1. Утверждаю
2. Директор института СПИНТех
3. НИУ МИЭТ
4. Проф. \_\_\_\_\_\_\_\_\_\_/Гагарина Л.Г./
5. «\_\_\_» \_\_\_\_\_\_\_\_\_ 2021 г.
6. Пояснительная записка

**Направление подготовки – 09.03.04**

1. **Квалификация – бакалавр**

Руководитель выпускной работы:

К.т.н., профессор \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Касимов Р.А./

2. Исполнитель:
3. Студент гр. ПИН-42 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Федотов А.А./

Москва 2021

Содержание

[Перечень используемых сокращений 3](#_Toc70193912)

[Введение 4](#_Toc70193913)

[1. Исследовательский раздел 5](#_Toc70193914)

[1.1 Протоколы синхронизации времени 5](#_Toc70193915)

[1.2 Виды сетевых устройств 8](#_Toc70193916)

[1.3 Обзор аналогичных программных решений 12](#_Toc70193917)

[1.4 Требования к ПМ НКСС. 18](#_Toc70193918)

[1.5 Постановка цели и задач 19](#_Toc70193919)

[Предполагаемый алгоритм решения 19](#_Toc70193920)

[Выводы по разделу 20](#_Toc70193921)

[2. Конструкторский раздел 20](#_Toc70193922)

[2.1 Выбор языка и среды программирования 20](#_Toc70193923)

[2.2 связь программного модуля с другими компонентами 20](#_Toc70193924)

# Перечень используемых сокращений

ПМ – Программный модуль

НКСС – настройка конфигурации сетевых сервисов

ПАК – программно-аппаратный комплекс

# Введение

Информационные технологии все больше и больше проникают в жизнь людей. С их развитием появилась необходимость во взаимодействии компьютеров между собой. Для того, чтобы это взаимодействие обеспечить, создаются и постоянно улучшаются различные сетевые сервисы, которые позволяют решать те или иные задачи. Одной из таких задач является синхронизация времени сетевых устройств. Синхронизация времени необходима для согласования работы устройств и приложений, осуществляющих обработку данных в режиме реального времени. Кроме того, она необходима в системах управления для протоколирования каких-либо событий и своевременного реагирования на них. Для контроля синхронизации используется протокол NTP (Network Time Protocol – протокол сетевого времени), однако, чтобы правила протокола соблюдались, нужно правильно настроить конфигурацию устройства.

Целью данной работы является создание программного модуля, реализующего настройку конфигурации NTP на сетевых устройствах.

Выполнение выпускной квалификационной работы проходило в компании ООО «С-Терра СиЭсПи», специализирующейся на разработке и производстве средств для обеспечения сетевой информационной безопасности, а также ПО для реализации этих средств.

Разрабатываемый модуль ПМ НКСС имеет высокую практическую значимость при решении задач по настройке синхронизации сетевого времени на устройствах, поддерживающих протокол NTP.

Пояснительная записка состоит из введения, исследовательского, конструкторского, технического разделов, заключения и списка литературы.

В исследовательском разделе производится анализ предметной области настройки конфигурации сетевых устройств по протоколу NTP, рассматривается актуальность выбранной темы, производится сравнение существующих решений и выявляются их недостатки.

В конструкторском разделе выбирается язык и среда программирования, определяются необходимые библиотеки, производится разработка алгоритма программного модуля.

# Исследовательский раздел

Во всех сетевых устройствах есть внутренние часы. Они инициализируются при загрузке системы, далее уже время поддерживается с помощью регулярных прерываний от таймера, благодаря чему работают даже при выключенном устройстве. Эти внутренние часы отслеживают как время, так и дату. Важно следить за точностью часов, иначе могут возникнуть различные проблемы с корреляцией журналов, программным обеспечением и т.д.

Рассмотрим некоторые протоколы, которые используются для синхронизации времени на различных устройствах

# Протоколы синхронизации времени

Daytime и Time

Первыми протоколами точного времени, используемым на компьютерах, были DAYTIME (RFC 867) и TIME (RFC 868). Первый предназначался для сообщения даты и времени в понятном человеку виде, второй - понятном компьютеру виде. Формат ответа DAYTIME строго не регламентируется и не предназначен для машинной обработки - предполагается лишь, что человеку, прочитавшему полученную строку, станет ясно текущее время.

Протокол TIME, напротив, предназначен для обмена времени между машинами. На подключившийся к TIME-серверу компьютер приходит UDP-пакет, содержащий единственное 32-битное беззнаковое число, соответствующе числу прошедших с 1 января 1900 г. секунд по UTC. Поскольку такое число переполняется через 136 лет, этот протокол способен функционировать только до 2036 г.

NTP – протокол

Понятно, что ни DAYTIME, ни TIME не могут обеспечить необходимую точность синхронизации времени. В связи с этим, в 1985 г. Дэвидом Л. Миллсом (David L. Mills) из университета Дэлавера был разработан сетевой протокол синхронизации времени NTP, точнее его начальная, позднее названная нулевой (NTPv0) версия, описанная в RFC 958. Протокол NTP использует алгоритм Марзулло (предложен Кейтом Марзулло (Keith Marzullo) из Университета Калифорнии, Сан-Диего), включая такую особенность, как учёт времени передачи. В версии 4 способен достигать точности 10 мс при работе через Интернет, и до 0,2 мс внутри локальных сетей.

Как известно, этот протокол используется для синхронизации времени между устройствами. Для своей работы он использует протокол UDP, при этом учитывает время передачи. Система протокола очень устойчива к изменениям латентности (в данном случае имеется в виду время для прохождения пакета данных от одной точки к другой).

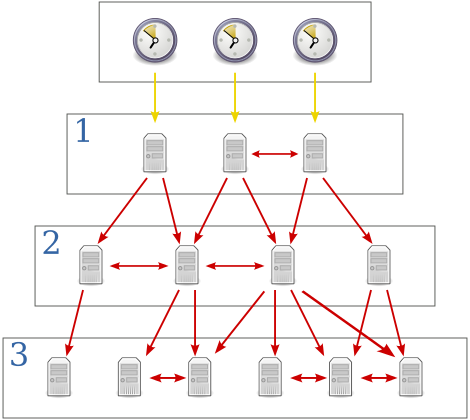


Рисунок 1

NTP-серверы работают в иерархической сети (рисунок 1). Каждый уровень этой сети называют ярусом (stratum). На самом верхнем (нулевом, stratum0) уровне находятся эталонные часы. В качестве таких часов используются сигнал GPS (Global Positioning System) или службы ACTS (Automