 1/0/17-22
Switch(config-if-range)# switchport mode access
Switch(config-if-range)# switchport access vlan 30
Switch(config-if-range)# exit
Switch(config)#interface ethernet 1/0/23
Switch(config-if)#switchport mode trunk
Switch(config-if)#end
```

4. Проверить настройку функции Q-in-Q VLAN можно командой show dot1q-tunnel. Зафиксировать полученный результат.

5. Проверить настройки VLAN можно командой show vlan. Зафиксировать полученный результат.

6. Подключить к коммутаторам оконечные устройства и настроить адресацию на них.

7. Запустить пинг между оконечными устройствами в разных VLAN. Зафиксировать полученный результат.

Содержание отчета

1. Титульный лист;
2. Цель работы;
3. Теоретические сведения;
4. Описание выполнения работы (по шагам);
5. Полученные результаты (выводы команд, скриншоты);
6. Выводы по результатам выполнения работы.

Контрольные вопросы

1. Что такое Q-in-Q?
2. Назовите преимущества технологии Q-in-Q?
3. Назовите область применения технологии Q-in-Q?
4. Чем отличаются Outer VLAN и Inner VLAN?
5. Чем отличаются порты коммутатора UNI и NNI?

7.2.9. Изучение возможности снятия данных по протоколу SNMP.

Ключевые слова

Протокол SNMP, база данных MIB.

Цель работы

Изучить протокол SNMP.

Краткие теоретические сведения

Протокол SNMP (Simple Management Network Protocol — простой протокол сетевого администрирования) используется в системах сетевого управления для контроля подключённых к сети устройств на предмет условий, которые требуют внимания сетевого администратора.

Протокол SNMP относится к прикладному уровню стека TCP/IP. Для транспортировки своих сообщений он использует дейтаграммный транспортный протокол UDP, который, как известно, не обеспечивает надежную доставку. Протокол TCP, организующий надежную передачу сообщений на основе соединений, весьма загружает управляемые устройства, которые на момент разработки протокола SNMP были не очень мощные, поэтому от услуг протокола TCP было решено отказаться. SNMP — это протокол типа «запрос-ответ», то есть на каждый запрос, поступивший от менеджера, агент должен передать ответ, то есть агент работает с менеджером в режиме опроса. Особенностью протокола является его чрезвычайная простота — он включает в себя всего несколько команд:

- Команда GetRequest используется менеджером для запроса агента о значении какой-либо переменной по ее стандартному имени;
- Команда GetNextRequest применяется менеджером для извлечения значения следующего объекта (без указания его имени) при последовательном просмотре таблицы объектов;
- С помощью команды Response SNMP-агент передает менеджеру ответ на команду GetRequest или GetNextRequest;
- Команда SetRequest позволяет менеджеру изменять значения какой-либо переменной или списка переменных. С помощью команды SetRequest и происходит собственно управление

устройством. Агент должен «понимать» смысл значений переменной, которая используется для управления устройством, и на основании этих значений выполнять реальное управляющее воздействие — отключить порт, приписать порт определенной линии VLAN и т.п. Команда SetRequest пригодна также для задания условия, при выполнении которого SNMP-агент должен послать менеджеру соответствующее сообщение. Таким образом, может быть определена реакция на такие события, как инициализация агента, рестарт агента, обрыв связи, восстановление связи, неверная аутентификация и потеря ближайшего маршрутизатора. Если происходит любое из этих событий, то агент посылает менеджеру сообщение по своей инициативе, используя команду Trap. В этом случае агент уже не работает в режиме опроса, но он используется только для очень ограниченного набора аварийных ситуаций и не является основным;

- Команда GetBulk позволяет менеджеру получить несколько переменных за один запрос.

SNMP-сообщения, в отличие от сообщений многих других коммуникационных протоколов, не имеют заголовков с фиксированными полями. Любое SNMP-сообщение состоит из трех основных частей: версии протокола, общей строки и области данных.

Общая строка (community string) используется для группирования устройств, управляемых определенным менеджером. Общая строка является своего рода паролем, так как для того, чтобы устройства могли взаимодействовать по протоколу SNMP, они должны иметь одно и то же значение этого идентификатора (по умолчанию часто употребляется строка «public»). Однако этот механизм служит скорее для «распознавания» партнеров, нежели для безопасности.

В области данных содержатся описанные команды протокола, а также имена объектов и их значения. Область данных состоит из одного или более блоков, каждый из которых может относиться к одному из перечисленных типов команд протокола SNMP. Для каждого типа команды определен свой формат. Например, формат блока, относящегося к команде GetRequest, включает следующие поля:

- идентификатор запроса;

- статус ошибки (есть или нет);
- индекс ошибки (тип ошибки, если она есть);
- список имен объектов SNMP MIB (Management Information Base), включенных в запрос.

База данных MIB

База данных MIB содержит значения множества различных типов переменных, характеризующих конкретный управляемый объект. В самой первой версии стандарта (MIB v1) для характеристики устройства предлагалось использовать 114 типов переменных. Эти переменные организованы в виде дерева. Из корня выходит 8 ветвей, соответствующих следующим восьми группам переменных:

- System — общие данные об устройстве (например, идентификатор поставщика, время последней инициализации системы);
- Interfaces — параметры сетевых интерфейсов устройства (например, их количество, типы, скорости обмена, максимальный размер пакета);
- Address Translation Table — описание соответствия между сетевыми и физическими адресами (например, по протоколу ARP);
- Internet Protocol — данные, относящиеся к протоколу IP (адреса IP-шлюзов, хостов, статистика об IP-пакетах);
- ICMP — данные, относящиеся к протоколу ICMP;
- TCP — данные, относящиеся к протоколу TCP (число переданных, принятых и ошибочных TCP-сообщений);
- UDP — данные, относящиеся к протоколу UDP (число переданных, принятых и ошибочных UPD-дейтаграмм);
- EGP — данные, относящиеся к протоколу EGP (число принятых с ошибками и без ошибок сообщений).

Каждая группа характеристик образует отдельное поддерево.

Версия SNMP MIB 2 позволяет строить более гибкие модели управляемых устройств за счет новых типов данных (например, 64-битных указателей), поддерживает режим групповой передачи данных (нужный для передачи табличных данных), а также улучшает безопасность систем управления за счет поддержки нескольких механизмов аутентификации вместо примитивного механизма аутентификации общей строки в SNMP MIB v1.

Модели SNMP MIB описываются и используются менеджерами и управляемыми устройствами в виде текстовых файлов, эти файлы можно читать любым текстовым редактором. Для понимания функциональности некоторой MIB нужно иметь в виду, что ее текст следует синтаксису документа «Структура управляющей информации» (Structure of Management Information Version 2, SMIv2, RFC 2578).

Для учета специфики конкретного сетевого устройства — маршрутизатора, коммутатора, модема и т. д. — производители часто выпускают свои собственные, фирменные MIB, которые расширяют возможности сбора данных о состоянии этого устройства.

Задание

Выполнить получение некоторых характеристик сетевого оборудования по протоколу SNMP.

Алгоритм выполнения лабораторной работы

1. Выполнить вход по протоколу SSH на Зонд агрегации;
2. Проверить работу протокола командой sudo systemctl status snmpd. При успешной работе будет выведен результат со статусом **active (running)**.

```
 # sudo systemctl status snmpd
 ● snmpd.service - Simple Network Management Protocol (SNMP) Daemon.
    Loaded: loaded (/lib/systemd/system/snmpd.service; enabled; preset:
enabled)
    Active: active (running) since Mon 2025-08-18 08:25:44 UTC; 1 day
2h ago
          Main PID: 1628 (snmpd)
```

3. Выполнить получение списка интерфейсов с Зонда КМУТ (сниффер), выполнив следующую команду:

```
# snmpwalk -v 2c -c public 198.18.10.5 1.3.6.1.2.1.31.1.1.1.1
```

4. Выполнить получение времени работы с Зонда КМУТ, выполнив следующую команду:

```
# snmpwalk -v 2c -c public 198.18.10.5 1.3.6.1.2.1.31.1.1.1.1
```

5. Выполнить получение загрузки процессора за одну минуту с Зонда КМУТ, выполнив следующую команду:

```
# snmpwalk -v 2c -c public 198.18.10.5 1.3.6.1.4.1.2021.10.1.3.1
```

6. Вместо IP-адреса 198.18.10.5 выбрать другое устройство и получить аналогичные характеристики;
7. Зафиксировать полученные результаты.

Содержание отчета

1. Титульный лист;
2. Цель работы;
3. Теоретические сведения;
4. Описание выполнения работы (по шагам);
5. Полученные результаты (выводы команд, скриншоты);
6. Выводы по результатам выполнения работы.

Контрольные вопросы

7. На каком уровне стека TCP/IP работает протокол SNMP?
8. Зачем используется протокол SNMP?
9. Какие команды используются в протоколе SNMP?
10. Что содержится в базе данных MIB?

7.2.10. Изучение протокола туннелирования GRE.

Ключевые слова

Протокол туннелирования GRE.

Цель работы

Изучить протокол туннелирования GRE. Построить тоннель между двумя маршрутизаторами.

Краткие теоретические сведения

Generic Routing Encapsulation (GRE) — это протокол туннелирования, разработанный компанией Cisco, который позволяет инкапсулировать широкий спектр протоколов сетевого уровня в point-to-point каналах (точка-точка).

Туннель GRE используется, когда пакеты должны быть отправлены из одной сети в другую через Интернет или незащищенную сеть. В GRE виртуальный туннель создается между двумя конечными точками (маршрутизаторами Cisco), а пакеты отправляются через туннель GRE.

Важно отметить, что пакеты, проходящие внутри туннеля GRE, не шифруются, поскольку GRE не шифрует туннель, а инкапсулирует его с заголовком GRE. Если требуется защита данных, дополнительно должен быть настроен протокол IPSec для обеспечения конфиденциальности данных - тогда GRE-туннель преобразуется в безопасный туннель GRE.

На рис. 74 показана процедура инкапсуляции простого незащищенного пакета GRE, проходящего через маршрутизатор и входящего в туннельный интерфейс.

Рис. 74. Процедура инкапсуляции простого незащищенного пакета GRE

Туннель GRE использует интерфейс «туннель» - логический интерфейс, настроенный на маршрутизаторе с IP-адресом, где пакеты инкапсулируются и декапсулируются при входе или выходе из туннеля GRE (рис. 75).

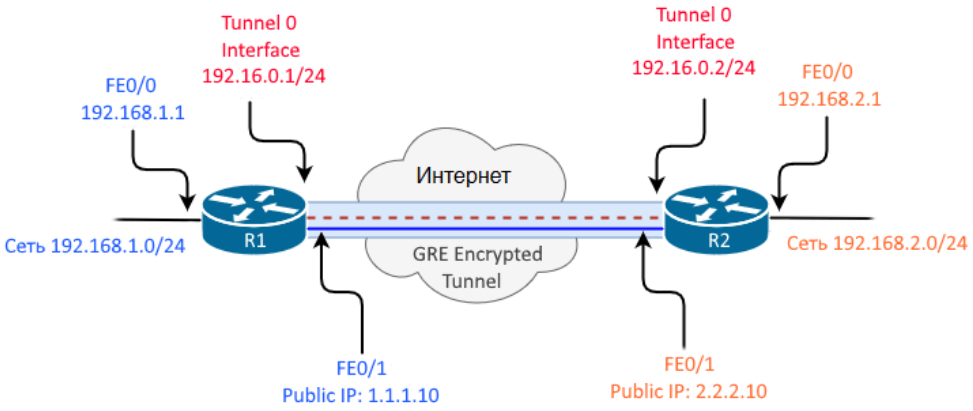


Рис. 75. Процедура инкапсуляции простого незащищенного пакета GRE

Задание

Выполнить построение GRE тоннеля между двумя маршрутизаторами.

Алгоритм выполнения лабораторной работы

Работу выполнять в виртуальной среде эмулирования PNetLab.

При достаточной подготовке возможно выполнить работу на оборудовании лаборатории, используя маршрутизаторы Cisco 2801.

1. Собрать схему связи (рис.51);
2. Создать туннельный интерфейс на R1:

```
R1(config)# interface Tunnel0
R1(config-if)# ip address 172.16.0.1 255.255.255.0
R1(config-if)# ip mtu 1400
R1(config-if)# ip tcp adjust-mss 1360
R1(config-if)# tunnel source 1.1.1.10
R1(config-if)# tunnel destination 2.2.2.10
```

Все туннельные интерфейсы участвующих маршрутизаторов всегда должны быть настроены с IP-адресом, который не используется где-либо еще в сети. Каждому туннельному интерфейсу назначается IP-адрес в той же сети, что и другим туннельным интерфейсам.

В нашем примере оба туннельных интерфейса являются частью сети 172.16.0.0/24.

Поскольку GRE является протоколом инкапсуляции, мы устанавливаем максимальную единицу передачи (MTU - Maximum Transfer Unit) до 1400 байт, а максимальный размер сегмента (MSS - Maximum Segment Size) - до 1360 байт. Поскольку большинство транспортных MTU имеют размер 1500 байт и у нас есть дополнительные издержки из-за GRE, мы должны уменьшить MTU для учета дополнительных служебных данных. Установка 1400 является обычной практикой и гарантирует, что ненужная фрагментация пакетов будет сведена к минимуму.

В заключение мы определяем туннельный источник, который является публичным IP-адресом R1, и пункт назначения - публичный IP-адрес R2.

3. Как только мы завершим настройку R1, маршрутизатор подтвердит создание туннеля и сообщит о его состоянии:

```
R1#
*September 21 16:33:27.321: %LINEPROTO-5-UPDOWN: Line protocol on
Interface Tunnel0, changed state to up
```

Поскольку интерфейс Tunnel 0 является логическим интерфейсом, он останется включенным, даже если туннель GRE не настроен или не подключен на другом конце.

4. Далее создадим интерфейс Tunnel 0 на R2:

```
R2(config)# interface Tunnel0
R2(config-if)# ip address 172.16.0.2 255.255.255.0
R2(config-if)# ip mtu 1400
R2(config-if)# ip tcp adjust-mss 1360
R2(config-if)# tunnel source 2.2.2.10
R2(config-if)# tunnel destination 1.1.1.10
```

Интерфейс туннеля R2 настроен с соответствующим IP-адресом источника и назначения туннеля. Как и в случае с R1, маршрутизатор R2 сообщит нам, что интерфейс Tunnel0 работает:

```
R2#
*May 21 16:45:30.442: %LINEPROTO-5-UPDOWN: Line protocol on Interface
Tunnel0, changed state to up
```

5. Проверить командой ping доступность R2 с узла R1.

```
R1# ping 172.16.0.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.0.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

6. Теперь две конечные точки туннеля могут видеть друг друга. Оконечные устройства, подключенные к маршрутизатору, в любой сети по-прежнему не смогут достичь другой стороны, если на каждой конечной точке не установлен статический маршрут. На R1 мы добавляем статический маршрут к удаленной сети 192.168.2.0/24 через 172.16.0.2, который является другим концом нашего туннеля GRE. Когда R1 получает пакет для сети 192.168.2.0, он теперь знает, что следующим переходом является 172.16.0.2, и поэтому отправит его через туннель:

```
R1(config)# ip route 192.168.2.0 255.255.255.0 172.16.0.2
```

7. Та же конфигурация должна быть повторена для R2:

```
R2(config)# ip route 192.168.1.0 255.255.255.0 172.16.0.1 172.16.0.2
```

8. Подключить к маршрутизаторам оконечные устройства и настроить адресацию на них.

9. Запустить пинг между оконечными устройствами, подключенные к разным маршрутизаторам. Зафиксировать полученный результат.

Содержание отчета

1. Титульный лист;
2. Цель работы;
3. Теоретические сведения;
4. Описание выполнения работы (по шагам);
5. Полученные результаты (выводы команд, скриншоты);
6. Выводы по результатам выполнения работы.

Контрольные вопросы

1. Какие заголовки добавляются к пакету в процессе инкапсуляции по протоколу GRE?
2. Назовите недостаток использования протокола GRE без использования IPSec?

7.2.11. Изучение управления трафиком по протоколу BGP.

Ключевые слова

Динамическая маршрутизация, протокол BGP, резервирование каналов связи.

Цель работы

Научиться настраивать протокол BGP. Изучить функцию автоматического перенаправления трафика из основного канала связи в резервный и обратно по протоколу BGP.

Краткие теоретические сведения

Определение динамической маршрутизации

При динамической маршрутизации все изменения конфигурации сети автоматически отражаются в таблицах маршрутизации благодаря протоколам маршрутизации. Эти протоколы собирают информацию о топологии связей в сети, что позволяет им оперативно отражать все текущие изменения. В таблицах маршрутизации при адаптивной маршрутизации обычно имеется информация об интервале времени, в течение которого данный маршрут будет оставаться действительным. Это время называют временем жизни (TTL) маршрута. Если по истечении времени жизни существование маршрута не подтверждается протоколом маршрутизации, то он считается нерабочим, и пакеты по нему больше не посылаются. Протоколы адаптивной маршрутизации бывают распределенными и централизованными. При распределенном подходе все маршрутизаторы сети находятся в равных условиях, они находят маршруты и строят собственные таблицы маршрутизации, работая в тесной кооперации друг с другом, постоянно обмениваясь информацией о конфигурации сети. При централизованном подходе в сети существует один выделенный маршрутизатор, собирающий всю информацию о топологии и состоянии сети от других маршрутизаторов. На основании этих данных выделенный маршрутизатор (иногда называемый сервером маршрутов) строит таблицы маршрутизации для остальных маршрутизаторов сети, распространяя их затем по сети, чтобы каждый маршрутизатор получил собственную таблицу и в дальнейшем самостоятельно принимал решение о продвижении каждого пакета.

Определение BGP

Протокол **BGP (Border Gateway Protocol)** — это основной протокол динамической маршрутизации, который используется в Интернете.

Маршрутизаторы, использующие протокол BGP, обмениваются информацией о доступности сетей. Вместе с информацией о сетях передаются различные атрибуты этих сетей, с помощью которых BGP выбирает лучший маршрут и настраиваются политики маршрутизации.

Один из основных атрибутов, который передается с информацией о маршруте — это список автономных систем, через которые прошла эта информация. Эта информация позволяет BGP определять, где находится сеть относительно автономных систем, исключать петли маршрутизации, а также может быть использована при настройке политик.

Маршрутизация осуществляется пошагово от одной автономной системы к другой. Все политики BGP настраиваются, в основном, по отношению к внешним/соседним автономным системам. То есть описываются правила взаимодействия с ними.

Так как BGP оперирует большими объемами данных (текущий размер таблицы для IPv4 более 450 тысяч маршрутов), то принципы его настройки и работы отличаются от внутренних протоколов динамической маршрутизации – IGP.Что такое акввввв

Терминология протокола BGP

- Внутренний протокол маршрутизации (interior gateway protocol, IGP) – протокол, который используется для передачи информации о маршрутах внутри автономной системы.
- Внешний протокол маршрутизации (exterior gateway protocol, EGP) – протокол, который используется для передачи информации о маршрутах между автономными системами.
- Автономная система (autonomous system, AS) — набор маршрутизаторов, имеющих единые правила маршрутизации, управляемых одной технической администрацией и работающих на одном из протоколов IGP (для внутренней маршрутизации AS может использовать и несколько IGP).

- Транзитная автономная система (transit AS) — автономная система, через которую передается трафик других автономных систем.
- Путь (path) — последовательность, состоящая из номеров автономных систем, через которые нужно пройти для достижения сети назначения.
- Атрибуты пути (path attributes, PA) — характеристики пути, которые позволяют выбрать лучший путь.
- BGP speaker — маршрутизатор, на котором работает протокол BGP.
- Соседи (neighbor, peer) — любые два маршрутизатора, между которыми открыто TCP-соединение для обмена информацией о маршрутизации.
- Информация сетевого уровня о доступности сети (Network Layer Reachability Information, NLRI) — IP-префикс и длина префикса.

Описание протокола BGP

BGP выбирает лучшие маршруты не на основании технических характеристик пути (пропускной способности, задержки и т.п.), а на основании политик. В локальных сетях наибольшее значение имеет скорость сходимости сети, время реагирования на изменения. И маршрутизаторы, которые используют внутренние протоколы динамической маршрутизации, при выборе маршрута, как правило, сравнивают какие-то технические характеристики пути, например, пропускную способность линков.

При выборе между каналами двух провайдеров, зачастую имеет значение не то, у какого канала лучше технические характеристики, а какие-то внутренние правила компании. Например, использование какого канала обходится компании дешевле. Поэтому в BGP выбор лучшего маршрута осуществляется на основании политик, которые настраиваются с использованием фильтров, анонсирования маршрутов, и изменения атрибутов.

Как и другие протоколы динамической маршрутизации, BGP может передавать трафик только на основании IP-адреса получателя. Это значит, что с помощью BGP нет возможности настроить правила маршрутизации, в которых будет учитываться, например, то, из какой сети был отправлен пакет или данные какого

приложения передаются. Если принимать решение о том, как должен маршрутизироваться пакет, необходимо по каким-то дополнительным критериям, кроме адреса получателя, необходимо использовать механизм policy-based routing (PBR).

Типы сообщений протокола BGP

Для обмена информацией маршрутизатор создаёт сессии с соседними устройствами. Сессии работают поверх протоколов TCP/IP через порт 179, а их установление происходит в несколько последовательных стадий (рис. 76).

Idle — начальное состояние маршрутизатора, когда он готов установить соединение, но пока не выполняет активных действий. В это состояние маршрутизатор переходит при первом запуске или перезапуске протокола BGP — например, когда администратор сбрасывает все сессии для внесения изменений в конфигурацию.

Connect — на этом этапе маршрутизатор пытается установить TCP/IP-соединение с соседним устройством. Например, маршрутизатор с IP-адресом 192.0.2.1 пытается подключиться к соседу с IP-адресом 192.0.2.2. После успешного соединения маршрутизатор переходит в состояние OpenSent. Если соединение не удаётся (из-за закрытого порта или недоступности соседа), маршрутизатор переходит в состояние Active.

Active — в этом состоянии маршрутизатор пытается восстановить разорванное соединение с соседом. Он периодически отправляет TCP-запросы на подключение, пока не получит ответ или не истечёт время ожидания. При получении ответа маршрутизатор переходит в состояние Connect и пытается повторно установить соединение. Если время ожидания истекло, маршрутизатор возвращается в состояние Idle.

OpenSent — после перехода в это состояние маршрутизатор отправляет OPEN-сообщение соседу и ждёт ответа. В полученном OPEN-сообщении он проверяет версию BGP, номер автономной системы и другие параметры, а также определяет тип соединения. Если параметры не соответствуют локальным настройкам, соединение разрывается и маршрутизатор возвращается в состояние Idle. При корректных параметрах маршрутизатор переходит в состояние OpenConfirm.

OpenConfirm — на этом этапе маршрутизатор, отправив OPEN-сообщение с настройками и ожидает KEEPALIVE-сообщения от соседа. Это короткое сообщение подтверждает принятие переданных настроек. После его получения маршрутизатор переходит в состояние Established.

Established — это рабочее состояние сессии, когда маршрутизаторы установили соединение и обмениваются маршрутной информацией. Для поддержания соединения они обмениваются обновлениями маршрутов и регулярными KEEPALIVE-сообщениями. Если маршрутизатор не получает KEEPALIVE-сообщение в установленный период, соединение считается разорванным и сессия возвращается в состояние Idle.

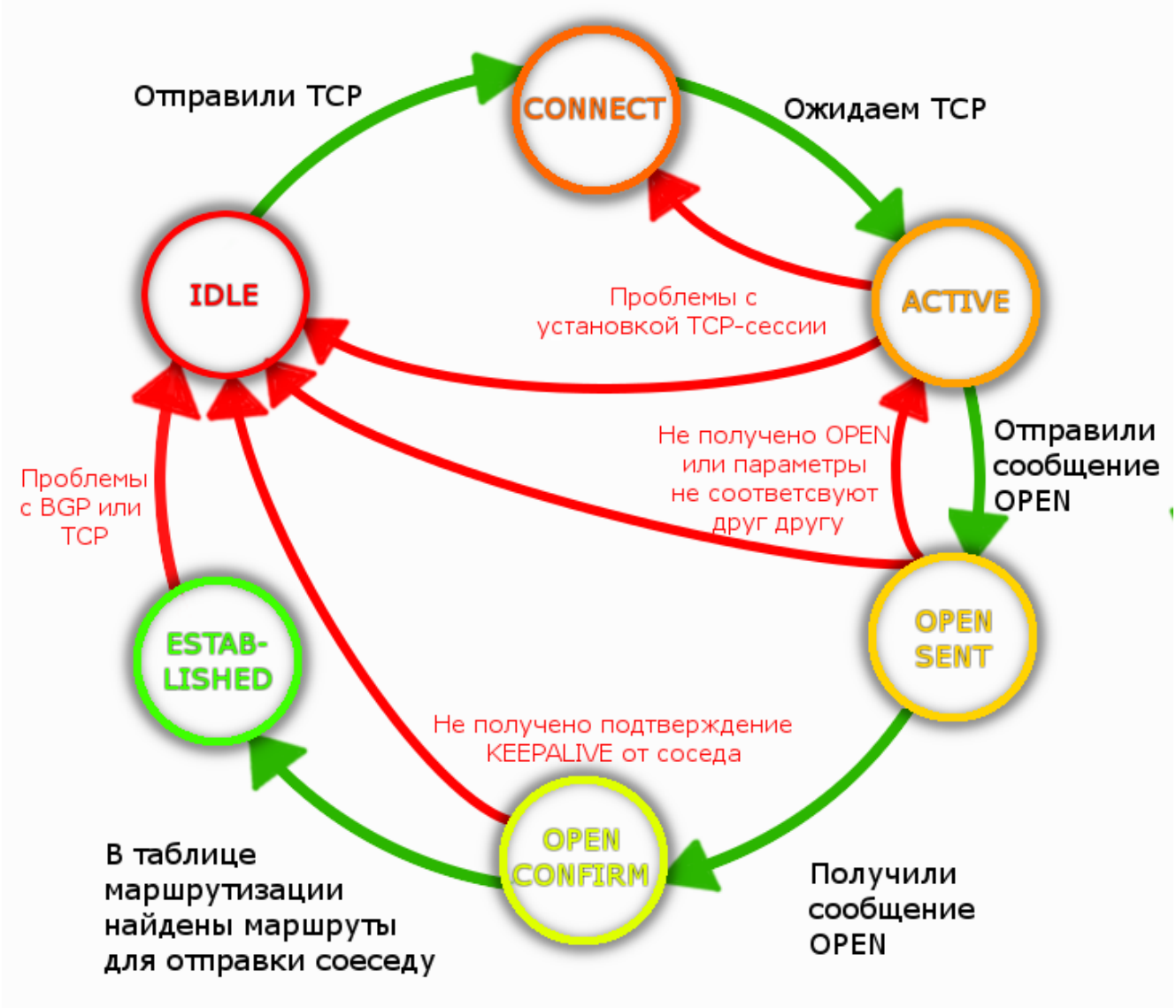


Рис. 76. Типы сообщений протокола BGP

Проверка состояния BGP-соседей

Для проверки состояния BGP-соседей на оборудовании лаборатории можно зайти на Зонд агрегации с правами администратора. Выполнить команду vtysh:

```
root@ZA-M11:~# vtysh

Hello, this is FRRouting (version 10.2).
Copyright 1996-2005 Kunihiro Ishiguro, et al.
```

Далее ввести команду show ip bgp summary:

```
ZA-M11# show ip bgp summary

IPv4 Unicast Summary:
BGP router identifier 198.18.20.1, local AS number 65000 VRF default vrf-id 0
BGP table version 12
RIB entries 10, using 1280 bytes of memory
Peers 6, using 143 KiB of memory

Neighbor        V         AS   MsgRcvd   MsgSent   TblVer  InQ OutQ  Up/Down State/PfxRcd   PfxSnt Desc
198.18.10.2     4      65001      3215      3215       12    0    0 2d05h31m            1        2 N/A
198.18.10.4     4      65002      3298      3303       12    0    0 2d06h53m            1        2 N/A
198.18.10.10    4      65003      9240      9242       12    0    0 3d07h21m            1        2 N/A
198.18.20.2     4      65001         0      1575        0    0    0    never       Active        0 N/A
198.18.20.4     4      65002         0      1636        0    0    0    never       Active        0 N/A
198.18.20.10    4      65004      9235      9233       12    0    0 3d07h21m            1        2 N/A

Total number of neighbors 6
```

Задание

Собрать схему для изучения функции автоматического перенаправления трафика из основного канала связи в резервный и обратно по протоколу динамической маршрутизации BGP.

Алгоритм выполнения лабораторной работы

Работу выполнять в виртуальной среде эмулирования PNetLab.

При достаточной подготовке возможно выполнить работу на оборудовании лаборатории, используя маршрутизаторы Cisco 2801, Зонд агрегации и Зондах CPE1 и CPE2.

1. Собрать схему сети (рис. 77). В этой схеме соединение между маршрутизаторами A и C было основным для входящего и исходящего трафика, а соединение между B и D было резервным и использовалось только при аварии основного.

2. Сконфигурировать интерфейсы в соответствии с таблицей:

interfejs/router	Router A	Router B	Router C	Router D
Serial 1/0	150.0.0.9/30	150.0.0.10/30	150.0.0.13/30	150.0.0.14/30
Serial 1/1	150.0.0.1/30	150.0.0.5/30	150.0.0.2/30	150.0.0.6/30

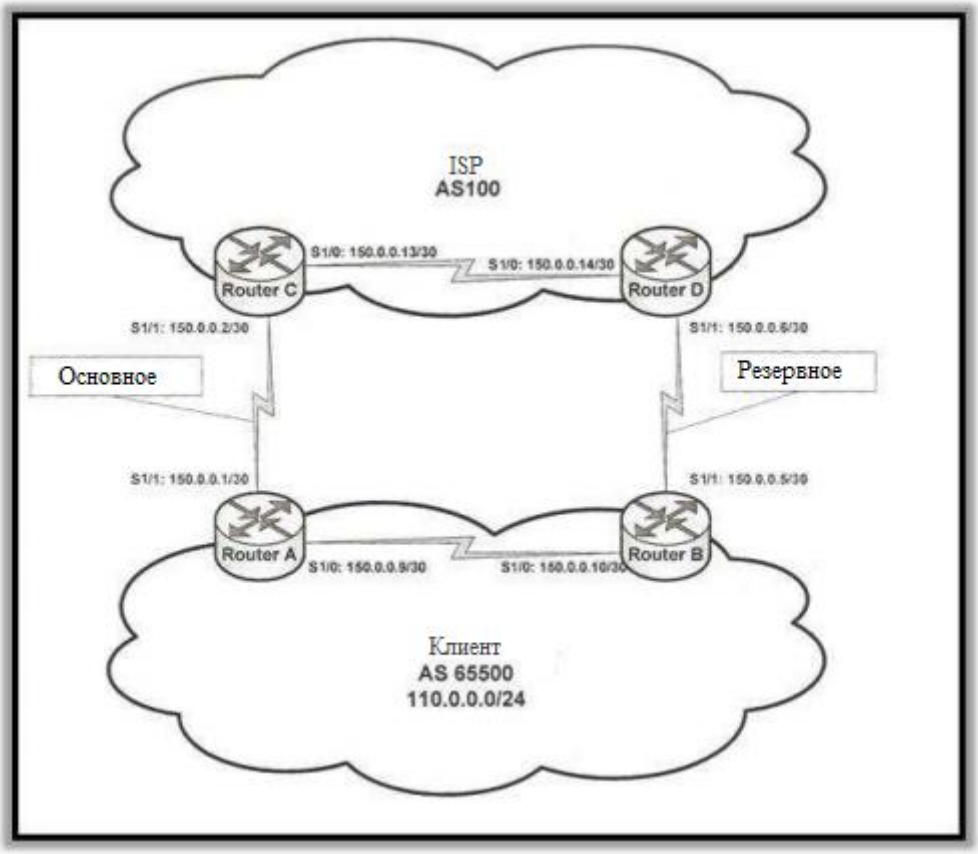


Рис. 77. Схема сети для изучения протокола BGP

Для Serial 1/1 маршрутизатора A:

```
R1(config)# interface S1/1
R1(config-if)# ip address 150.0.0.1 255.255.255.252
R1(config-if)# no shutdown
```

Остальные интерфейсы следует сконфигурировать самостоятельно аналогичным способом.

3. Настроить номера AS. Каждый маршрутизатор должен иметь номер ASN (в соответствии со схемой), который определяет к какому AS принадлежит маршрутизатор. Плюс каждый маршрутизатор будет обслуживать две сессии: одна iBGP с маршрутизатором в той же AS для работы внутри автономной системы и одна eBGP с маршрутизатором находящимся в другой AS для работы между автономными системами. Сессии должны быть сконфигурированы по обе стороны соединения.

```
R1(config)# router bgp 65500
R1(config-router)# neighbor 150.0.0.10 remote-as 65500
R1(config-router)# neighbor 150.0.0.2 remote-as 100
```

Конфигурация остальных маршрутизаторов остается самостоятельным заданием.

4. Далее воспользуемся нестандартным механизмом next-hop-self. Он позволяет на изменения адреса Next_Hop перед анонсированием другому маршрутизатору, находящемуся в той же AS. Маршрутизатор изменяет значение атрибута Next_Hop на свой собственный адрес. Благодаря чему мы не должны конфигурировать внутридоменную маршрутизацию в каждой AS. Механизм этот должен быть сконфигурирован только между маршрутизаторами в одной и той же AS.

```
R1(config)# router bgp 65500
R1(config-router)# neighbor 150.0.0.10 next-hop-self
```

Конфигурация остальных маршрутизаторов выполняется самостоятельно.

Адресное пространство, использованное в AS 65500 – 110.0.0.0/24. Чтобы симулировать такую сеть, подключенную к маршрутизатору, можно сконфигурировать на маршрутизаторе B интерфейс Loopback.

Loopback-интерфейс в компьютерных сетях — виртуальный интерфейс, который позволяет устройствам обмениваться данными с самим собой. Он не связан с физическим интерфейсом устройства.

```
R2(config)# interface loopback 0
R2(config-if)# ip address 110.0.0.1 255.255.255.0
```

Затем маршрутизаторы должны анонсировать подсети, к которым имеют доступ. Маршрутизатор B анонсирует сеть 110.0.0.0 с маской 255.255.255.0, которую имеет на интерфейсе Loopback 0.

```
R2(config)# router bgp 65500
R2(config-router)# network 110.0.0.0 mask 255.255.255.0
```

Маршрутизаторы C и D будут анонсировать A и B только маршрут по умолчанию (маршрут, который будет использоваться для всего исходящего трафика с AS 65500).

```
R3(config)# router bgp 100
R3(config-router)# neighbor 150.0.0.1 default-originate
```

5. Настроим управление исходящим трафиком. Если мы хотим, чтобы маршрутизаторы A и B использовали исключительно основное соединение для исходящего трафика, можно воспользоваться атрибутом Local Preference. По умолчанию атрибут имеет значение равное 100. Значение Local Preference

выменивается между всеми маршрутизаторами, находящимися в одной AS. Чем выше значение атрибута, тем выше приоритет соединения. Изменить значение Local Preference можно с помощью Route map. Делаем соединение между A и C основным:

```
R1(config)# router-map primary
R1(config-route-map)# match ip address 1
R1(config-route-map)# set local-preference 150
R1(config-route-map)# exit
R1(config)# access-list 1 permit host 0.0.0.0
```

Затем:

```
R1(config)# router bgp 65500
R1(config-router)# neighbor 150.0.0.2 route-map primary in
```

Чтобы изменения Local Preference вступили в силу, необходимо сбросить предыдущие настройки сессии BGP.

```
R1# clear ip bgp *
```

Чтобы проверить правильность конфигурации, а заодно насладиться результатом, можно воспользоваться утилитой traceroute, выключая и включая определенные интерфейсы, тем самым симулируя аварию соединения.

6. Настроим управление входящим трафиком. Допустим мы хотим сделать так, чтобы маршрутизаторы C и D весь свой трафик направляли через основное соединение. Для этого мы можем воспользоваться атрибутом MED. Основное соединение должно получить меньшее значение (в нашем случае MED 20), чем резерв (MED 30). Делаем это с помощью того же Route map.

```
R1(config)# route-map trafic_out permit 10
R1(config-route-map)# match ip address 10
R1(config-route-map)# set metric 20
R1(config-route-map)# exit
R1(config)# access-list 10 permit host 110.0.0.0
R1(config)# router bgp 65500
R1(config-router)# neighbor 150.0.0.2 route-map trafic_out out
```

На маршрутизаторе B делаем аналогично со значением MED 50. Не забываем сбросить настройки на маршрутизаторах ISP, чтобы могли получить новые значения метрик.

7. Далее выполнить проверку состояния BGP-соседей на каждом маршрутизаторе. Зафиксировать полученные результаты.

Содержание отчета

1. Титульный лист;
2. Цель работы;
3. Теоретические сведения;
4. Описание выполнения работы (по шагам);
5. Полученные результаты (выводы команд, скриншоты);
6. Выводы по результатам выполнения работы.

Контрольные вопросы

1. Как называется параметр времени жизни маршрута?
2. Как функционирует динамическая маршрутизация в сети связи?
3. Как называется основной протокол динамической маршрутизации?
4. Что такое автономная система?
5. Что называется соседями в протоколе BGP?
6. Какие используются типы сообщений для обмена информацией между маршрутизаторами?
7. Какой номер порта используется для обмена информацией между маршрутизаторами?
8. Как называется стадия рабочего состояния сессии?

7.2.12. Изучение управления трафиком по протоколу OSPF.

Ключевые слова

Динамическая маршрутизация, протокол OSPF.

Цель работы

Изучить работу протокола OSPF.

Научиться настраивать протокол OSPF.

Краткие теоретические сведения

Определение OSPF

OSPF (Open Shortest Path First) — протокол динамической маршрутизации, который автоматически находит и перестраивает маршруты при изменениях в сети. Он постоянно анализирует состояние всех доступных маршрутов и с помощью алгоритма выбирает лучший. Благодаря этому OSPF особенно эффективен в высоконагруженных сетях, где необходимо бесперебойно и без задержек передавать большие объёмы данных.

OSPF можно сравнить с автомобильным навигатором: когда маршрутизатор имеет полную карту сети, он легко строит маршрут до любой точки. Это эффективно, поскольку вероятность возникновения ошибок или петель в такой ситуации минимальна. Однако, чем больше маршрутов, тем больше времени и ресурсов требуется для поиска оптимального пути — как и при прокладывании маршрута на навигаторе: поиск пути от Москвы до Санкт-Петербурга занимает больше времени, чем поиск пути в соседний район.

Поэтому OSPF обычно используется внутри сети одной организации и не подходит для маршрутизации в интернете. Однако он может подключаться к внешним сетям через пограничные маршрутизаторы для обмена информацией между внутренней сетью и внешними провайдерами. Для глобальной маршрутизации в интернете чаще применяется протокол BGP.

Принцип работы протокола OSPF

Каждый маршрутизатор, настроенный по протоколу OSPF, обменивается с соседними устройствами информацией о доступных маршрутах и их стоимости (OSPF Cost). Обмен данными происходит по принципу «все со всеми». Полученная информация сохраняется

в базе данных LSDB (link-state database), и на её основе маршрутизаторы определяют соседей и рассчитывают маршруты.

Представьте сеть из пяти маршрутизаторов, как на изображении ниже. Допустим, стоимость передачи данных через каждый интерфейс одинакова. Нам нужно выбрать наиболее быстрый путь для передачи данных с первого на третий маршрутизатор. Оптимальный путь пройдёт через второй маршрутизатор, поскольку задействование четвёртого маршрутизатора добавит лишний узел и увеличит задержку при передаче данных.

При одинаковой стоимости передачи данных через каждый интерфейс оптимальный путь — тот, что проходит через наименьшее количество маршрутизаторов между отправителем и получателем (рис. 78).

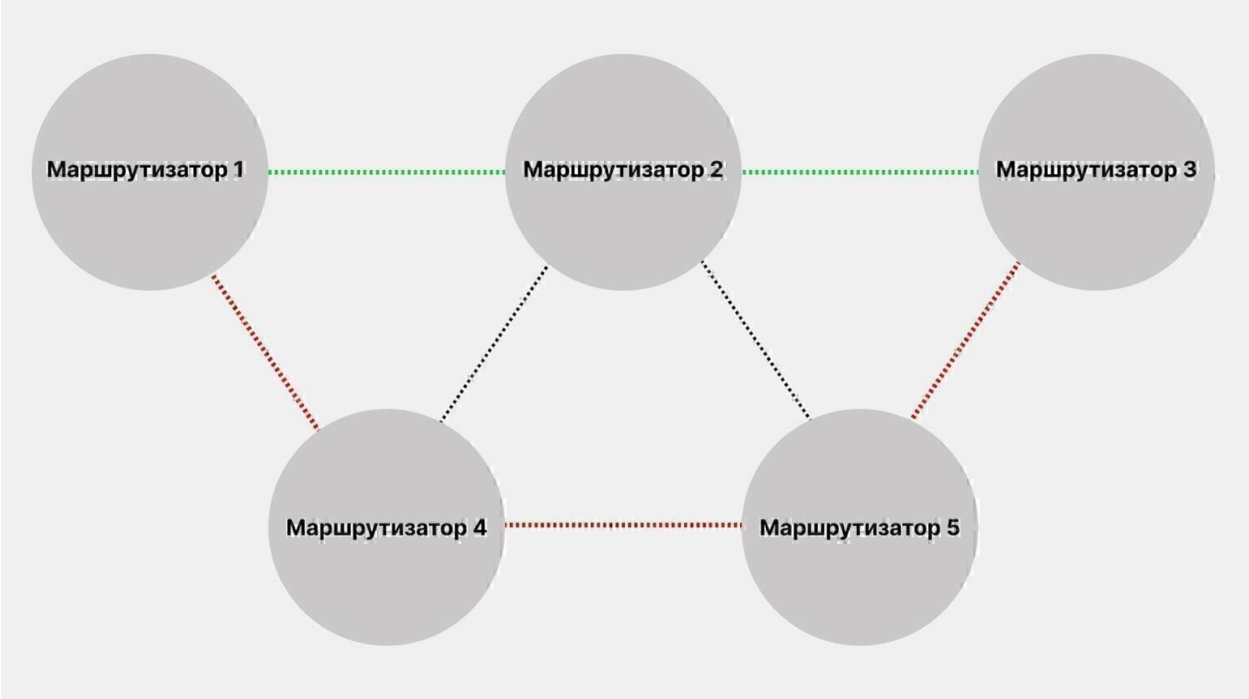


Рис. 78. Выбор оптимального маршрута по протоколу OSPF

При этом каждый маршрутизатор в OSPF автономно выбирает конкретный маршрут, а сообщения о доступных сетевых путях служат лишь справочной информацией. Например, если второй маршрутизатор будет перегружен, первый маршрутизатор может отправить данные через маршрутизатор 4 — в текущих условиях такой маршрут может оказаться оптимальным.

При одинаковой стоимости передачи данных через каждый интерфейс оптимальный путь может измениться, если один из маршрутизаторов окажется перегружен, недоступен или изменится пропускная способность сети (рис.79).

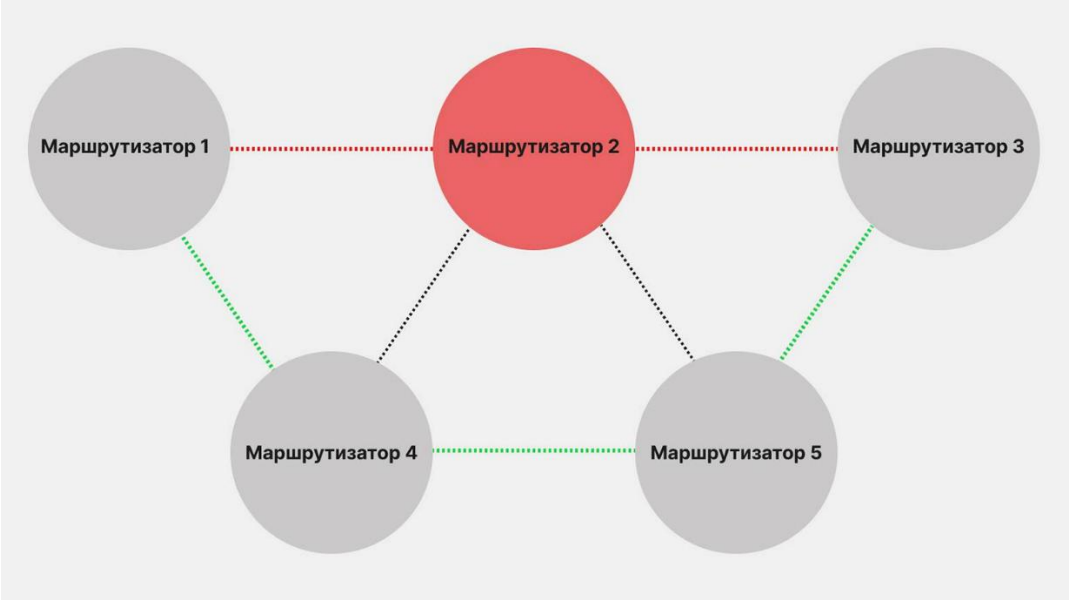


Рис. 79. Выбор маршрута при отказе одного из маршрутизаторов

Задание

Настроить и изучить протокол динамической маршрутизации OSPF.

Алгоритм выполнения лабораторной работы

Работу выполнять в виртуальной среде эмулирования PNetLab.

При достаточной подготовке возможно выполнить работу на оборудовании лаборатории, используя маршрутизаторы Cisco 2801, Зонд агрегации и Зондах CPE1 и CPE2.

1. Собрать схему связи (рис. 80);

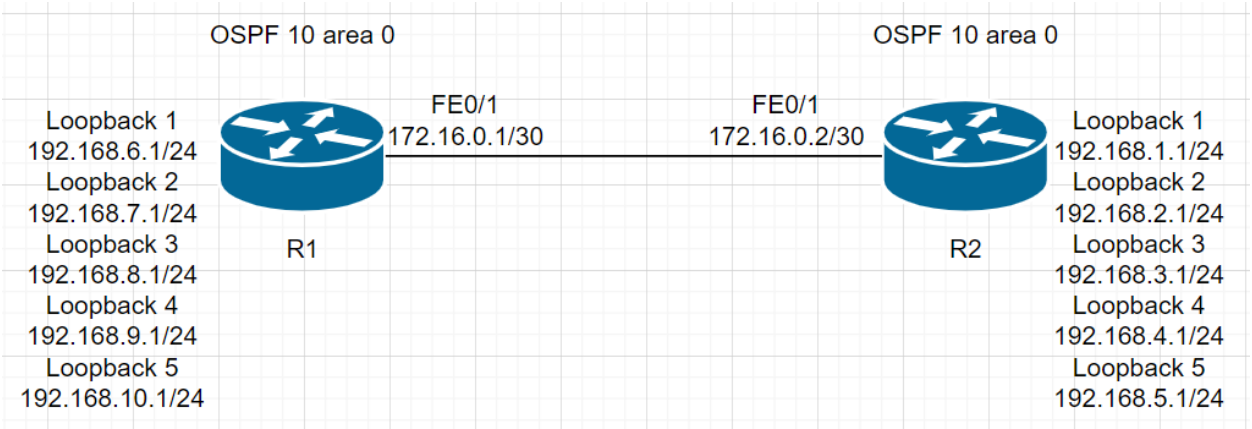


Рис. 80. Схема сети для изучения протокола OSPF

2. Настроить Loopback интерфейс на R2

Маршрутную информацию о loopback интерфейсах мы и будет передавать через OSPF.

Подключаемся к маршрутизатору R2,

conf t – переход в режим глобальной конфигурации,

interface loopback 1 – переход в режим конфигурации интерфейса,

ip address 192.168.1.1 255.255.255.0 – назначение IP адреса.

Результат ввода команд (рис. 81):

```
R2(config)#interface loopback 1
R2(config-if)#ip ad
R2(config-if)#ip address
*Jan  1 00:25:04.983: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback1, changed state to up?
  A.B.C.D  IP address
  pool     IP Address autoconfigured from a local DHCP pool

R2(config-if)#ip address 192.168.1.1 255.255.255.0
```

Рис. 81. Результат ввода команд

Далее вводим команды:

```
interface loopback 2
ip address 192.168.2.1 255.255.255.0
interface loopback 3
ip address 192.168.3.1 255.255.255.0
interface loopback 4
ip address 192.168.4.1 255.255.255.0
interface loopback 5
ip address 192.168.5.1 255.255.255.0
```

Результат ввода команд (рис. 82):

```
R2(config-if)#interface loopback 2
R2(config-if)#ip address 192.168.2.1 255.255.255.0
R2(config-if)#interface loopback 3
R2(config-if)#ip address 192.168.2.1 255.255.255.0
*Jan  1 00:26:33.835: %LINEPROTO-5-UPDOWN: Line protocol on Inteip address 192.168.3.1 255.255.255.0
R2(config-if)#interface loopback 4
R2(config-if)#
*Jan  1 00:27:07.803: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback4, changed state to up
R2(config-if)#ip address 192.168.4.1 255.255.255.0
R2(config-if)#
R2(config-if)#interface loopback 5
R2(config-if)#
R2(config-if)#
*Jan  1 00:27:29.075: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback5, changed state to up
R2(config-if)#ip address 192.168.5.1 255.255.255.0
```

Рис. 82. Результат ввода команд

3. Настроим протокол OSPF на R2

Введем команды:

router ospf 10 – Запуск процесс OSPF с номером 10

router-id 2.2.2.2 - router id это важная часть процесса OSPF, можно считать, что это имя маршрутизатора в процессе OSPF

network 172.16.0.2 0.0.0.3 area 0 – анонсируем сеть и wildcard маску другим маршрутизаторам в OSPF

network 192.168.1.1 0.0.0.255 area 0

network 192.168.2.1 0.0.0.255 area 0

network 192.168.3.1 0.0.0.255 area 0

network 192.168.4.1 0.0.0.255 area 0
network 192.168.5.1 0.0.0.255 area 0

Результат ввода команд (рис. 83):

```
R2(config-if)#
R2(config-if)#exit
R2(config)#router ospf 10
R2(config-router)#rout
R2(config-router)#router-id 2.2.2.2
R2(config-router)#ne
R2(config-router)#net
R2(config-router)#network 172.16.0.2 0.0.0.3
% Incomplete command.

R2(config-router)#network 172.16.0.2 0.0.0.3 ar
R2(config-router)#network 172.16.0.2 0.0.0.3 area 0
R2(config-router)#ne
R2(config-router)#net
R2(config-router)#network 192.168.1.1 0.0.0.255 are
R2(config-router)#network 192.168.1.1 0.0.0.255 area 0
R2(config-router)#network 192.168.2.1 0.0.0.255 area 0
R2(config-router)#network 192.168.3.1 0.0.0.255 area 0
R2(config-router)#network 192.168.4.1 0.0.0.255 area 0
R2(config-router)#network 192.168.5.1 0.0.0.255 area 0
```

Рис. 83. Результат ввода команд

4. Настроить Loopback интерфейс на R1

Подключаемся к маршрутизатору R1 и настраиваем loopback интерфейсы:

```
conf t
interface loopback 1
ip address 192.168.6.1 255.255.255.0
interface loopback 2
ip address 192.168.7.1 255.255.255.0
interface loopback 3
ip address 192.168.8.1 255.255.255.0
interface loopback 4
ip address 192.168.9.1 255.255.255.0
interface loopback 5
ip address 192.168.10.1 255.255.255.0
```

Результат ввода команд (рис. 84):

```
R1>en
R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)#int
R1(config)#interface lo
R1(config)#interface loopback 1
R1(config-if)#
*Nov 29 12:30:02.631: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback1, changed state to up
R1(config-if)#ip ad
R1(config-if)#ip address 192.168.6.1 255.255.255.0
R1(config-if)#interface loopback 2
R1(config-if)#ip address 192.168.6.1 255.255.255.0
*Nov 29 12:30:25.023: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback2, changed state to sdf255.255.255.0
                                          ^
% Invalid input detected at '^' marker.

R1(config-if)#ip address 192.168.7.1 255.255.255.0
R1(config-if)#interface loopback 3
R1(config-if)#
*Nov 29 12:30:52.219: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback3, changed state to up
R1(config-if)#
R1(config-if)#
R1(config-if)#ip address 192.168.8.1 255.255.255.0
R1(config-if)#interface loopback 4
R1(config-if)#
*Nov 29 12:31:42.503: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback4, changed state to up
R1(config-if)#ip address 192.168.9.1 255.255.255.0
R1(config-if)#interface loopback 5
R1(config-if)#
*Nov 29 12:31:58.079: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback5, changed state to up
R1(config-if)#ip address 192.168.10.1 255.255.255.0
R1(config-if)#
```

Рис. 84. Результат ввода команд

5. Настроим протокол OSPF на R1.

Введем команды:

router ospf 10

router-id 1.1.1.1

network 172.16.0.1 0.0.0.3 area 0

Обратите внимание, что после добавления в процесс ospf сети 172.16.0.1 состояние с LOADING до FULL, это означает что база связность OSPF между маршрутизаторами установилась и они обменялись друг с другом базой данных о маршрутизируемых сетях. Далее продолжаем добавлять сети в процесс, тем самым пополняю базу данных для обмена.

Далее введем команды:

network 192.168.6.1 0.0.0.255 area 0

network 192.168.7.1 0.0.0.255 area 0

network 192.168.8.1 0.0.0.255 area 0

network 192.168.9.1 0.0.0.255 area 0

network 192.168.10.1 0.0.0.255 area 0

Результат ввода команд (рис. 85):

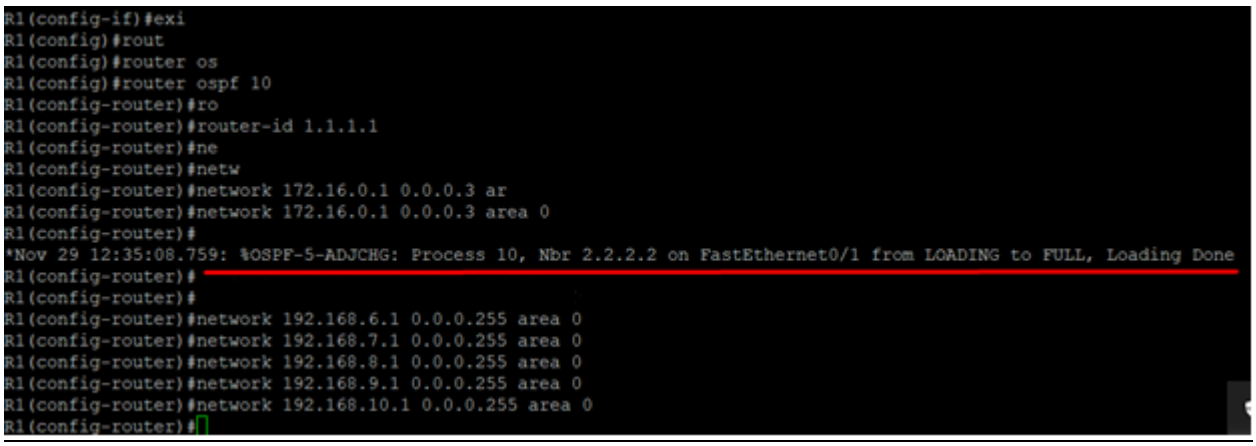


Рис. 85. Результат ввода команд

6. Чтобы проверить как маршрутизаторы обменялись маршрутами, используйте команду show ip route.

Обратите внимание на коды слева от маршрутов и на пояснения что они означают.

7. Проверьте статус процесса ospf с помощью команды show ip ospf neighbor.

Пример вывода команды (рис. 86):

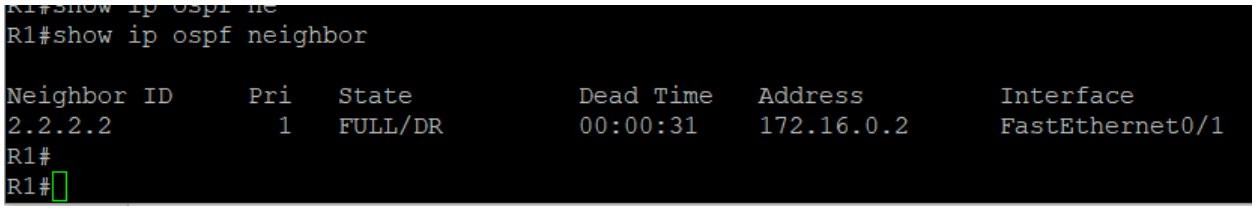


Рис. 86. Пример вывода команды

Детальную информацию можно проверить с помощью команды show ip ospf neighbor detail.

Пример вывода команды (рис. 87):

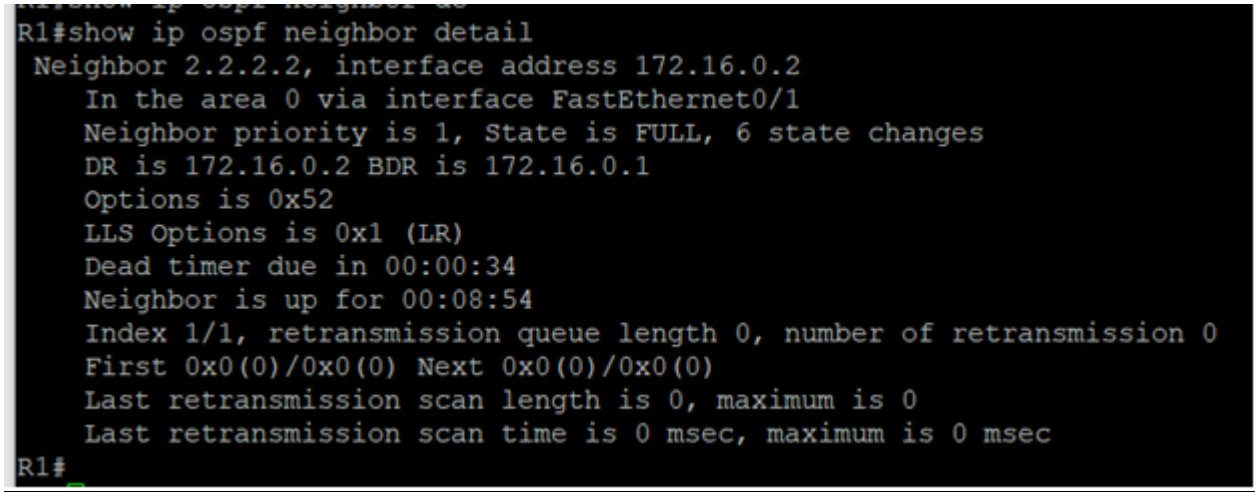


Рис. 87. Пример вывода команды

8. Зафиксируйте результаты полученных измерений.

Содержание отчета

1. Титульный лист;
2. Цель работы;
3. Теоретические сведения;
4. Описание выполнения работы (по шагам);
5. Полученные результаты (выводы команд, скриншоты);
6. Выводы по результатам выполнения работы.

Контрольные вопросы

1. По какому принципу работает протокол OSPF?
2. Что находится в базе данных LSDB?
3. Чем протокол OSPF отличается от протокола BGP?
4. При каких условиях оптимальный путь по протоколу OSPF может измениться?

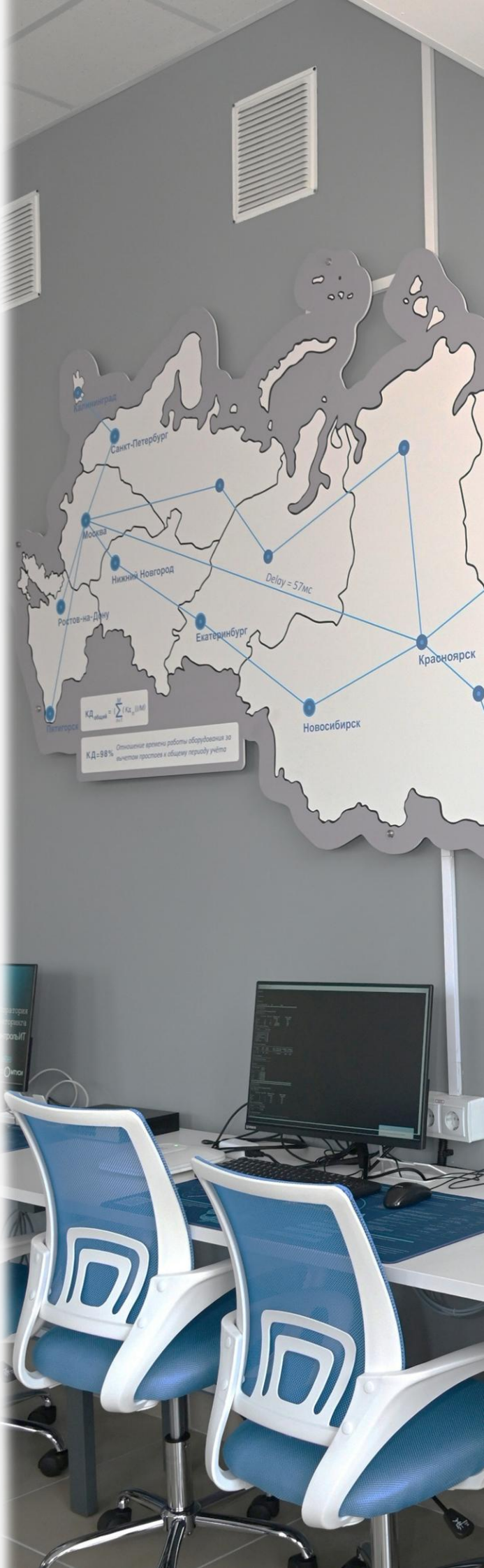
7.3. Перечень лабораторных работ по модулю «Операционная система Linux»

В результате выполнения лабораторных работ вы познакомитесь:
с основными командами командной строки,
с текстовыми редакторами VIM и Nano.

Научитесь:
создавать, просматривать и управлять файлами и директориями,
работать со справочными командами,
устанавливать и настраивать программное обеспечение и пакеты.

Освоите:
основы обработки текстов.

Изучите:
управление пользователями и группами,
настройку разрешений,
работу с текущими процессами операционной системы Linux.



7.3.1. Основы работы с командной строкой.

Ключевые слова

Командная строка, приглашение командной строки, запуск команды.

Цель работы

Изучить основы работы с командной строкой. Уметь "читать" приглашение командной строки. Научиться запускать команды. Изучить механизм истории командной строки и механизм поиска ранее набранных команд. Научиться запускать команды с опциями. Изучить механизм выполнения команд с повышением привилегий. Научиться пользоваться утилитой man.

Краткие теоретические сведения

Командная строка (терминал) в ОС Linux — это текстовый интерфейс, который позволяет взаимодействовать с операционной системой через ввод команд. В отличие от графического интерфейса (GUI), командная строка предоставляет более прямой и гибкий способ управления системой.

Приглашение командной строки

Командная строка состоит из приглашения и области ввода команды, которая находится справа от приглашения. Приглашение предназначено для информирования пользователя о текущем пользователе, сервере и директории. Формат приглашения может быть перенастроен, но его типичный вид следующий:

{USER}@{HOSTNAME}:{DIR}{SIGN}

где USER - имя (login) пользователя, HOSTNAME - имя устройства, DIR - текущая директория (если текущая директория - домашний каталог пользователя, то вместо указания наименования директории выводится символ ~), SIGN - символ приглашения ввода (может быть равен $ для обычного пользователя и # для привилегированного пользователя).

Запуск команд с опциями и без опций

Для запуска команды после символа приглашения необходимо написать текст команды, а затем нажать клавишу Enter. Некоторые команды предполагают указание опций запуска

команды, в таком случае опции пишутся после команды через пробел.

Например, команда echo выводит в консоль произвольную строку. При этом строка, которую требуется вывести передается команде echo в качестве аргумента:

echo некоторая_строка,

при этом некоторая_строка - опция запуска (или "аргумент"), которая сообщает команде echo, что необходимо вывести в консоль.

История запуска команд

Команды, запускаемые пользователем, сохраняются в истории запуска командной оболочки.

Чтобы посмотреть всю историю для текущего терминала, нужно запустить команду **history** без параметров. Она выводит список команд, пронумерованных для удобства поиска и редактирования.

Можно указать количество команд после history — например, вывести только последние 5 введённых команд: history 5.

Для повторного запуска команды можно ввести её номер и поставить перед ним восклицательный знак «!» — например, !1592.

Можно удалить конкретную команду из истории с помощью опции -d — например, history -d 1996

Для очистки всей истории используется опция -c — history -c.

Поиск ранее запущенных команд

Оболочка (например Bash) предоставляет инструмент поиска по ранее набранным командам.

Навигация по истории: для перемещения вверх и вниз используются стрелки или Ctrl+P (предыдущая команда) и Ctrl+N (следующая).

Поиск по истории: можно нажать Ctrl+R, начать вводить любой текст, и терминал покажет совпадения в реальном времени. Повторное нажатие Ctrl+R ищет дальше назад, для поиска вперёд — Ctrl+S.

Повышение привилегий при запуске команд

Некоторые команды требуют административных привилегий у пользователя, запускающего их. Для того, чтобы запустить такую

команду простому пользователю, необходимо перед текстом команды написать **sudo**. Если пользователю разрешено повышение привилегий при запуске этой или всех команд, то она будет выполнена. Например, команда **apt** требует для работы административных привилегий, поэтому для того, чтобы обновить список пакетов, непривилегированный пользователь должен написать **sudo apt update**.

Утилита просмотра встроенной документации man

Пользовательские переменные командной строки

Пользовательские переменные командной строки в Linux — это переменные, определённые пользователем для текущей оболочки (например, оболочки Bash). Они могут быть установлены временно или постоянно. При запуске оболочки пользовательских переменных в ней нет — они появляются, только если пользователь сам объявит переменную, присвоив ей значение.

Переменная доступна только в пределах процесса оболочки, в которой она была объявлена. Однако её можно превратить в переменную среды командой export, тогда она станет доступна и для дочерних процессов, запускаемых оболочкой.

Синтаксис объявления переменной: var1="Переменная1". Если значение переменной не содержит пробелов, кавычки можно опустить, в противном случае они обязательны.

Обращение к переменной: используется её имя, перед которым ставится символ $. Например, вывод значения переменной с помощью команды echo:

echo "$message"

Присвоение значения через ввод с клавиатуры с помощью команды read. Например, скрипт запрашивает у пользователя имя и сохраняет его в переменной: read name

Встроенное выполнение команд

Результат выполнения команды можно «встраивать» в другие команды или в присвоение переменных. Для того, чтобы встроить команду, её надо обрамить обратными кавычками.

Например, если написать var1=`некоторая_команда`, то результат выполнения команды будет помещен в переменную var1.

Если написать:

echo "первая_часть_текса `некоторая_команда` вторая_часть_текста",

то результат выполнения команды встроится между первой и второй частями текста и будет выведена в консоль.

Алгоритм выполнения лабораторной работы

В данной работе используются команды:

- Вывода строки текста в консоль – echo;
- Вывода и установки системного времени и даты – date;
- Вывода имени машины – hostname.

В процессе выполнения работы в консоль мы выведем произвольные строки, изменим текущее системное время и просмотрим список запущенных процессов.

1. Глядя на приглашение командной строки, определите имя машины;
2. С помощью команды hostname получите имя машины и сравните с именем машины из приглашения командной строки;
3. Создайте переменную оболочки с именем deviceName, поместите в неё имя машины, полученное с помощью команды hostname с помощью механизма встроенного выполнения команд;
4. С помощью команды echo выведите имя машины в консоль несколькими способами:
   4.1. Передав команде непосредственно имя машины;
   4.2. Передав команде переменную deviceName;
5. Выведите с помощью команды echo строку: «на машине имя_машины системное время равно системное_время», при этом имя_машины должно быть подставлено из переменной deviceName, а системное_время должно быть подставлено динамически, с помощью встраивания команды date;
6. Повторно выполните п.3 (создайте переменную), но не набирайте команду "с нуля", а запустите её с помощью механизма истории команд (через номер команды);
7. Повторите п.5 (выведите строку с именем машины и системным временем), но не набирайте команду удаления "с нуля", а найдите ранее набранную команду через механизм поиска ранее использованных команд;
8. Попробуйте выполнить команду apt update - она должна завершиться с ошибками, так как для её выполнения требуется

повышение привилегий пользователя. Затем выполните ту же команду с повышением привилегий пользователя и убедитесь, что теперь ошибок нет.

Содержание отчета

1. Титульный лист.
2. Цель работы.
3. Теоретические сведения.
4. Описание выполнения работы (по шагам).
5. Полученные результаты, скриншоты.
6. Выводы.

Контрольные вопросы

1. Каков стандартный формат приглашения командной строки?
2. Как объявить переменную? Как её использовать после объявления?
3. Как передать команде результат выполнения другой команды?
4. Как произвести просмотр истории команд и запустить её из истории?
5. Как произвести поиск ранее выполненных команд?
6. Как выполнять команды с повышенными привилегиями?

7.3.2. Изучение команд создания и просмотра файлов.

Ключевые слова

Команды touch, file, cat, less, history.

Цель работы

Изучить основные команды создания и просмотра файлов, которые используются в Unix-подобных ОС.

Краткие теоретические сведения

Команда touch

При входе в консольной программе на устройство с ОС Linux, как правило, попадаем в домашний каталог учетной записи, отведенный для хранения файлов и создания директорий.

Иногда возникает необходимость в создании некоторых файлов. Очень простой способ это сделать - использовать команду touch. Эта позволяет создавать новые пустые файлы.

Пример:

$ **touch mysuperduperfile**

Команда touch также используется для изменения времени последнего изменения на файлах и директориях.

Для начала выполним команду ls -l в директории и обратим внимание на время:

```
$ ls -l
total 1380
-rw-r--r-- 1 root root      0 мая 29  2024 1
-rw-r--r-- 1 root root      4 июн 20 13:33 1.txt
-rw-r--r-- 1 root root    730 июн 11 15:18 kmut-zond.bak
-rw-r--r--   1   root   root   421688   июн   28      2024
libstrongswan_5.7.2-1+deb10u2_amd64.deb
-rw-r--r-- 1 root root 141660 июн 28  2024 libstrongswan-
standard-plugins_5.7.2-1+deb10u2_amd64.deb
-rw-r--r-- 1 root root      0 июл  9 08:13 mysuperduperfile
```

Затем выполним команду **touch mysuperduperfile**, который обновит время изменения файла, и снова обратим внимание на время:

```
$ ls -l
total 1380
-rw-r--r-- 1 root root      0 мая 29  2024 1
-rw-r--r-- 1 root root      4 июн 20 13:33 1.txt
-rw-r--r-- 1 root root    730 июн 11 15:18 kmut-zond.bak
-rw-r--r--  1  root  root  421688  июн  28   2024  libstrongswan_5.7.2-
1+deb10u2_amd64.deb
      -rw-r--r-- 1 root root 141660 июн 28  2024 libstrongswan-standard-
plugins_5.7.2-1+deb10u2_amd64.deb
      -rw-r--r-- 1 root root      0 июл  9 08:32 mysuperduperfile
```

Как видим, время изменения файла обновилось.

Команда file

Можно заметить, что в Unix-подобных системах имя файла не соответствует стандарту наименования файлов, который есть в других операционных системах (например, Windows). В Linux имена файлов не требуют представлять содержимое файла. Можно создать файл funny.gif, который на самом деле не будет соответствовать формату GIF.

Чтобы обнаружить, к какому формату принадлежит файл, можно использовать команду file. Она покажет описание содержимого файла.

Примеры:

```
  $ file test.lot
  test.lot: ASCII text
  $ file /bin/ping
  /bin/ping: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV),
dynamically    linked,    interpreter    /lib64/ld-linux-x86-64.so.2,
BuildID[sha1]=4f75c6991b771bd1c2409e239d5354b58f4d3062, for GNU/Linux
3.2.0, stripped
```

Команда cat

Попробуем считать файл. Простая команда для этого – cat, сокращенно от "конкатенация" (concatenate, catenate), которая не только отображает содержимое файла, но также может комбинировать содержимое нескольких файлов и отображать их вывод.

Команда не очень хорошо подходит для просмотра больших файлов и предназначена для короткого содержимого (для просмотра файлов большего размера существуют другие инструменты, см. лабораторную работу 7.3.8, 7.3.9).

Пример:

$ **cat test.lot**

d myv

Команда less

Если просматривать текстовые файлы, которые больше, чем простой вывод, то less (меньше) — значит больше (существует также команда, названная more (больше), которая делает похожие вещи, но обладает меньшим функционалом,). Текст отображается постранично, а значит можно перемещаться по тексту страница за страницей.

Посмотрим на содержимое файла с less. Введя команду less можно перемещаться по документу с помощью клавиатуры.

Пример:

$ **less /home/pete/Documents/text1**

Можно использовать эти команды для навигации с помощью less:

- q - Используется для выхода из less и возврата в вашу оболочку;
- Page up, Page down, Вверх и Вниз - Перемещение с помощью стрелок и кнопок страниц;
- g - Перемещает к началу файла;
- G - Перемещает в конец файла;
- /search - Можно искать (search) определенный текст внутри документа. Определять слова, которые нужно найти с помощью /
- h - Если нужна небольшая помощь по использованию less, пока вы в less, то можно воспользоваться помощью (help).

Команда history

В оболочке присутствует история команд, которые вводились ранее, их можно просмотреть. Довольно полезно бывает найти и выполнить команду, которая вводилась ранее, не печатая ее снова.

Пример:

```
$ history
```

Для запуска команды, которая запускалась до этого момента, можно просто нажать стрелку вверх.

Для запуска команды, введенной ранее, без ее набора можно использовать **!!**. Если вы ввели cat file1 и хотите запустить ее еще раз, вы можете просто набрать !!, это выполнит последнюю команду, которую вы запускали.

Другой способ пробежаться по истории - Ctrl + R, это команда обратного (Reverse) поиска, если вы нажмете Ctrl + R и начнете печатать часть команды, которую вы хотите, команда покажет вам совпадения, по которым вы можете перемещаться, нажав Ctrl + R снова. Найдя нужную команду, нажмите Enter.

Дополнительно

Для очистки экрана консольной программы от лишних строк можно использовать команду **clear** (очистить).

Также стоит упомянуть одну из наиболее полезных функций в любой командной среде - автозавершение по tab. Если вы начнете печатать начало команды, файла, директории и т.д. и нажмете клавишу Tab, она автоматически завершит ввод, основываясь на том, что было найдено в директории, если не найдено других названий, начинающихся на те же буквы.

Например, если вы хотите ввести команду history, вы можете ввести his и нажать Tab, которая дополнит слово до history.

Алгоритм выполнения лабораторной работы

1. Создайте новый файл.
2. Зафиксируйте время изменения.
3. Выполните команду touch и проверьте дату изменения снова.
4. Выполните команду file на трех разных файлах в трех каталогах.
5. Выполните команду cat на различных файлах и каталогах, затем попробуйте использовать cat на нескольких файлах одновременно.
6. Запустите less на файле, затем пролистайте пару страниц. Попробуйте найти какое-либо слово. Переместитесь в начало и в конец файла.
7. Поперемещайтесь по истории ваших команд используя клавиши «Вверх» и «Вниз».
8. Выполните команду обратного поиска на Ctrl + R.

Содержание отчета

1. Титульный лист.
2. Цель работы.
3. Теоретические сведения.
4. Описание выполнения работы (по шагам).
5. Полученные результаты, скриншоты.
6. Выводы.

Контрольные вопросы

1. Как вы создадите файл, названный myfile?
2. Какую команду вы будете использовать для определения типа файла?
3. Как вы можете просмотреть содержимое файла?
4. Как выйти из команды less?
5. Какая команда очищает экран консольной программы?

7.3.3. Взаимодействие с файлами и директориями в командной оболочке.

Ключевые слова

Команды pwd, ls, cd, mkdir, touch, cp, mv, rm.

Цель работы

Изучить основные команды для работы с файлами и директориями. Научиться создавать, копировать, перемещать, переименовывать и удалять файлы и папки, а также управлять их правами доступа.

Краткие теоретические сведения

В операционной системе Linux все данные организованы в виде файлов и директорий (папок). Для работы с ними используется командная оболочка (терминал). Рассмотрим основные команды.

Для навигации по файловой системе используются следующие команды:

- pwd – показать текущую директорию;
- ls – список файлов и папок в текущей директории;
- cd <директория> – перейти в указанную директорию.

Работа с файлами и директориями:

- touch <файл> – создать пустой файл;
- mkdir <директория> – создать папку;
- cp <источник> <назначение> – копировать файл или папку;
- mv <источник> <назначение> – переместить или переименовать файл/папку;
- rm <файл> – удалить файл.
- rm -r <директория> – удалить папку рекурсивно.

Права доступа (подробнее права доступа рассматриваются в лабораторной работе «Управление разрешениями»):

- chmod <права> <файл> – изменить права доступа;
- chown <пользователь>:<группа> <файл> – изменить владельца.

Алгоритм выполнения лабораторной работы

Задание 1: Навигация и создание файлов

- Откройте терминал;
- Выведите текущую директорию, выполнив команду pwd;
- Создайте папку lab_files:

mkdir lab_files

- Перейдите в неё:

cd lab_files

- Создайте файл test.txt:

touch test.txt

- Проверьте содержимое папки:

ls

Задание 2: Копирование и перемещение

- Скопируйте test.txt в backup.txt:

cp test.txt backup.txt

- Переименуйте backup.txt в new_backup.txt:

mv backup.txt new_backup.txt

- Создайте папку archive и переместите туда new_backup.txt:

mkdir archive

mv new_backup.txt archive/

Задание 3: Удаление

- Удалите файл test.txt:

rm test.txt

- Удалите папку archive (с содержимым):

rm -r archive

Задание 4: Права доступа

- Создайте файл secret.txt:

touch secret.txt

- Установите права 600 (только владелец может читать и писать):

chmod 600 secret.txt

- Проверьте права:

ls -l

Содержание отчета

1. Титульный лист.
2. Цель работы.
3. Теоретические сведения.
4. Описание выполнения работы (по шагам).
5. Полученные результаты, скриншоты.
6. Выводы.

Контрольные вопросы

1. Какая команда показывает текущую директорию?
2. Как создать папку в Linux?
3. Чем отличается cp от mv?
4. Как удалить папку с содержимым?
5. Как изменить права доступа к файлу?

7.3.4. Изучение справочных команд.

Ключевые слова

Команды man, info, help, whatis, apropos.

Цель работы

Освоить основные справочные команды Linux для получения информации о системе, командах и их опциях.

Краткие теоретические сведения

В ОС Linux существует несколько способов получения справочной информации:

Команда man - основной справочник (manual pages);

Команда info - альтернативная справочная система;

Команда help - встроенная справка по командам оболочки;

Команда whatis - краткое описание команды;

Команда apropos - поиск команд по ключевым словам,

Опция --help - краткая справка по команде.

Алгоритм выполнения лабораторной работы

Задание 1: Работа с командой man

- Откройте терминал
- Выведите список всех разделов справочника:

$ man -k

- Просмотрите справочную страницу команды ls:

$ man ls

- Используйте клавиши:

Пробел - пролистать вперед;
b- пролистать назад;
/ - поиск (например, /option для поиска слова "option");
q – выход.

- Найдите все команды, связанные с паролями:

$ man -k password

Задание 2: Команда info

- Просмотрите информацию о команде ls с помощью info:

$ info ls

- Сравните с выводом команды man ls

Задание 3: Встроенная справка help

- Получите список всех встроенных команд оболочки:

$ help

- Просмотрите справку по команде cd:

$ help cd

Задание 4: Использование --help

- Получите краткую справку по команде ls:

$ ls --help

- Сравните с выводом man ls

Задание 5: Команды whatis и apropos

- Узнайте краткое описание команды grep:

$ whatis grep

- Найдите все команды, связанные с сетью:

$ apropos network

Содержание отчета

1. Титульный лист.
2. Цель работы.
3. Теоретические сведения.
4. Описание выполнения работы (по шагам).
5. Полученные результаты, скриншоты.
6. Выводы.

Контрольные вопросы

1. Какие разделы существуют в справочнике man? Для чего предназначен каждый?
2. Чем отличается вывод man от info?
3. Какие команды можно просматривать с помощью help?
4. Как получить справку по опциям конкретной команды?
5. Как найти все команды, связанные с определенной темой?

7.3.5. Основы работы с текстом. Часть 1.

Ключевые слова

Команды env, cut, paste, grep, tail, head, текстовые утилиты, обработка текста.

Цель работы

Освоить базовые команды ОС Linux для работы с текстом. Научиться извлекать, комбинировать и фильтровать данные из текстовых файлов.

Краткие теоретические сведения

Команда env (Environment)

Отображает переменные окружения текущей сессии или запускает программу в изменённом окружении.

Синтаксис:

env [опции] [переменная=значение] [команда]

Команда cut

Извлекает заданные части строк из файла или ввода.

Основные опции команды:

- -d – задает разделитель полей (по умолчанию TAB);
- -f – указывает номера полей для вывода.

Рассмотрим пример использования команды.

Извлечь 1-е и 3-е поля из файла /etc/passwd, разделённые двоеточием:

cut -d':' -f1,3 /etc/passwd

Команда paste

Объединяет строки из нескольких файлов в одну строку с разделителем.

Синтаксис:

paste [опции] файл1 файл2

Рассмотрим пример использования команды.

Объединить строки через запятую:

paste file1.txt file2.txt -d','

Команда grep

Ищет текст по шаблону (регулярным выражениям).

Основные опции команды:

- -i – игнорировать регистр.
- -v – инвертировать поиск (вывести строки, не содержащие шаблон).
- -n – выводить номера строк.

Рассмотрим пример использования команды.

Найти слово "error" без учёта регистра:

grep -i "error" log.txt

Команды head и tail

- head – выводит первые N строк файла (по умолчанию 10).
- tail – выводит последние N строк файла.

Рассмотрим примеры использования команд.

Вывести первые пять строк:

head -n 5 file.txt

Вывести последние три строки:

tail -n 3 file.log

Алгоритм выполнения лабораторной работы

Задание 1. Подготовка файлов

➢ Создайте два текстовых файла:

```
echo -e "user1:1000:bash\nuser2:1001:zsh\nuser3:1002:fish" > users.txt
echo -e "error: file not found\nwarning: low memory\ninfo: process started" > log.txt
```

Задание 2. Работа с cut

➢ Извлеките из users.txt имена пользователей и их оболочки:

```
cut -d':' -f1,3 users.txt
```

Задание 3. Работа с paste

➢ Создайте два файла и объедините их построчно:

```
echo -e "1\n2\n3" > nums.txt
echo -e "A\nB\nC" > letters.txt
paste nums.txt letters.txt -d':'
```

Задание 4. Работа с grep

➢ Найдите в log.txt строки, содержащие "error" или "warning":

```
grep -E "error|warning" log.txt
```

Задание 5. Работа с head и tail

➢ Выведите первые 2 строки users.txt и последнюю строку log.txt:

```
head -n 2 users.txt
tail -n 1 log.txt
```

Содержание отчета

1. Титульный лист.
2. Цель работы.
3. Теоретические сведения.
4. Описание выполнения работы (по шагам).
5. Полученные результаты, скриншоты.
6. Выводы.

Контрольные вопросы

1. Для чего используется команда cut?
2. Как с помощью grep найти строки, **не** содержащие определённое слово?
3. Чем отличаются команды head от tail?
4. Как объединить два файла построчно с помощью команды paste?
5. Как вывести переменные окружения в Linux?

7.3.6. Основы работы с текстом. Часть 2.

Ключевые слова

Команды sort, wc, nl, uniq, tr, expand, unexpand, join, split.

Цель работы

Освоить базовые команды ОС Linux для работы с текстом. Научиться оперировать данными в текстовых файлах.

Краткие теоретические сведения

Команда sort

Команда sort полезна для сортировки строк. Ее можно использовать для сортировки текста из одного или нескольких файлов или с помощью нее может быть выполнена сортировка вывода для какой-либо команды.

Сначала рассмотрим общий синтаксис команды:

$ **sort [параметры] [файл]**

Основные параметры

Теперь рассмотрим основные опции утилиты sort.

- -b — не учитывать пробелы
- -d — использовать для сортировки только буквы и цифры
- -n — сортировка строк linux по числовому значению
- -r — сортировать в обратном порядке

Вот несколько принципов, по которым команда sort сортирует строки:

- Строки с цифрами размещаются выше других строк
- Строки, начинающиеся с букв нижнего регистра, размещаются выше
- Сортировка выполняется в соответствии алфавиту
- Строки сначала сортируются по алфавиту, а уже вторично по другим правилам.

Пример использования команды sort

Для примера используется файл file1.txt в котором следующие строки:

dog
cow
cat
elephant
bird

После выполнения команды сорт строки выведутся в алфавитном порядке

```
$ sort file1.txt
bird
cat
cow
dog
elephant
```

Тот же файл можно отсортировать в обратном порядке, используя опцию -r.

```
$ sort -r file1.txt
elephant
dog
cow
cat
bird
```

Команды wc и nl

Анализ файлов - неотъемлемая часть работы с ними. Иногда возникает необходимость подсчитать количество строк или слов в тексте. С этой задачей эффективно справляется команды wc и nl.

Команды wc

Команда wc подсчитывает число строк, слов и байт в файлах, указанных в параметре «файл».

Синтаксис команды wc:

$ **wc [параметры] [объект]**

Основные параметры команды wc:

- -c — Отобразить размер объекта в байтах;
- -m — Показать количесто символов в объекте;
- -l — Вывести количество строк в объекте;
- -w — Отобразить количество слов в объекте.

Под объектом следует понимать файл или данные, полученные на стандартный поток ввода.

Команда может обработать несколько файлов, если указать их через пробел или выбрать по шаблону.

Примеры использования wc:

– Анализ файла

$ wc test

3 4 31 test

Согласно анализу, в файле 3 строки, содержащих 4 слова, объёмом в 31 байт.

– Подсчет количества пользователей в системе

$ wc -l /etc/passwd

46 /etc/passwd

Согласно анализу в системе 46 пользователей

Синтаксис команды nl

Команда nl считывает указанный файл (по умолчанию - стандартный поток ввода), нумерует строки и записывает их в стандартный поток вывода. Строки нумеруются слева в соответствии с действующими опциями команды.

$ **nl [параметры] [файл]**

Параметры команды nl:

- -b — Использовать СТИЛЬ для нумерации строк основного текста;
- -f — Использовать СТИЛЬ для нумерации строк нижнего колонтитула;
- -h — Использовать СТИЛЬ для нумерации строк верхнего колонтитула;
- -i — Шаг увеличения номеров строк;
- -l — Заданное ЧИСЛО пустых строк считать одной строкой;
- -n — Использовать ФОРМАТ для номеров строк;
- -p — Не начинать нумерацию заново после каждой логической страницы;
- -s — Добавлять СТРОКУ после номера;
- -v — Первый номер строки на каждой логической странице;
- -w — Использовать заданное ЧИСЛО столбцов для номеров строк.

Примеры использования:

Команда nl добавила номера строк слева от каждой строки в файле и вывела в командную оболчку, как команда cat.

```
$ nl config.txt
     1  # Server Configuration
     2  port=8080
     3  max_connections=100

     4  # Database Settings
     5  db_host=localhost
     6  db_port=5432
     7  db_name=myapp
    ...
```

По умолчанию nl не нумерует пустые строки, для из подсчета используются параметр “–b a”

```
$ nl –b a config.txt
     1  # Server Configuration
     2  port=8080
     3  max_connections=100
     4
     5  # Database Settings
     6  db_host=localhost
     7  db_port=5432
     8  db_name=myapp
    ...
```

Команда uniq

Команда uniq предназначена для поиска одинаковых строк в массивах текста. При этом с найденными совпадениями пользователь может совершать множество действий — например, удалять их из вывода либо наоборот, выводить только их. Работа команды осуществляется как с текстовыми файлами, так и с текстом, напечатанным в командной строке терминала.

Синтаксис команды uniq

$ **uniq параметры файл_источник файл_для_записи**

Основные параметры:

- -u — Выводит исключительно те строки, у которых нет повторов;
- -d— Если какая-либо строка повторяется несколько раз, она будет выведена лишь единожды;
- -D — Выводит только повторяющиеся строки.

Примеры использования:

Прежде всего следует отметить главную особенность команды uniq — она сравнивает только строки, которые находятся рядом. То есть, если две строки, состоящие из одинакового набора символов, идут подряд, то они будут обнаружены, а если между ними расположена строка с отличающимся набором символов — то не будут поэтому перед сравнением желательно отсортировать строки с помощью sort.

Без задействования файлов и параметров uniq работает так:

```
$ echo -e
небо\\noблака\\noблака\\noблака\\nсолнце\\nзвезды | uniq
небо
облака
солнце
звезды
```

После команды uniq можно использовать её опции. Вот пример вывода, где не просто удалены повторы, но и указано количество одинаковых строк:

```
$ echo -e
небо\\noблака\\noблака\\noблака\\nсолнце\\nзвезды | uniq -c
      1 небо
      3 облака
      1 солнце
      1 звезды
```

Команда **tr**

Консольная команда tr, используется для замены, замещения или удаления символов из стандартного ввода, отправляя результат на стандартный вывод.

Синтаксис команды tr

Программа обрабатывает текст посимвольно. По умолчанию у её синтаксиса следующий вид (квадратные скобки указывают, что аргумент не обязателен):

tr [КЛЮЧ]... НАБОР1 [НАБОР2]

Основные параметры:

- -c — Сначала получить дополнение НАБОРА1
- -d — Удалить знаки из НАБОРА2, не превращать

- -s — Замещать последовательность знаков, которые повторяются, из перечисленных в последнем НАБОРЕ, на один такой знак
- -t — Сначала сократить НАБОР1 до размеров НАБОРА2

НАБОРЫ указываются как символьные строки. В большинстве случаев символы представляют сами себя.

Примеры использования:

Превращение осуществляется, если не указано -d для обоих НАБОРОВ. -t можно использовать только во время превращения. Если нужно, НАБОР2 будет расширен до размеров НАБОРА1 повторением последнего символа. Лишние символы НАБОРА2 будут пропущены.

Пример 1. Заменить все y на f.

$ **tr y f**

ayyy yfq #ввод данных с клавиатуры

afff ffq #вывод результаты замены

Пример 2. Удалить все буквы в нижнем регистре.

$ **tr --delete [:lower:]**

AsdfgVFHwer

AVFH

Команда tr — это качественный инструмент для работы с символами строк в терминале GNU/Linux. С её помощью можно редактировать информацию со стандартного или перенаправленного потока ввода и выводить результат на экран или в файл.

Команды expand и unexpand

Команды Expand и Unexpand используются для замены символов табуляции в файлах символами пробела и наоборот. Табуляции часто используются в текстовых файлах для создания отступов, но они могут вызвать проблемы с форматированием при просмотре или обработке файла в разных системах. Команда expand помогает стандартизировать форматирование, заменяя табуляции эквивалентным количеством пробелов.

Синтаксис команды expand

команда Expand заменяет символы табуляции в файле на символы пробела.

$ expand [параметры] файл

Основные параметры:

- -a — Конвертировать все символы табуляции;
- -i — Не преобразуйте табуляции после непустых символов;
- -t — Табуляция с определенным количеством символов, а не с 8 (значение по умолчанию):
- -t — Используется список позиций табуляции, разделенных запятыми

Примеры использования:

$ **cat test.txt**

123 123 123 123

Замена символа табуляции по определенному шаблону

$ **expand -t=5,10,15 test.txt**

123 123 123 123

Замена символов табуляции на 5 пробелов

$ **expand -t 5 test.txt**

123 123 123 123

Синтаксис команды unexpand

Команда Unexpand делает противоположное команде Expand, т.е. она преобразует символы пробела в символы табуляции.

$ expand [параметры] файл

Основные параметры

- -a не преобразуйте табуляции после непустых символов
- -t табуляция может содержать определенное количество символов, а не 8
- -t указать несколько позиций табуляции с разделением запятой

Примеры использования

$ unexpand –a

$ unexpand -t 5

$ unexpand -t "5 10 15"

Команды join и split

Команда **join** выполняет соединение строк из двух текстовых файлов на основании совпадения указанных полей. Это удобно, например, для слияния файлов, особенно логов, например, два файла логов и вам необходимо их сопоставить по времени.

Синтаксис команды join

join [параметры] [входной_файл1] [входной_файл2]

Основные параметры:

- -a n — Задает включение в выходной поток строк из файла n (n -l или 2), для которых не было найдено ни одного совпадения по указанному полю.
- -1 поле — Объединяет строки по указанному полю первого файла (по умолчанию таковым является первое поле)
- -2 поле — Объединяет строки по указанному полю второго файла (по умолчанию таковым является первое поле)
- -t — Этот символ задает разделитель полей во входном и выходном потоках.

Примеры использования:

echo -e "1 apple\n2 banana\n3 cherry" > fruits.txt

echo -e "1 red\n2 yellow\n3 red" > colors.txt

join fruits.txt colors.txt

1 apple red

2 banana yellow

3 cherry red

Команда **split** бьет файл на куски. Даная команда работает следующим образом. Данная команда разбивает файл на части, но при этом исходный не меняет.

Синтаксис команды split:

split[параметры] [файл] [префикс]

Основные параметры:

- -b — Записывать в выходной файл заданное ЧИСЛО байт;
- -d — Использовать числовые (начинающиеся с 0), а не буквенные суффиксы;
- -l — Записывать в каждый выходной файл заданное ЧИСЛО строк;
- -n — Генерировать выходные файлы по ПОРЦИЯМ (задать количество выводимых файлов).

Примеры использования:

- Разобьем по строчкам фал 1.txt. split -l 2 1.txt. Разбивку делаем на 2 строчки. И мы видим, что у нас исходный файл остался неизменным, а появилось еще 2 файла xaa и xab. Они как раз и содержат разбиение.

• Данную команду удобно применять к большим файлам и использовать ключик для разбивки по размеру, например, по байтам b и указываем на какие куски разбить в байтах.

Пример:

split b 5 путь_к_файлу

Алгоритм выполнения лабораторной работы

1. Создайте новый файл с именем languages.txt с помощью следующей команды:

```
 echo -e
"Python\nJava\nRuby\nGo\nJavaScript\nPHP\nRust\nC++\nSwift\nKotlin\nRuby\nRus
t" > languages.txt
```

2. Отсортируйте этот файл по алфавиту с использованием команды sort и занесите результат выполнения в отчет;
3. Проанализируйте ранее созданный файл languages.txt с помощью команды wc;
4. Проанализируйте с помощью команды nl количество строк в файле /etc/network/intefaces, включая пустые;
5. Выполните команду uniq к файлу languages.txt;
6. С помощью команды tr замените PHP на C++ в файле languages.txt;
7. Разбейте файл languages.txt на 2 файла командой split;
8. Объедините полученные файлы из предыдущего пункта командой join.

Содержание отчета

1. Титульный лист.
2. Цель работы.
3. Теоретические сведения.
4. Описание выполнения работы (по шагам).
5. Полученные результаты, скриншоты.
6. Выводы.

Контрольные вопросы

1. Какими командами можно узнать количество строк в файле?
2. Каким образом отсортировать строки в файле в обратном порядке?
3. Каким образом можно разбить файл на три файла?
4. Какой командой можно сделать замену?

7.3.7. Основы работы с текстом. Часть 3.

Ключевые слова

Команды pipe, tee, stdin, stdout, stderr.

Цель работы

Изучить работу с перенаправлением потоков ввода-вывода, конвейерами (pipe), командами tee, а также стандартными потоками stdin, stdout и stderr.

Краткие теоретические сведения

Команды pipe и tee

Конвейер в Linux — это процесс направления вывода одной команды в качестве входных данных для другой команды. То, что выводит одна команда, подаётся на вход другой команды.

Для обозначения pipe и использования конвейера используется символ вертикальной черты **|**. То, что команда, находящаяся в левой части конструкции, отправляет в stdout, попадает в stdin команды, которая находится справа от символа конвейера.

Чтобы вывести вывод моей команды в два различных потока, можно использовать команду tee (разделитель). Команда tee читает данные из стандартного ввода и записывает их одновременно в файл и стандартный вывод.

Пример:

$ **ls | tee peanuts.txt**

Команда stderr (стандартный поток ошибок)

Stderr (стандартный вывод ошибок) в ОС Linux — это поток, который программы используют для вывода сообщений об ошибках и другой диагностической информации. По умолчанию он связан с терминалом, но может быть перенаправлен в отдельное место.

stderr (дескриптор 2) — поток для вывода ошибок.

> — оператор перенаправления, позволяющий нам изменить направление стандартного вывода.

Команда stdin (стандартный ввод)

Stdin (дескриптор 0) — поток ввода данных (обычно с клавиатуры).

Команда stdout (стандартный вывод)

Stdout (дескриптор 1) — поток для вывода данных (по умолчанию — терминал).

Алгоритм выполнения лабораторной работы

Задание 1. Работа с командой tee

1. Используйте конвейер (|) для передачи вывода команды ls команде grep:

#ls -l | grep ".txt"

Объяснение: вывод ls -l передаётся grep, который фильтрует строки, содержащие .txt.

2. Сохраните вывод команды ls в файл и одновременно выведите его на экран с помощью tee:

#ls | tee file_list.txt

Пояснение: tee сохраняет вывод ls в file_list.txt и выводит его в терминал.

3. Используйте tee с конвейером:

#cat /etc/passwd | grep "root" | tee root_user.txt

Объяснение: вывод cat /etc/passwd фильтруется grep, затем сохраняется в root_user.txt и выводится на экран.

Задание 2. Работа с командой stderr

1. Создайте ошибку (например, попробуйте прочитать несуществующий файл):

#cat non_existent_file.txt

Объяснение: команда выведет ошибку в stderr.

2. Перенаправьте stderr в файл:

#cat non_existent_file.txt 2> error.log

Объяснение: ошибка запишется в error.log вместо вывода в терминал.

3. Перенаправьте stdout и stderr в разные файлы:

#ls existing_file.txt non_existent_file.txt > output.log 2> error.log

Объяснение: корректный вывод попадёт в output.log, ошибки — в error.log.

4. Перенаправьте stderr в stdout (для обработки ошибок вместе с обычным выводом):

#ls existing_file.txt non_existent_file.txt > combined.log 2>&1

Пояснение: и вывод, и ошибки сохранятся в combined.log.

Задание 3. Работа с командой stdin

1. Используйте stdin для передачи данных команде grep:

#echo -e "apple\nbanana\ncherry" | grep "banana"

Объяснение: echo передаёт строки в grep через stdin.

2. Перенаправьте ввод из файла в команду:

#grep "bash" < /etc/passwd

Объяснение: grep читает /etc/passwd через перенаправленный stdin.

3. Используйте cat для чтения нескольких файлов и передачи их в конвейер:

#cat file1.txt file2.txt | sort

Пояснение: cat объединяет файлы, sort обрабатывает их через stdin.

Задание 4. Работа с командой stdout

1. Перенаправьте stdout в файл:

#echo "Hello, Linux!" > greeting.txt

Объяснение: вывод echo сохраняется в greeting.txt вместо терминала.

2. Добавьте текст в конец файла (>>):

#echo "New line" >> greeting.txt

Объяснение: текст добавляется в конец файла без перезаписи.

3. Используйте stdout в конвейере:

#ls -l | head -n 3 > first_three_files.txt

Пояснение: первые 3 строки вывода ls -l сохраняются в first_three_files.txt.

Содержание отчета

1. Титульный лист.
2. Цель работы.
3. Теоретические сведения.
4. Описание выполнения работы (по шагам).
5. Полученные результаты, скриншоты.
6. Выводы.

Контрольные вопросы

1. Что делает конвейер (|)? Приведите пример использования.

2. Как команда tee отличается от простого перенаправления (>)?

3. Как перенаправить ошибки команды в файл, игнорируя стандартный вывод?

4. Как одновременно перенаправить stdout и stderr в один файл?

5. Чем отличается > от >> при перенаправлении вывода?

6. Как записать вывод команды в файл и одновременно вывести его на экран?

7.3.8. Редактор текста VIM.

Ключевые слова

Текстовый редактор, VIM.

Цель работы

Изучить текстовой редактор VIM, научиться: создавать новый файл, открывать на редактирование существующий, сохранять файл, выходить из редактора, производить массовую замену строк в файле.

Краткие теоретические сведения

Текстовой редактор VIM — это свободно-распространяемое ПО для ОС Linux, текстовой редактор, а также среда разработки.

Редактор VIM использует интерфейс командной строки и не обладает графическим интерфейсом. Управление редактором, например, такие действия, как сохранение файла, поиск по тексту и закрытие файла, производится с помощью специальных сочетаний клавиш. При работе в редакторе использование мыши не требуется.

Редактор обладает четырьмя режимами работы:

- Нормальный режим (normal mode) — используется для команд редактора. Обычно это режим по умолчанию; нажатие клавиши ESC возвращает редактор VIM в этот режим;
- Режим вставки (insert mode) — используется для ввода текста аналогично большинству редакторов. В этом режиме открытый текст в буферах можно модифицировать текстом, введенным с клавиатуры;
- Визуальный режим (visual mode) — используется для выделения областей текста. Выбранную область можно перемещать, редактировать и т. д.;
- Режим выбора (select mode) — аналогичный визуальному, но команды не интерпретируются, вместо этого выделенный текст напрямую заменяется вводом с клавиатуры. Аналогичен режиму выделения, используемому в редакторах на платформах Microsoft Windows;
- Режим командной строки (command-line mode или cmdline mode) — обеспечивает однострочный ввод в нижней части окна Vim. Команды, начинающиеся со знака «двоеточие», и некоторые другие клавиши (например поиск по шаблону и команда фильтра)

активируют этот режим. По завершении команды VIM возвращается в предыдущий режим.

Основные командные клавиши навигации:

h - переместить курсор влево;

j - переместить курсор вниз;

k - переместить курсор вверх;

l - переместить курсор вправо;

w - переместить курсор на начало следующего слова;

b - переместить курсор на начало предыдущего слова;

e - переместить курсор на конец текущего слова;

0 - переместить курсор в начало строки;

$ - переместить курсор в конец строки;

gg - переместить курсор в начало файла;

G - переместить курсор в конец файла;

Основные командные клавиши редактирования текста;

i - вставить текст перед курсором;

a - вставить текст после курсора;

o - вставить новую строку после текущей строки и перейти в режим вставки;

dd - вырезать текущую строку;

yy - скопировать текущую строку;

p - вставить скопированный или вырезанный текст после курсора;

u - отменить последнее действие;

Ctrl + r - повторить отмененное действие;

dNd (где N - количество строк) - удаляет заданное количество строк.

Команды сохранения и закрытия файла:

:w - сохранить файл;

:q - выйти из VIM;

:wq - сохранить файл и выйти.

Команды поиска и замены подстрок в файле:

/*text* (где *text* - искомая подстрока) - поиск подстроки (переход к следующему вхождению нажатием клавиши n);

:%s/*search*/*replace*/g (где search - искомая подстрока, replace - строка замены).

Алгоритм выполнения лабораторной работы

В процессе выполнения лабораторной работы должен быть создан файл, в котором будет набран текст первой части первого тома "Война и мир".

1. В открытом доступе найдите текст первой части первого тома, скопируйте в буфер обмена;
2. Находясь в домашней директории откройте на редактирование в редакторе VIM несуществующий файл "war-and-peace";
3. Перейдите в редакторе в "Режим вставки" и вставьте текст из буфера обмена;
4. Перейдите в командный режим;
5. Сохраните файл;
6. Выйдите из редактора VIM;
7. Повторно откройте на редактирование файл;
8. Удалите первые десять строк;
9. Перейдите в "Режим вставки" и напишите случайную строку;
10. С помощью команды отмены действий отмените ввод случайной строки;
11. Выйдите из редактора VIM с отказом от сохранения изменений в файле;
12. Повторно откройте на редактирование файл;
13. С помощью команды массовой замена произведите замену по всему тексту подстроки "Анна Павловна" на "Мария Петровна";
14. Сохраните изменения в файле.

Содержание отчета

1. Титульный лист.
2. Цель работы.
3. Теоретические сведения.
4. Описание выполнения работы (по шагам).
5. Полученные результаты, скриншоты.
6. Выводы.

Контрольные вопросы

1. Какие режимы работы редактора Vim вы знаете?
2. Как перейти в "Режим вставки" из "Командного" режима?

3. Как перейти в командный режим
4. Какой командой производится сохранение файла?
5. Какой командой производится выход из редактора?
6. Как выйти из редактора без сохранения изменений, если они были произведены?
7. Как найти заданную подстроку в файле?
8. Как произвести замену подстрок в файле?

7.3.9. Редактор текста Nano.

Ключевые слова

Текстовый редактор, nano.

Цель работы

Ознакомление с текстовым редактором nano.

Краткие теоретические сведения

nano — небольшой и удобный текстовый редактор. Помимо стандартных функций терминального текстового редактора nano может выполнять отмену/возврат изменений, подсвечивать синтаксис, выполнять интерактивный поиск и замену текста и многое другое.

Чтобы открыть редактор, просто введите в команду nano.

Открыть файл для редактирования в nano можно командой:

nano [файл]

Чтобы открыть файл только для чтения, используйте параметр -v:

nano -v /etc/hello.txt

Создать резервную копию файла, можно с помощью параметра -B:

nano -B /etc/hello.txt

Nano навигация

Экран nano (рис. 88) состоит из четырех областей (рис. XX) и включает: строку заголовка (1), окно редактирования (2), строку состояния (3) и две строки справки (4).

Строка заголовка отображает версию nano, имя файла или «New Buffer», если файлу еще не было присвоено имя.

Строка состояния — третья по счету строка снизу экрана. Выводит информационные сообщения, сообщения об ошибках. Все вопросы пользователю и пользовательский ввод будут отображены в строке состояния. Две строки справки в нижней части показывают некоторые из наиболее важных функций редактора.

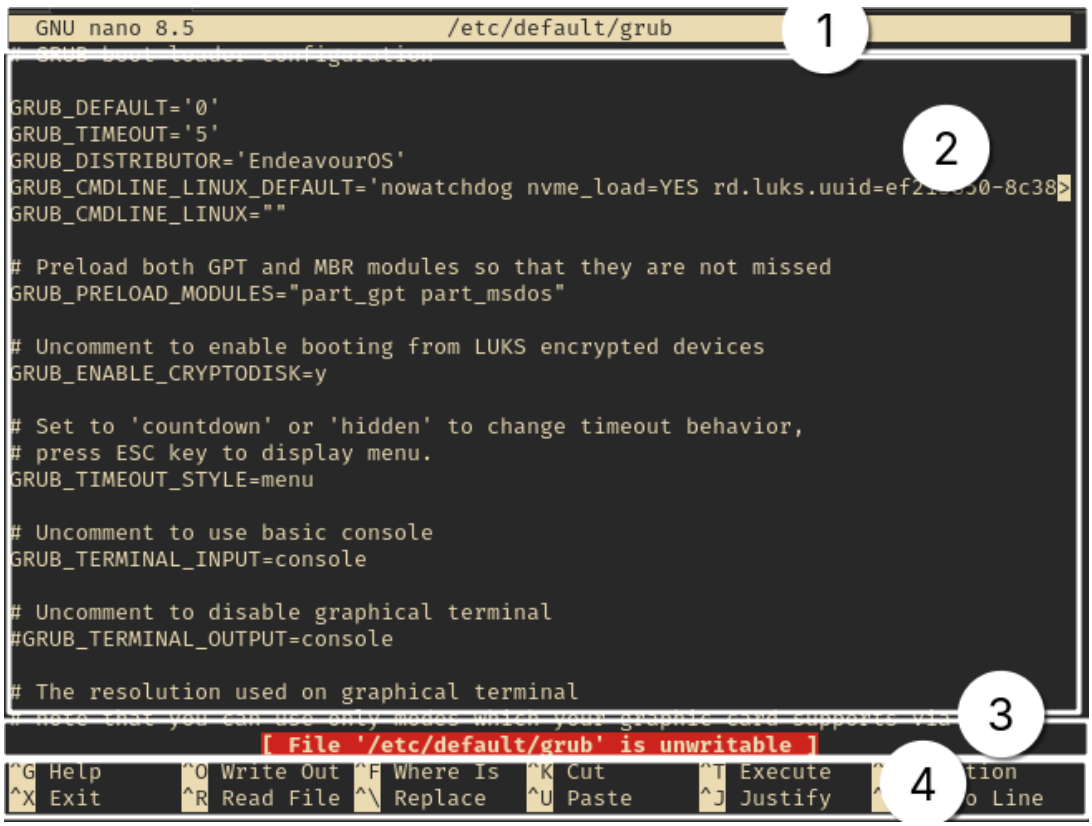


Рис. 88. Экран nano

Ввод текста и команды управления

Текст в nano вводится с клавиатуры, для ввода команд управления используются последовательности Control (клавиша Ctrl, обозначена как ^) и Meta (клавиша Alt или Cmd, обозначена как M-). Перемещение курсора выполняется с помощью стрелок.

Управляющая команда вводится нажатием нужной клавиши при удерживании клавиши Ctrl или Alt.

Если по каким-то причинами у вас не работают клавиши Ctrl и Alt, их можно заменить клавишей Esc. Вместо Ctrl один раз нажмите Esc, а затем клавишу команды, вместо Alt — два раза Esc, а затем клавишу команды.

Вырезать/копировать/вставить

В nano можно вырезать и копировать текст целыми строками. Чтобы вырезать, установите курсор в нужную строку и используйте комбинацию Ctrl+K. Вырезанная строка будет записана в буфер обрезки. После чего содержимое буфера можно будет вставить в текущую позицию курсора командой Ctrl+U. Каждый вызов Ctrl+K перезаписывает буфер обрезки.

Чтобы скопировать строку без вырезания нажмите Alt+6 и вставьте в нужное место командой Ctrl+U.

Вырезать и копировать текст можно не только строками, но и выделив произвольный текст с помощью клавиши Shift и стрелок.

Поиск

Для поиска текста в nano используются следующие команды:

- Ctrl+W для поиска начиная от курсора к концу файла;
- Ctrl+Q для поиска начиная от курсора к началу файла.

Переключение между найденными элементами осуществляется с помощью команд Alt+W и Alt+Q.

Для поиска и замены текста используется команда Alt+R.

Сохранение и выход

Работа с файлами:

- Ctrl+S — Сохранить текущий файл;
- Ctrl+O — Записать файл («Сохранить как...»);
- Ctrl+R — Вставить другой файл в текущий;
- Ctrl+X — Выйти из nano.

Алгоритм выполнения лабораторной работы

1. Скопируйте файл /etc/passwd в директорию /tmp/passwd;
2. Откройте файл /tmp/passwd с помощью редактора nano;
3. С помощью Ctrl+W или Ctrl+Q найдите имя пользователя, под которым вы зашли в систему;
4. Нажмите Ctrl+C и найдите номер строки;
5. С помощью Ctrl+K вырежьте строку вашего пользователя;
6. Переместитесь в конец файла;
7. Вставьте строку с помощью Ctrl+U.

Содержание отчета

1. Титульный лист.
2. Цель работы.
3. Теоретические сведения.
4. Описание выполнения работы (по шагам).
5. Полученные результаты, скриншоты.
6. Выводы.

Контрольные вопросы

1. В какой области можно узнать версию программы nano?
2. Какие клавиши используются для начала последовательности при управлении nano?

3. Как называется строка, в которой выводятся информационные сообщения и сообщения об ошибках?

4. Какие комбинации используются для копирования и вставки текста?

7.3.10. Управление пользователями и группами.

Ключевые слова

Команды работы с группами и пользователями.

Цель работы

Изучить основные команды для управления пользователями и группами в ОС Linux, научиться создавать, изменять и удалять учетные записи, а также управлять их правами доступа.

Краткие теоретические сведения

В ОС Linux каждый пользователь имеет уникальную учетную запись с определенными правами. Администратор (root) может управлять пользователями и группами с помощью специальных команд.

- Основные команды для работы с пользователями:
- useradd – создать нового пользователя;
- usermod – изменить параметры пользователя;
- userdel – удалить пользователя;
- passwd – установить или изменить пароль.
- Работа с группами:
- groupadd – создать новую группу;
- groupmod – изменить параметры группы;
- groupdel – удалить группу;
- gpasswd – управлять участниками группы.
- Просмотр информации:
- id – показать UID, GID и группы пользователя;
- whoami – вывести текущего пользователя;
- getent passwd – список всех пользователей;
- getent group – список всех групп.
- Права доступа и sudo:
- sudo – выполнить команду с правами root;
- visudo – редактировать файл /etc/sudoers.

Алгоритм выполнения лабораторной работы

Задание 1. Создание и настройка пользователя

1. Создайте нового пользователя testuser:

```
sudo useradd testuser
```

2. Установите пароль для testuser:

```
sudo passwd testuser
```

3. Проверьте информацию о пользователе:

id testuser

4. Добавьте пользователя в группу sudo (для прав администратора):

sudo usermod -aG sudo testuser

Задание 2. Работа с группами

1. Создайте новую группу developers:

sudo groupadd developers

2. Добавьте testuser в группу developers:

sudo usermod -aG developers testuser

3. Проверьте состав группы:

getent group developers

Задание 3. Удаление пользователя и группы

1. Удалите пользователя testuser (без удаления домашней директории):

sudo userdel testuser

2. Удалите пользователя вместе с домашней директорией:

sudo userdel -r testuser

3. Удалите группу developers:

sudo groupdel developers

Задание 4. Права sudo

1. Откройте файл /etc/sudoers для редактирования:

sudo visudo

2. Добавьте строку для разрешения testuser выполнять любые команды:

text

testuser ALL=(ALL:ALL) ALL

3. Сохраните изменения и выйдите.

Содержание отчета

1. Титульный лист.
2. Цель работы.
3. Теоретические сведения.
4. Описание выполнения работы (по шагам).
5. Полученные результаты, скриншоты.
6. Выводы.

Контрольные вопросы

1. Как создать нового пользователя в Linux?
2. Как добавить пользователя в группу?
3. Какая команда показывает UID и GID пользователя?
4. Как удалить пользователя вместе с домашней папкой?
5. Как дать пользователю права root через sudo?

7.3.11. Управление разрешениями.

Ключевые слова

Команды chmod, chown, umask, chattr, lsattr.

Цель работы

Изучить права доступа к файлам и научиться управлять распределением прав на файлы.

Краткие теоретические сведения

Права доступа

Права доступа в дистрибутивах семейства Unix являются их неотъемлемой частью, которая обеспечивает управление безопасностью и распределением прав.

Как правило, модели доступа можно условно разделить на 4 типа:

1. *DAC* – Discretionary Access Control (Дискреционный контроль доступа) – механизм контроля доступа, который позволяет владельцу ресурсу настраивать доступ к нему, определяя кто может выполнять определённые действия с файлом, а кто нет.
2. *MAC* – Mandatory Access Control (Мандатное управление доступом) – доступ к объектам осуществляется на основе уровня конфиденциальности (точнее сказать определённой метки). За доступом следит центральное приложение (в большинстве случаев его работа основана на модулях ядра).

Пользователь с более высоким уровнем конфиденциальности может иметь право на чтение документа с более низким уровнем конфиденциальности. Читать документы пользователю с уровнем доступа ниже, чем доступ к файлу не представляется возможным.

Пользователь имеет право на изменение документа с уровнем конфиденциальности, совпадающим или меньшим, чем его уровень конфиденциальности.

Пример сравнения моделей DAC и MAC (рис. 89):

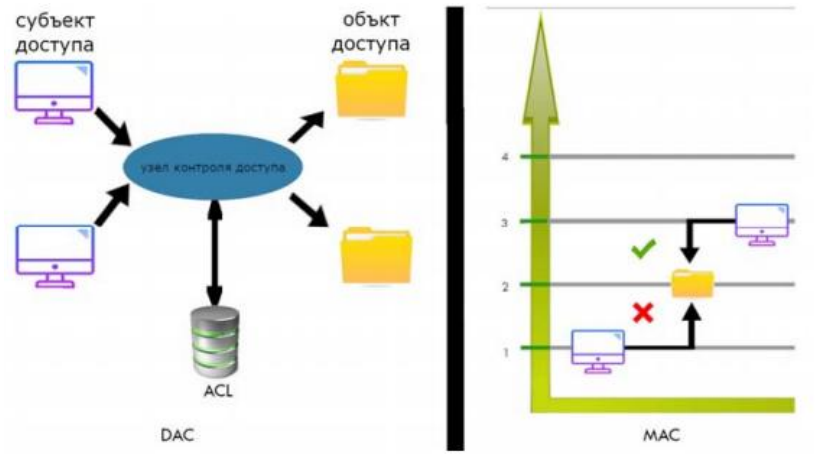


Рис. 89. Пример сравнения моделей DAC и MAC

3. *ABAC* – Attribute-Based Access Control (модель доступа на основе атрибутов) -данный механизм основывается на определённых атрибутах пользователя, файла или окружения. В отличие от модели доступа DAC, используются свойства определённого объекта или субъекта.

4. *RBAC* – Role-Based Access Control (модель доступа на основе ролей). В основе данной системы лежит механизм управления ролями определённых групп или пользователей. При настройке данной системе возможна выдача определённого функционала вышестоящих ролей нижестоящим.

Изменение разрешений chmod (числовой и буквенный методы)

С помощью команды chmod можно изменить права доступа к файлу. Это чтение, запись и выполнение. Каждый пользователь может изменять права для своих файлов.

К примеру, вы являетесь владельцем файла с именем new_file и хотите установить его разрешения таким образом, чтобы: пользователь мог читать, писать и выполнять его; члены группы могут прочитать и выполнить его; а также другие могут только читать его. Эта команда будет выглядеть вот так.

Пример:

$ *chmod u=rwx,g=rx,o=r new_file*

В качестве альтернативы можно использовать запись в восьмеричном виде для обозначения прав.

Пример:

$ *chmod 754 new_file*

Также для добавления или удаления каких-либо прав можно использовать знаки ‘+’ и ‘-‘ соответственно.

Например, команда для добавления права на исполнение пользователю и группы будет выглядеть так.

Пример:

$ *chmod ug+x new_file*

Числовые значения прав доступа в ОС Linux показаны на рис. 90:

OCT	BIN	Mask	Комментарий
0	000	---	отсутствие прав
1	001	--x	права на выполнение
2	010	-w-	права на запись
3	011	-wx	права на запись и выполнение
4	100	r--	права на чтение
5	101	r-x	права на чтение и выполнение
6	110	rw-	права на чтение и запись
7	111	rwx	полные права

Рис. 90. Числовые значения прав доступа в ОС Linux

Также команда chmod может использовать 4 цифры, где первая цифра будет означать использование битов suid, sgid, sticky.

Suid – при установки этого бита запущенная программа будет работать от имени владельца исполняемого файла. ($ *chmod u+s file*)

Sgid – выполнение программы от имени группы. В случае с директориями все объекты внутри неё наследуют группу директории. ($ *chmod g+s dir*)

Sticky bit – при установке данного бита на директорию пользователи смогут модифицировать/удалять только те файлы, владельцами которых они являются.

Пример:

$ *chmod +t dir*

Таблица со специальными правами и первой цифрой из четырёх в команде chmod (рис. 91):

suid	sgid	Sticky bit	цифра
+	+	+	7
+	+	-	6
+	-	+	5
+	-	-	4
-	+	+	3
-	+	-	2
-	-	+	1

-	-	-	0

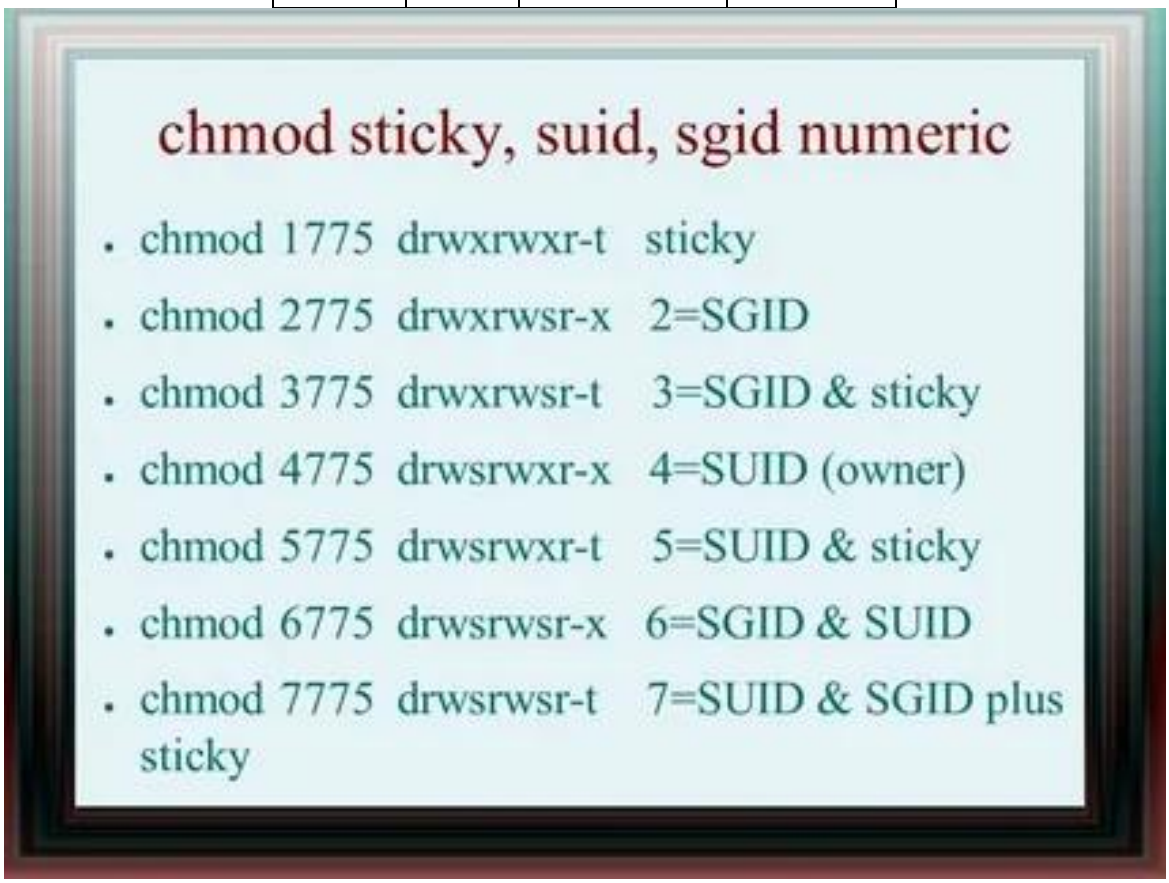


Рис. 91. Пример реализации прав доступа

Изменение владельца chown

Помимо изменения прав доступа к файлам, вы также можете изменить права доступа к файлам для групп и пользователей.

Чтобы поменять владельца воспользуемся командой chown (change owner). Но даже если мы владелец этого файла, без прав суперпользователя мы это сделать не сможем. Потому что если какой-то пользователь создаст файл, а потом укажет, что владельцем файла является другой пользователь, то он сможет подставить другого пользователя. Или, например, использовать выделенное для второго пользователя место в своих целях, когда каждому пользователю выделено сколько-то места на диске. Поэтому чтобы поменять владельца, нужно использовать sudo (рис. 92):

Пример:

ll file

sudo chown user4 file

ll file

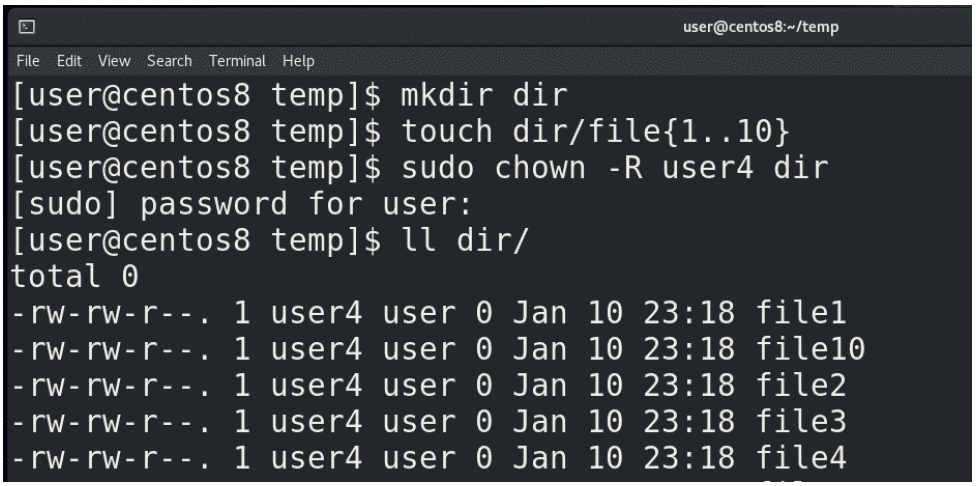


Рис. 92. Изменение владельца chown

Если мы хотим это сделать для всех файлов в директории и во всех поддиректориях, вы уже знаете, для этого нужно делать рекурсивно (рис. 93).

Пример:

mkdir dir

touch dir/file{1..10}

sudo chown -R user4 dir

ll dir

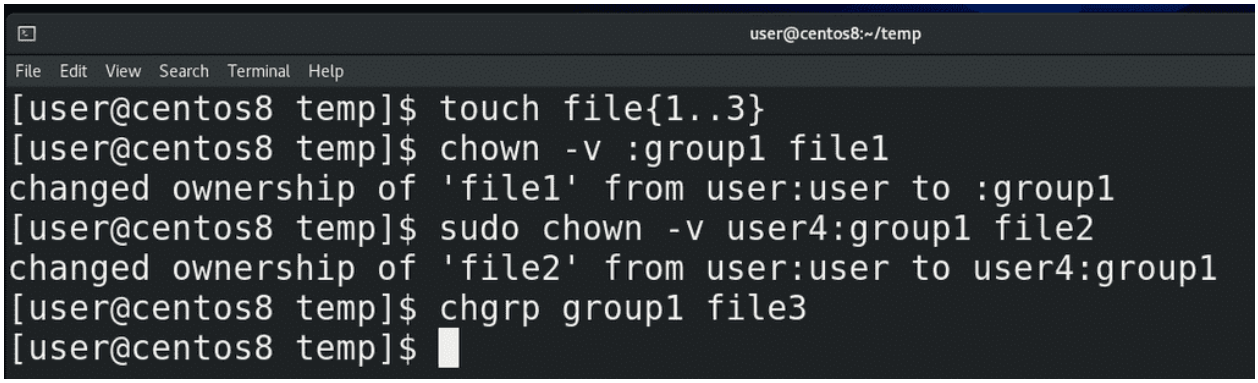


Рис. 93. Изменение владельца в директории

Группу также можно поменять с помощью chown.

Пример:

touch file{1..3}

chown -v :group1 file1

Здесь ключ -v - verbose - только для вывода информации. Можно сменить разом владельца и группу.

Пример:

sudo chown -v user:user file2

Разрешение для процесса

Разберемся с правами доступа к процессу. Если запустить команду passwd с включенным битом разрешения SUID, то

запускается программа от имени пользователя root. При этом нельзя изменять пароли других пользователей. Это связано с большим количеством UID, которые реализует Linux.

С каждым процессом связаны три UID:

Когда вы запускаете процесс, он запускается с теми же правами, что и у пользователя или группы, которые его запустили, это называется эффективным идентификатором пользователя. Этот UID используется для предоставления прав доступа к процессу. Поэтому, естественно, если один пользователь выполнит команду touch, процесс будет запущен от его имени, и все созданные им файлы будут принадлежать ему.

Есть еще один UID, называемый реальным идентификатором пользователя, это идентификатор пользователя, который запустил процесс. Они используются для отслеживания того, кто является пользователем, запустившим этот процесс.

Последний UID — это сохраненный идентификатор пользователя, который позволяет процессу переключаться между эффективным UID и реальным UID и наоборот. Это полезно, потому что мы не хотим, чтобы наш процесс постоянно запускался с повышенными привилегиями, просто рекомендуется использовать специальные привилегии в определенное время.

Соберем все это воедино, еще раз обратившись к команде passwd.

При выполнении команды passwd эффективный UID — это ваш идентификатор пользователя, допустим, на данный момент он равен 500. В то же время для команды passwd включено разрешение SUID. Таким образом, когда вы запускаете ее, ваш эффективный UID теперь равен 0 (0 — это UID пользователя root). Теперь эта программа может получать доступ к файлам от имени пользователя root.

Если попытаться изменить пароль с UID, равный 600, то у этого процесса также есть ваш реальный UID, в данном случае 500. Он знает, что ваш UID равен 500, и поэтому изменить пароль с UID равным 600 не получится.

Поскольку вы запустили passwd, он запустит процесс, используя ваш реальный UID, и сохранит UID владельца файла (действующий UID), чтобы вы могли переключаться между ними. Нет необходимости изменять все файлы с правами root, если это не требуется.

В большинстве случаев реальный UID и эффективный UID совпадают, но в таких случаях, как команда passwd, они будут меняться.

Утилита umask

Каждый создаваемый файл имеет набор разрешений по умолчанию. Если вы когда-нибудь захотите изменить этот набор разрешений по умолчанию, вы можете сделать это с помощью команды umask. Эта команда использует 3-разрядный набор разрешений, который мы видим в числовых разрешениях.

Однако вместо того, чтобы добавлять эти разрешения, umask удаляет их.

Пример:

$ umask 021

В приведенном выше примере мы заявляем, что хотим, чтобы разрешения по умолчанию для новых файлов предоставляли пользователям доступ ко всему, но для групп мы хотим лишить их прав на запись, а для других мы хотим лишить их прав на исполняемый файл. Маска обмена данными по умолчанию в большинстве дистрибутивов равна 022, что означает доступ для всех пользователей, но отсутствие доступа на запись для группы и других пользователей.

Когда вы запустите команду umask, она предоставит этот набор разрешений по умолчанию для любого создаваемого вами файла.

Атрибуты файла chattr, lsattr

Предположим, необходимо защитить некоторые важные файлы в ОС Linux. При чем они должны быть защищены не только от перезаписи, но и от случайного или преднамеренного удаления и перемещения. Предотвратить перезапись или изменение битов доступа к файлам можно с помощью стандартных утилит chmod и chown, но у суперпользователя по-прежнему остается полный доступ к этим файлам.

Решение – команда **chattr**.

Эта утилита позволяет устанавливать и отключать атрибуты файлов, на уровне файловой системы независимо от стандартных (чтение, запись, выполнение). Для просмотра текущих аттрибутов можно использовать **lsattr**. Изначально атрибуты управляемые

chattr и lsattr поддерживались только файловыми системами семейства ext (ext2, ext3, ext4), но теперь эта возможность доступна и в других популярных файловых системах таких как XFS, Btrfs, ReiserFS, и т д.

Утилиты chattr и lsattr предустановлены во всех современных дистрибутивах. Базовый синтаксис chattr выглядит следующим образом:

$ chattr опции [оператор][атрибуты] файлы

Вот основные опции утилиты, которые вы можете использовать:

-R - рекурсивная обработка каталога;

-V - максимально подробный вывод;

-f - игнорировать сообщения об ошибках;

-v - вывести версию.

Пользователь может принимать значения:

+ - включить выбранные атрибуты;

- - отключить выбранные атрибуты;

= - оставить значение атрибута таким, каким оно было у файла.

Вот некоторые доступные атрибуты:

a - файл может быть открыт только в режиме добавления;

A - не обновлять время перезаписи;

c - автоматически сжимать при записи на диск;

C - отключить копирование при записи;

D - работает только для папки, когда установлен, все изменения синхронно записываются на диск сразу же;

e - использовать extent'ы блоков для хранения файла;

i - сделать неизменяемым;

j - все данные перед записью в файл будут записаны в журнал;

s - безопасное удаление с последующей перезаписью нулями;

S - синхронное обновление, изменения файлов с этим атрибутом будут сразу же записаны на диск;

t - файлы с этим атрибутом не будут хранится в отдельных блоках;

u - содержимое файлов с этим атрибутом не будет удалено при удалении самого файла и потом может быть восстановлено.

Алгоритм выполнения лабораторной работы

1. Изучите права к файлам с помощью команды **ls -l** (рис. 94).

Пример:

$ *ls -l <file>* (имя file в команде опционально -a показать скрытые файлы)

Рис. 94. Права к файлам

2. Измените некоторые базовые права доступа к текстовым файлам и посмотрите, как меняются разряды при выполнении ls -l. Зафиксируйте результаты изменений.

3. Измените группу и пользователя для некоторых тестовых файлов. Затем проверьте права доступа с помощью ls -l. Зафиксируйте результаты изменений.

4. Далее откройте одно окно терминала и выполните команду: watch -n 1 "ps aux | grep passwd". Это позволит отслеживать процесс passwd.

5. Откройте второе окно терминала и выполните команду: passwd.

6. Посмотрите на первое окно терминала, вы увидите, что для ввода пароля появляется процесс. Первый столбец в таблице процессов — это действующий идентификатор пользователя.

7. Теперь создайте новый файл, затем обратите внимание на его права доступа.

8. Измените umask, а затем создайте другой новый файл.

9. Сравните права доступа для двух созданных файлов. Зафиксируйте результаты изменений.

10. Теперь посмотрим текущие атрибуты любого текстового файла.

11. Сделайте этот файл неизменяемым (для этого нужно иметь права суперпользователя для установки и удаления атрибутов).

12. Проверьте атрибуты файла. Зафиксируйте результаты изменений.

Содержание отчета

1. Титульный лист.
2. Цель работы.
3. Теоретические сведения.
4. Описание выполнения работы (по шагам).
5. Полученные результаты, скриншоты.
6. Выводы.

Контрольные вопросы

1. Какое число соответствует разрешению на чтение при использовании цифрового формата?
2. Какое число соответствует разрешению на запись при использовании цифрового формата?
3. Какую команду вы используете для изменения владельца файлов для групп?
4. Какой UID определяет, какой доступ предоставлять?
5. Какая команда используется для изменения прав доступа по умолчанию к файлам?
6. Назовите назначения команд chattr и lsattr?

7.3.12. Изучение процессов.

Ключевые слова

Команды ps, ps aux, top, kill.

Цель работы

Изучить основные понятия, связанные с процессами в ОС Linux. Научиться управлять процессами с помощью командной строки, а также понимать их состояние, приоритеты и взаимодействие с ОС.

Краткие теоретические сведения

Процессы — это программы, выполняющиеся в ОС. Каждый процесс имеет уникальный идентификатор (PID), который присваивается ядром в порядке создания процессов.

Команда ps

Выполните команду ps, чтобы увидеть список запущенных процессов:

Пример вывода команды:

$ **ps**

```
PID    TTY    STAT   TIME       CMD
41230  pts/4  Ss     00:00:00   bash
51224  pts/4  R+     00:00:00   ps
```

Основные поля вывода команды ps:

PID: Идентификатор процесса.

TTY: Терминал, связанный с процессом.

STAT: Состояние процесса.

TIME: Время использования CPU.

CMD: Команда, запустившая процесс.

В справочной странице ps (команда man ps) вы увидите множество опций командной строки.

Пример команды для просмотра всех процессов:

$ ps aux

Опции команды:

-a – показать процессы всех пользователей;

-u – вывести подробную информацию;

–x – включить процессы без связанного терминала (например, демоны).

Теперь можно увидеть больше полей.

Перечислим описания основных полей:

USER: Эффективный пользователь (от чьего имени выполняется доступ);

PID: Идентификатор процесса;

%CPU: Использование CPU относительно времени работы процесса;

%MEM: Доля резидентной памяти процесса от общей физической памяти;

VSZ: Виртуальная память, используемая процессом;

RSS: Резидентный размер (физическая память без подкачки);

TTY: Управляющий терминал;

STAT: Код состояния;

START: Время запуска;

TIME: Общее время CPU;

COMMAND: Исполняемая команда.

Как можно увидеть, команда ps может выводить много информации, но основные поля для анализа: PID, STAT и COMMAND.

Команда top

Еще одна полезная команда - top, которая предоставляет информацию о процессах в реальном времени (обновляется каждые 10 секунд по умолчанию). Это мощный инструмент для анализа загрузки системы.

$ **top**

Управляющий терминал (TTY)

TTY — это терминал, управляющий процессом. Существует два типа терминалов:

Регулярные терминалы (например, виртуальные консоли, доступные через Ctrl-Alt-F1);

Псевдотерминалы (PTS), эмулирующие терминалы в графической среде.

Процессы обычно привязаны к управляющему терминалу. Например, если закрыть окно терминала с выполняющейся командой find, процесс также завершится.

Демоны - специальные процессы, работающие в фоне. Они запускаются при загрузке системы и обычно не привязаны к терминалу (TTY отображается как ?).

Детали процессов

Процесс - это экземпляр выполняющейся программы, для которой система выделяет память, CPU и другие ресурсы. Например, если запустить команду cat в двух терминалах и выполнить ps aux | grep cat, вы увидите два разных процесса.

Ядро отвечает за некоторые процессы:

- Отслеживает их состояние;
- Управляет выделением ресурсов;
- Контролирует владельца;
- Обрабатывает сигналы.

Когда процесс завершается, его ресурсы освобождаются.

Создание и завершение процессов

Создание: Процессы создаются через системный вызов fork. Дочерний процесс получает новый PID, а родительский - PPID (Parent PID). Затем дочерний процесс может либо продолжать выполнение родительской программы, либо (чаще) запускать новую через execve().

Пример:

$ **ps l**

Опция l показывает расширенный формат с полем PPID. PID вашего shell будет совпадать с PPID команды ps.

"Матерью всех процессов" является init (PID 1), который создается при загрузке системы и порождает все остальные процессы.

Завершение: Процесс может завершиться через _exit(), освобождая ресурсы. При завершении процесс сообщает ядру статус (0 обычно означает успех). Родительский процесс должен подтвердить завершение через wait().

Также существуют особые случаи.

Процессы-сироты: когда родитель завершается раньше потомка. Ядро передает их init.

Зомби-процессы: завершенные процессы, ожидающие обработки родителем. Занимают место в таблице процессов, но не потребляют ресурсов. Удаляются после вызова wait() (reaping).

Сигналы — это уведомления процессу о событиях (программные прерывания).

Используются для:

- Обработки Ctrl-C, Ctrl-Z;
- Уведомления об аппаратных/программных ошибках;
- Межпроцессного взаимодействия.

Процесс может:

- Игнорировать сигнал;
- Перехватить и обработать;
- Завершиться;
- Блокировать (кроме SIGKILL).

Основные сигналы:

- SIGHUP(1): Разрыв соединения;
- SIGINT(2): Прерывание (Ctrl-C);
- SIGKILL(9): Немедленное завершение;
- SIGTERM(15): Запрос на завершение;
- SIGSTOP: Приостановка.

Команда kill

Команда kill в ОС Linux — инструмент для завершения процессов. Она отправляет процессу сигнал, который предписывает ему завершить работу.

Пример отправки сигнала:

```
$ kill -9 PID
```

Пример:

```
$ kill 12445      # SIGTERM (по умолчанию)
$ kill -9 12445   # SIGKILL
```

Различия сигналов:

- SIGHUP: При закрытии терминала;
- SIGINT: Graceful завершение (Ctrl-C);
- SIGTERM: Запрос завершения с очисткой;
- SIGKILL: Немедленное завершение;
- SIGSTOP: Приостановка.

Приоритеты процессов (nice)

Приоритет процесса определяется его "вежливостью" (nice value):

Высокое значение (например, 10) — низкий приоритет;

Низкое или отрицательное значение (например, -5) — высокий приоритет;

Команды для изменения приоритета:

$ nice -n 5 apt upgrade # Установка приоритета для нового процесса

$ renice 10 -p 3245 # Изменение приоритета существующего процесса

Состояния процессов

Основные состояния (STAT):

- R: Выполняется или готов к выполнению;
- S: Ожидание (прерываемое);
- D: Ожидание (непрерываемое);
- Z: Зомби-процесс;
- T: Остановлен.

Файловая система /proc

Информация о процессах хранится в /proc. Каждый процесс имеет подкаталог с именем, равным его PID.

Пример просмотра информации о процессе:

$ cat /proc/12345/status

Управление задачами (jobs)

Для запуска в фоне необходимо добавить & к команде:

$ sleep 1000 &

Просмотр задач:

$ Jobs

Перемещение между фоном и передним планом:

$ fg %1 # Переместить задачу 1 на передний план;

$ bg %1 # Переместить задачу 1 в фон.

Завершение задачи:

$ kill %1

Алгоритм выполнения лабораторной работы

Задание 1. Изучение процессов

1. Запустите команду ps и ps aux. Запишите PID и состояние нескольких процессов;
2. Найдите процессы без связанного терминала (TTY = ?).

Задание 2. Управление процессами

1. Запустите процесс sleep 1000 в фоне;
2. Приостановите процесс с помощью Ctrl-Z и возобновите его в фоне (bg);

3. Завершите процесс с помощью kill.

Задание 3. Приоритеты

1. Запустите команду top и найдите столбец NI (nice value);
2. Измените приоритет одного из процессов с помощью renice.

Задание 4. Сигналы

1. Запустите sleep 1000;
2. Отправьте ему сигнал SIGINT (Ctrl-C) и SIGKILL (через kill -9).

Задание 5. Изучение /proc

1. Найдите PID процесса bash с помощью ps;
2. Изучите содержимое /proc/PID/status.

Содержание отчета

1. Титульный лист.
2. Цель работы.
3. Теоретические сведения.
4. Описание выполнения работы (по шагам).
5. Полученные результаты, скриншоты.
6. Выводы.

Контрольные вопросы

1. Что такое PID и PPID?
2. Какие состояния может иметь процесс?
3. Как завершить процесс, который не реагирует на SIGTERM?
4. Что такое зомби-процесс и как его устранить?
5. Как изменить приоритет процесса?

7.3.13. Установка программ и пакетов.

Ключевые слова

Цель работы

Освоить методы установки программ в ОС Linux. Изучить работу с пакетным менеджером APT. Научиться устанавливать программы из исходного кода. Понять принципы работы с репозиториями и зависимостями.

Краткие теоретические сведения

В ОС Linux программы распространяются в виде:

- .deb пакетов - готовых к установке бинарных файлов;
- Исходного кода (обычно .tar.gz, .tar.xz) - требующих компиляции;
- Скриптов установки (.sh, .run).

Пакетный менеджер APT

APT (Advanced Package Tool) - основной инструмент управления пакетами в Debian.

Установка .deb пакетов

.deb - родной формат пакетов Debian.

Репозитории Debian

Репозитории - хранилища пакетов, откуда ОС получает программы.

Зависимости пакетов

Зависимости - дополнительные пакеты, необходимые для работы программы.

Работа с архивами

- .tar - архив без сжатия;
- .tar.gz - сжатие gzip;
- .tar.xz - сжатие xz.

Установка из исходников

Этапы установки:

1. Установка инструментов компиляции;
2. Распаковка архива;
3. Конфигурация;
4. Компиляция;
5. Установка.

Алгоритм выполнения лабораторной работы

Задание 1.

1. Проверьте установленные пакеты:

dpkg -l | less

2. Найдите конкретный пакет:

dpkg -l | grep nano

Задание 2.

1. Обновление списка пакетов:

#sudo apt update

2. Установка пакета (на примере htop):

#sudo apt install htop

3. Удаление пакета:

#sudo apt remove htop

Задание 3.

1. Установка пакета вручную:

#sudo dpkg -i package.deb

2. Если возникли проблемы с зависимостями:

#sudo apt install -f

Задание 4.

1. Просмотр списка репозиториев:

#cat /etc/apt/sources.list

2. Добавление репозитория (например, multiverse):

#sudo add-apt-repository multiverse

#sudo apt update

Задание 5.

1. Установка с автоматическим разрешением зависимостей:

#sudo apt install -f

2. Удаление ненужных зависимостей:

#sudo apt autoremove

Задание 6.

1. Распаковка архива:

tar -xvf archive.tar.gz

2. Создание архива:

tar -czvf backup.tar.gz /path/to/dir

Задание 7.

1. Установка build-essential:

sudo apt install build-essential

2. Установка из исходников:

#wget http://example.com/program.tar.gz
#tar -xvf program.tar.gz
#cd program/
#./configure
#make
#sudo make install

Содержание отчета

1. Титульный лист.
2. Цель работы.
3. Теоретические сведения.
4. Описание выполнения работы (по шагам).
5. Полученные результаты, скриншоты.
6. Выводы.

Контрольные вопросы

1. Какой пакетный менеджер используется по умолчанию?
2. Как обновить список доступных пакетов?
3. Как установить .deb пакет вручную?
4. Где хранится список репозиториев?
5. Как добавить новый репозиторий?
6. Как автоматически устранить проблемы с зависимостями?
7. Какие команды используются для работы с архивами?
8. Как правильно установить программу из исходников?
9. Как удалить ненужные зависимости?

8. Интерфейс виртуальной лаборатории PNetLab

PNETLab (Packet Network Emulator Tool Lab) – это эмулятор сетевых лабораторий, в которых можно конструировать свои сети и разнообразные сложные виртуальные инфраструктуры.

Порядок входа в эмулятор PnetLab:

1. Выполнить авторизацию в среде эмулирования (рис. 95). Данные для входа есть в разделе 11 Методических указаний.

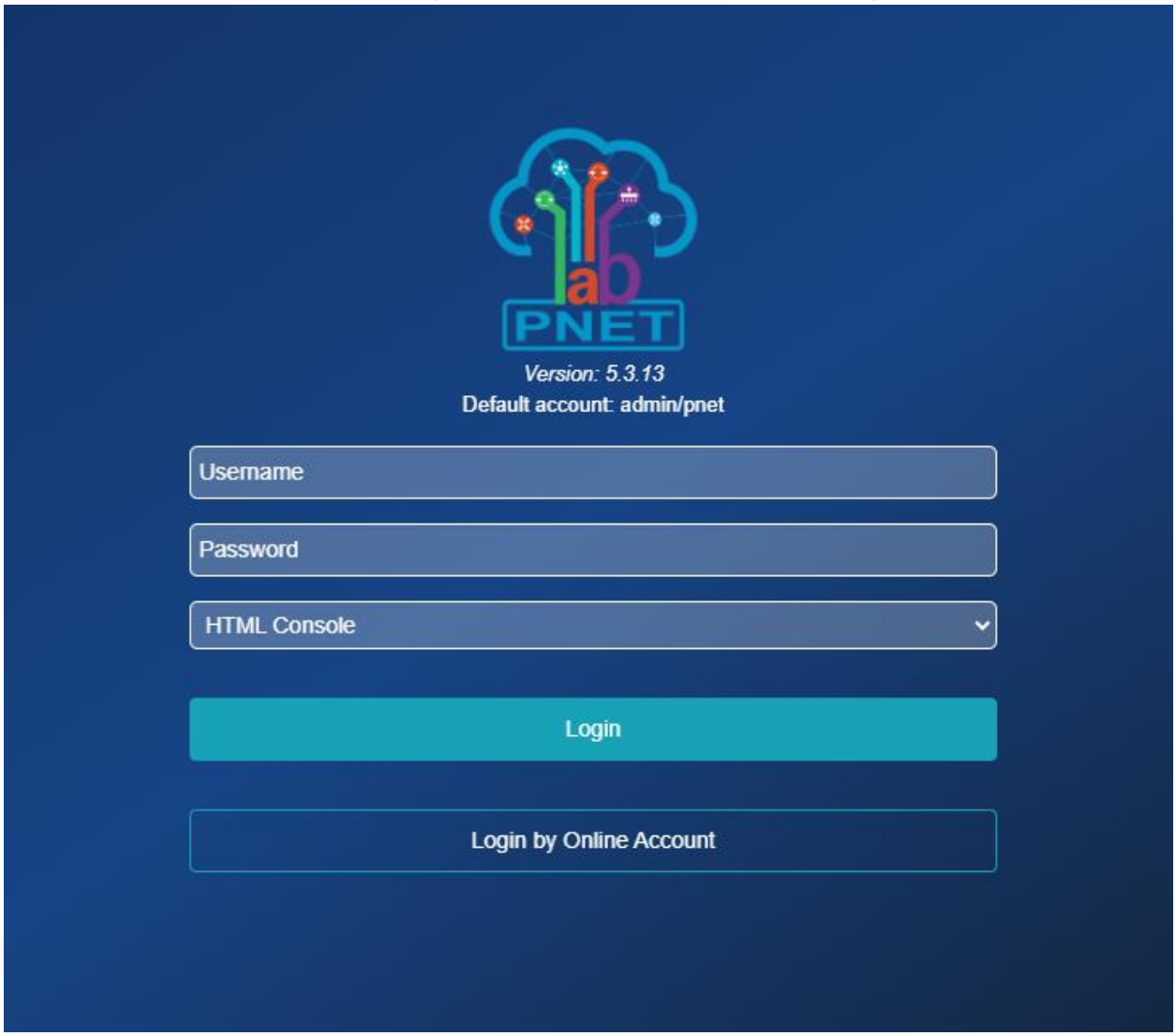


Рис. 95. Окно авторизации

2. В появившемся окне создаем новую папку (рис. 96).

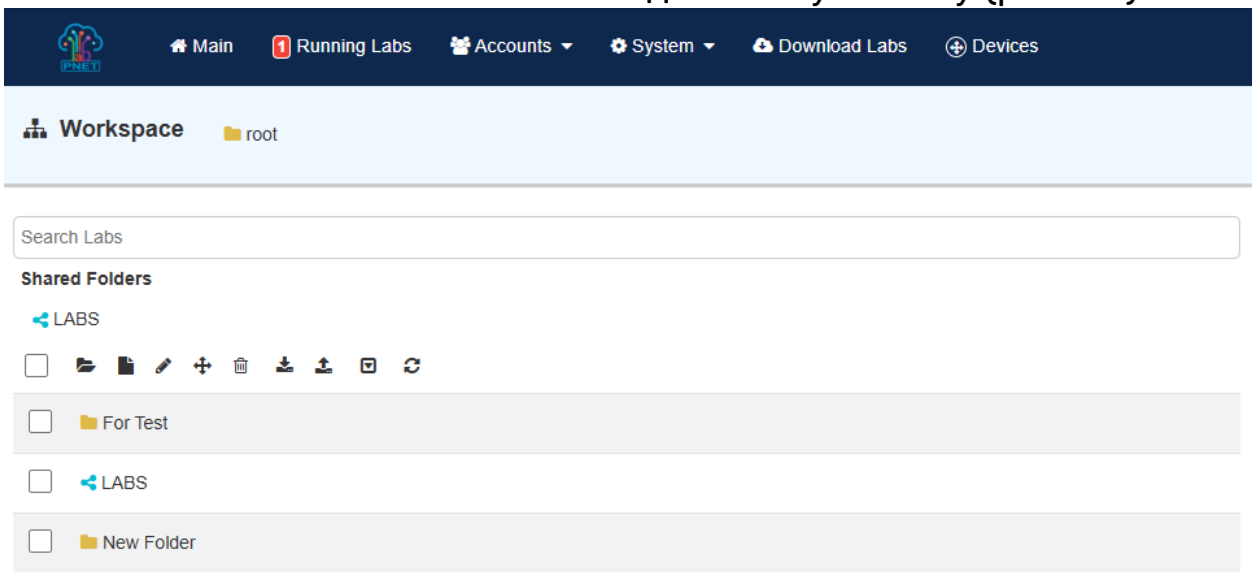


Рис. 96. Создание новой папки

3. Заходим в папку и создаем лабораторную работу (рис. 97).

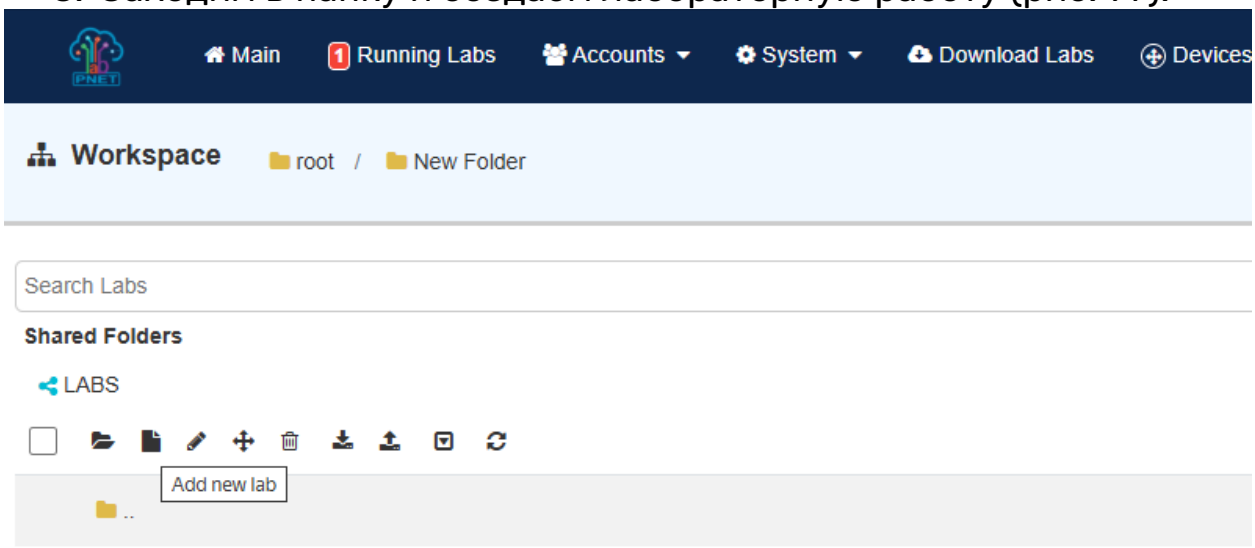


Рис. 97. Создание лабораторной работы

4. На данном этапе указываем наименование лабораторной работы и права пользования (рис. 98).

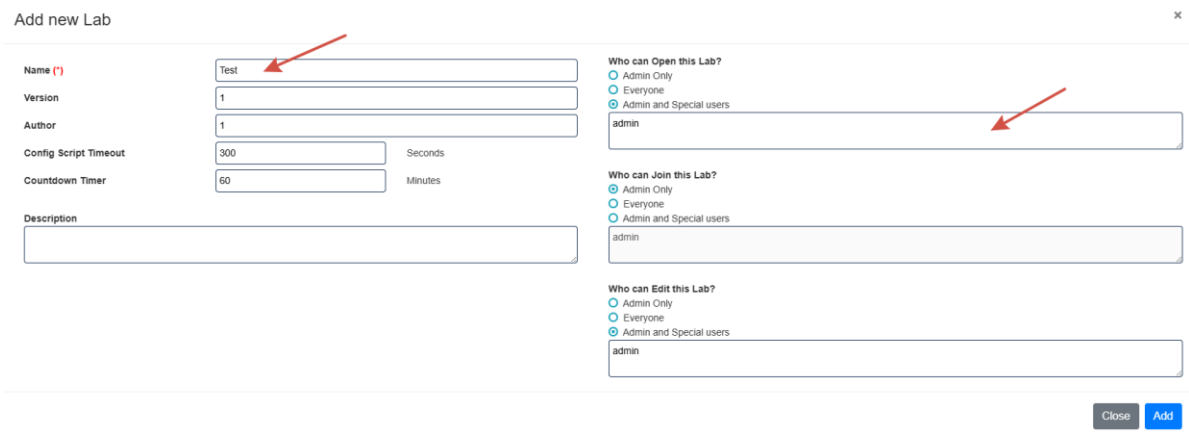


Рис. 98. Заполнение полей лабораторной работы

5. Выбираем пользователей, у кого будет доступ, и жмем кнопку «Добавить» (Add) (рис. 99).

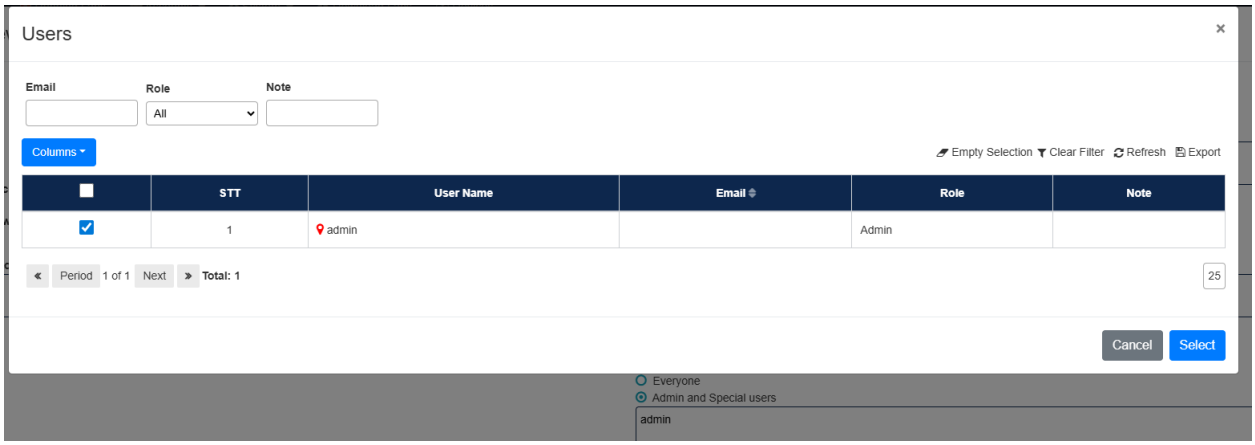


Рис. 99. Выбор пользователей

6. Слева во всплывающем меню нажимаем добавить объект и выбираем требуемый из списка (для примера сделаем виртуальную машину на ОС Linux, **Nodes-> Linux**) (рис. 100).

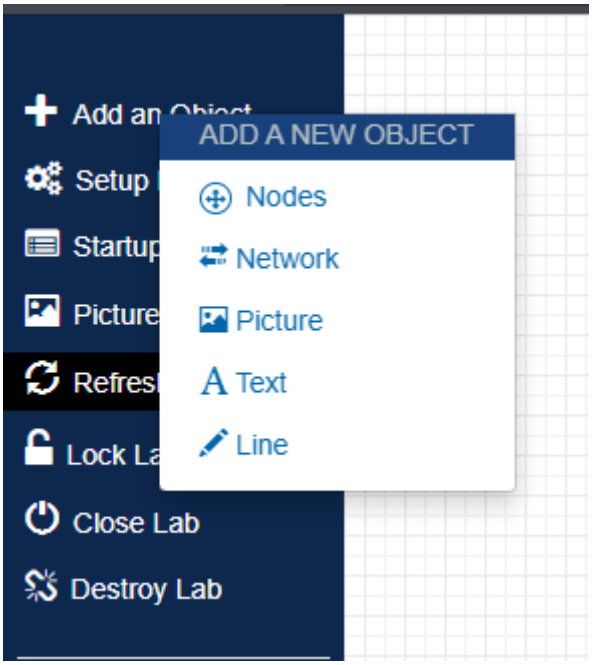


Рис. 100. Добавление объекта

7. В появившимся окне выбираем нужные нам характеристики нашей виртуальной машины.

8. Далее создаем мост для сети (Network->bridge).

9. Объединяем наш мост и виртуальную машину (рис. 101).

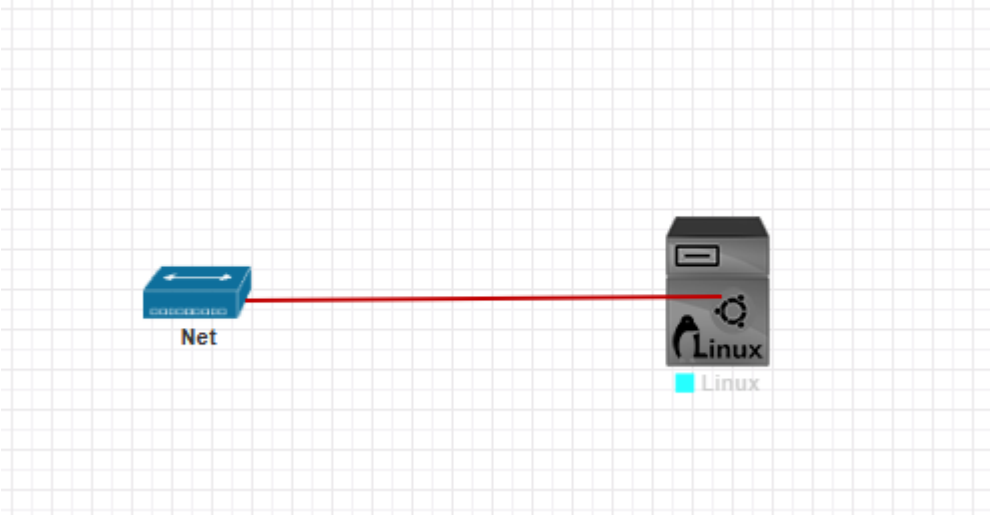


Рис. 101. Объединение моста и виртуальной машины

10. Таким образом можно объединить виртуальные машины в одну сеть (рис. 102).

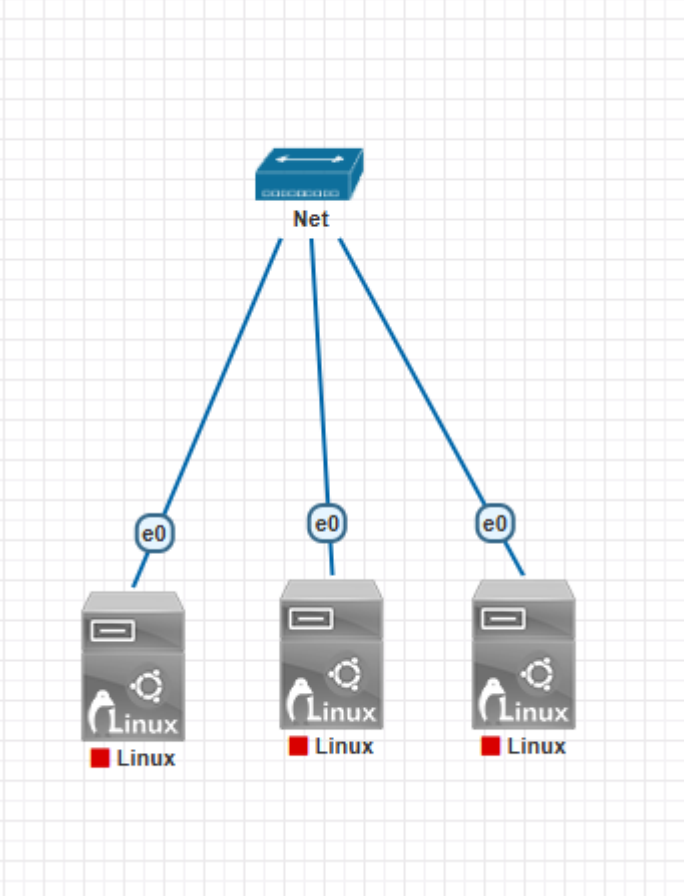


Рис. 102. Объединение виртуальных машин в одну сеть

11. Виртуальная машина запускается левой кнопкой мыши, при повторном нажатии открывается терминал операционной системы (рис. 103).

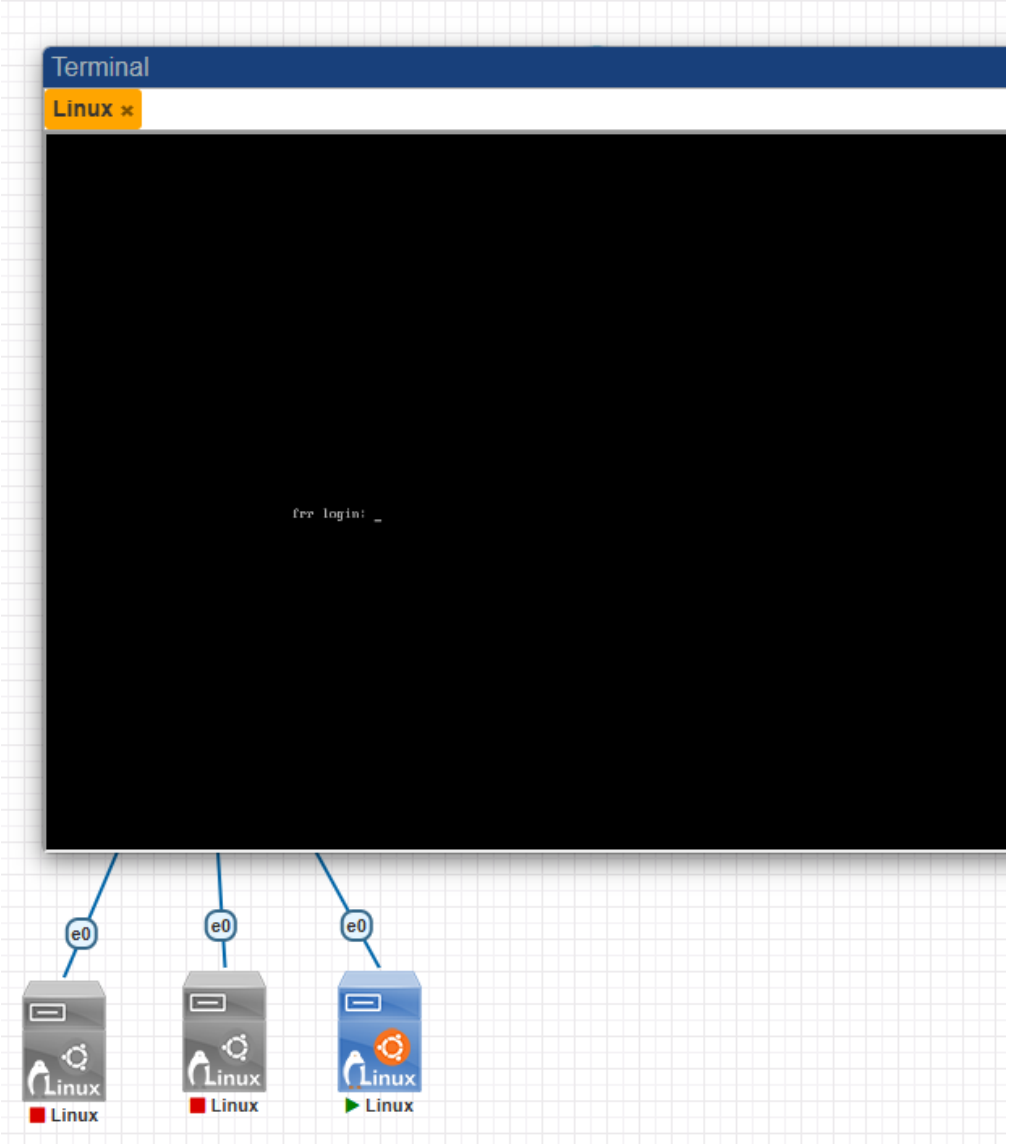


Рис. 103. Терминал операционной системы

9. Перечень оборудования Лаборатории

№	Наименование	Обозначение на сетевой схеме	Место установки	Серийный номер	MAC
1	Центральный сервер измерений IRU Rock S1204P	ЦСИ	ЦОД	2031224HAL0017	00:25:90:FB:3F:C8
2	Маршрутизатор Cisco 1921	Router	ТШ	FCZ1620206A	A4:4C:11:4E:D4:C0
3	Маршрутизатор Cisco 2801	CPE3	ТШ	FCZ112723JC	00:1E:13:74:5F:99
4	Маршрутизатор Cisco 2801	CPE4	ТШ	FCZ114591SV	00:1B:D5:A5:E6:E9
5	Коммутатор доступа Eltex MES2428 AC	AS1	ТШ	ESEB013846	90:54:B7:37:0C:80
6	Коммутатор доступа Eltex MES2428 AC	AS2	ТШ	ESEB013644	90:54:B7:38:21:00
7	Коммутатор доступа Eltex MES2428 AC	AS3	ТШ	ESEB012740	90:54:B7:37:EE:80
8	Коммутатор доступа Eltex MES2428 AC	AS4	ТШ	ESEB014292	90:54:B7:37:87:C0
9	Коммутатор доступа Eltex MES2428 AC	AS5	ТШ	ESEB013233	90:54:B7:38:90:80
10	Зонд периферийного узла КМУТ М5	CPE1-M5	ТШ	KM0006	00:0D:B9:54:21:D0
11	Зонд периферийного узла КМУТ М5	CPE2-M5	ТШ	KM1014	00:0D:B9:53:50:74
12	Зонд периферийного узла КМУТ М11	CH-M11	ТШ	K1S00216	1C:57:D8:18:D5:9A
13	Зонд периферийного узла КМУТ М11	ZA-M11	ТШ	K1S00241	1C:57:D8:18:CE:D1
14	Зонд периферийного узла КМУТ М7	CE1	ТШ	VI6F054696	68:13:E2:13:DD:28
15	Зонд периферийного узла КМУТ М7	CE2	ТШ	VI6F054757	68:13:E2:13:DF:10
16	Зонд периферийного узла КМУТ М7	CE3	ТШ	VI6F055749	68:13:E2:13:FE:10
17	Зонд периферийного узла КМУТ М7	CE4	ТШ	VI6F055754	68:13:E2:13:FE:38
18	Зонд периферийного узла КМУТ М7	P1	ТШ	VI6F054670	68:13:E2:13:DC:58
19	Зонд периферийного узла КМУТ М7	P2	ТШ	VI6F054736	68:13:E2:13:DE:68
20	Зонд периферийного узла КМУТ М7	CE5	ТШ	VI6F057523	68:13:E2:14:35:80
21	Зонд периферийного узла КМУТ М7	CE6	ТШ	VI6F054721	68:13:E2:13:DD:F0
22	Зонд периферийного узла КМУТ М7	CE7	ТШ	VI6F059579	68:13:E2:14:75:C0
23	Зонд периферийного узла КМУТ М7	CE8	ТШ	VI6F059555	68:13:E2:14:75:00
24	Зонд периферийного узла КМУТ М7	CE9	ТШ	VI6F057904	68:13:E2:14:41:68
25	Зонд периферийного узла КМУТ М7	CE10	ТШ	VI6F068087	68:13:E2:26:7F:A0
26	Зонд периферийного узла КМУТ М7	CE11	ТШ	VI6F068132	68:13:E2:26:81:08
27	Зонд периферийного узла КМУТ М7	CE12	ТШ	VI6F059540	68:13:E2:14:74:88
28	Зонд периферийного узла КМУТ М7	CE13	ТШ	VI6F063494	68:13:E2:14:F0:18
29	Зонд периферийного узла КМУТ М7	CE14	ТШ	VI6F056764	68:13:E2:14:1D:C8

10. Адресация оборудования Лаборатории

Hostname	Интерфейс	IP адрес
pcastra	br0	10.19.47.2/24 – 10.19.47.255/24
kmut-ml-mtusi	eth0	10.19.47.252/24
	tun0	10.8.0.3/24
Router	ge0/0	10.19.47.50/24
	ge0/1	10.10.212.1/24
ZA-M11	eth0	10.19.47.5/24
	eth1	198.18.1.1/30
	eth2	198.18.2.1/30
CPE1-M5	eth0	198.18.10.2/28
	eth1	10.10.1.1/30
	eth2	198.18.20.2/28
CPE2-M5	eth0	198.18.10.4/28
	eth1	10.10.2.1/30
	eth2	198.18.20.4/28
CPE3	ge0/0	198.18.10.10/28
	ge0/1	10.10.3.1/30
CPE4	ge0/0	198.18.20.10/28
	ge0/1	10.10.4.1/30
CH-M11	br0(eth0,2)	198.18.10.5/28
	br1(eth1,3)	198.18.20.5/28
AS1	- (vlan 1)	10.19.47.21/24
AS2	- (vlan 1)	198.18.10.6/24
AS3	- (vlan 1)	10.19.47.23/24
AS4	- (vlan 1)	10.19.47.24/24
AS5	- (vlan 1)	10.19.47.25/24
CE1	br0	10.10.1.2/30
CE2	br0	10.10.2.2/30
CE3	br0	10.10.3.2/30
CE4	br0	10.10.4.2/30
P1	br0	10.19.47.6/24
P2	br0	10.19.47.7/24
CE5	br0	10.19.47.8/24
CE6	br0	10.19.47.9/24
CE7	br0	10.19.47.10/24
CE8	br0	10.19.47.11/24
CE9	br0	10.19.47.12/24
CE10	br0	10.19.47.13/24
CE11	br0	10.19.47.14/24
CE12	br0	10.19.47.15/24
CE13	br0	10.19.47.16/24
CE14	br0	10.19.47.17/24

11. Данные для авторизации на оборудовании и сервисах

Центральный сервер измерений
Логин: mtusiadmin
Пароль: mtusi_adm
Порт: 8022

Коммутатор доступа Eltex MES2428 AC
Логин: admin
Пароль: admin
Руководство по эксплуатации:
https://eltexcm.ru/assets/files/products/182/mes_series_user_manual_10.3.6.13.pdf
Мониторинг и управление Ethernet-коммутаторами MES по SNMP:
https://eltexcm.ru/assets/files/products/182/mes_configuration_and_monitoring_via_snmp_10.3.6.13.pdf

Зонд периферийного узла КМУТ М5, КМУТ М11
Логин: user
Пароль: KMYTkmyt
Порт: 8022
Для суперпользователя:
Логин: su -
Пароль: kmytKMYT

Зонд периферийного узла КМУТ М7
Логин: user
Пароль: mtusi
Порт: 8022
Для суперпользователя:
Логин: su -
Пароль: mtusi_adm

Маршрутизатор Cisco 1921
Логин: admin
Пароль: Pa$$w0rd
Порт: 22

Маршрутизатор Cisco 2801 (протокол Telnet)
Логин: cisco
Пароль: kmytkmyt

АРМ студента
Логин: astra
Пароль: @stra1inux
grub qweasdzxc

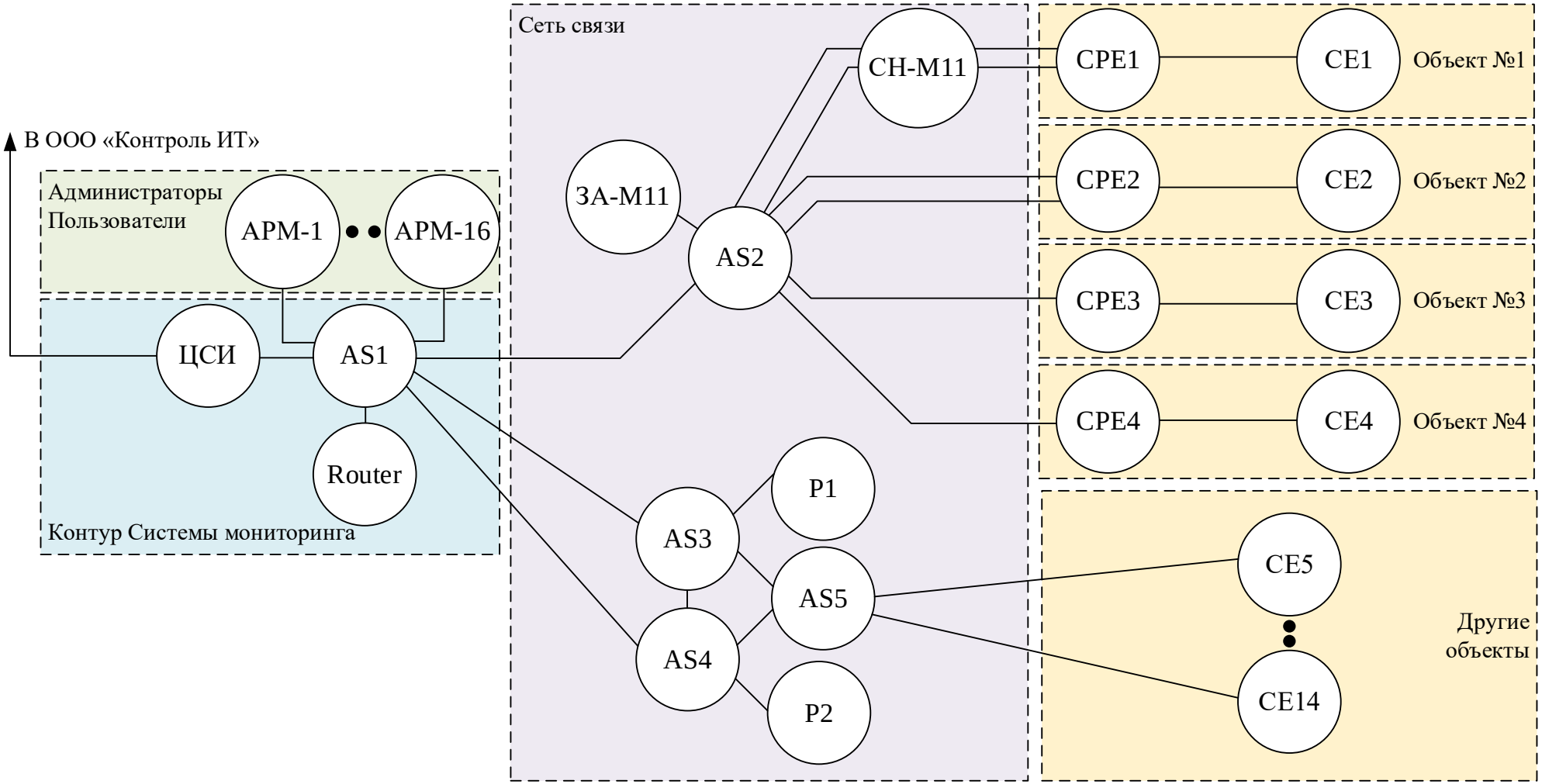
АРМ преподавателя
Логин: administrator
Пароль: mtusi_adm

Система мониторинга (для администратора)
Логин: mtusi_adm
Пароль: 22E!!!ux5LLz

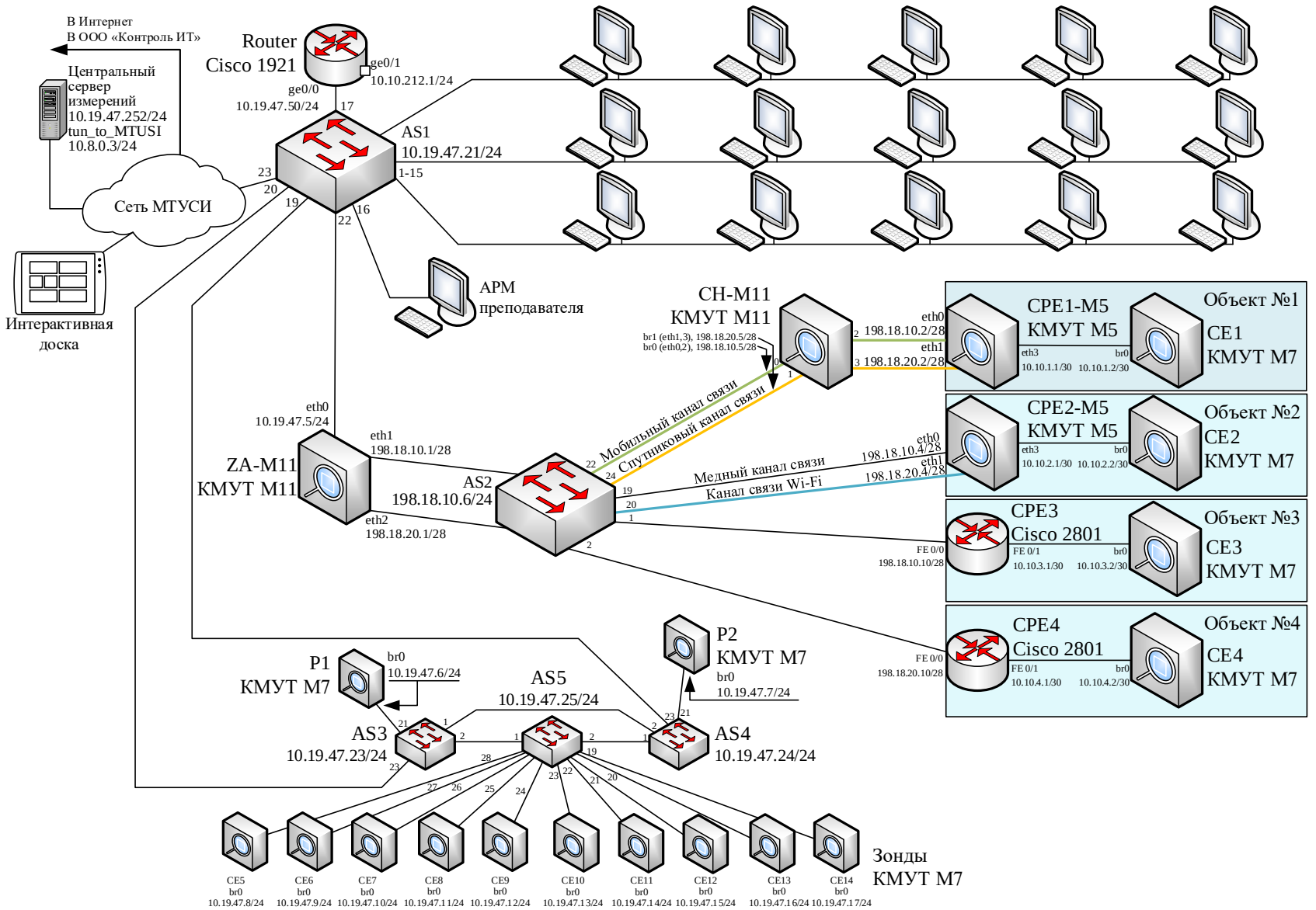
Система мониторинга (для студентов)
Логин: student
Пароль: Mfff<<<522CC

PNetLab (виртуальная лабораторная)
IP tunnel: 10.99.155.133
IP МТУСИ: 10.19.47.251
Логин: admin
Пароль: pnet

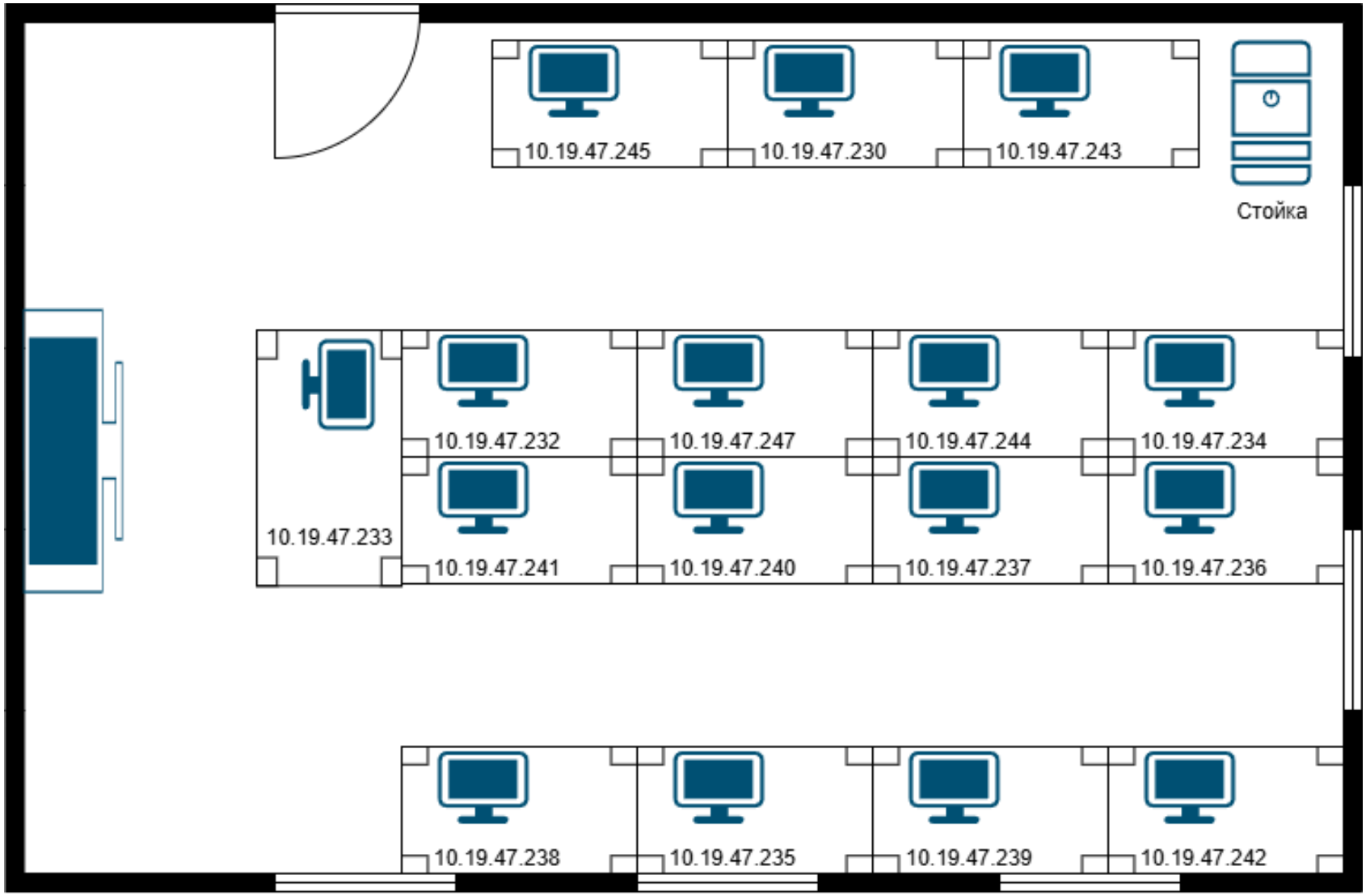
12. Функциональная схема Лаборатории



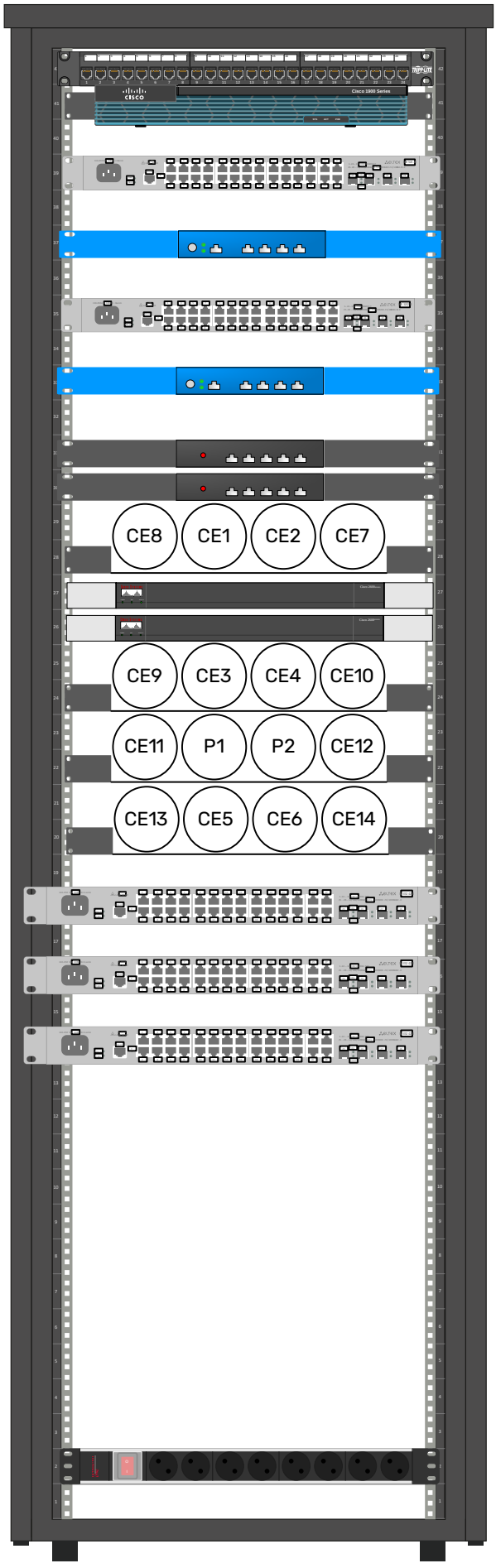
13. Сетевая схема Лаборатории



14. План размещения АРМ с адресацией



15. План размещения оборудования в стойке



Router
AS1
3A
AS2
CH-M11
CPE2-M5
CPE1-M5

CPE3
CPE4

AS3
AS5
AS4

Рекомендуемая литература

1. Олифер Н. А., Олифер В.Г. Компьютерные сети. Принципы, технологии, протоколы: Юбилейное издание, доп. и испр. — СПб.: Питер, 2024. — 1008 с.: ил. — (Серия «Учебник для вузов»).
2. Таненбаум Э., Уэзеролл Д. Компьютерные сети. Издание 5-е. СПб, Питер, 2012 г.
3. Шоттс У. Командная строка Linux. Полное руководство. 2-е межд. изд. — СПб.: Питер, 2020. — 544 с.: ил. ISBN 978-5-4461-1430-6
4. Я есть root. Повышение привилегий в ОС Linux через SUID/SGID. [Электронный ресурс]. URL: https://habr.com/ru/companies/jetinfosystems/articles/506750/
5. Понимание файла /etc/passwd. [Электронный ресурс]. URL: https://www.geeksforgeeks.org/understanding-the-etc-passwd-file/
6. /etc/shadow и создание хэшей паролей yescrypt, MD5, SHA-256 и SHA-512. [Электронный ресурс]. URL: https://www.baeldung.com/linux/shadow-passwords
7. Linux /etc/group File Parsing. [Электронный ресурс]. URL: https://www.iditect.com/guide/linux/linux-etc-group.html
8. Буч Г., Рамбо Д., Якобсон И. Язык UML. Руководство пользователя. 2-е изд.: Пер. с англ. Мухин Н. – М.: ДМК Пресс. – 496 с.: ил. ISBN 5-94074-334-X
9. Моисеев А.Н., Литовченко М.И. Основы языка UML: учеб. пособие. – Томск: Издательство Томского государственного университета, 2023. – 96 с. ISBN 978-5-907722-06-4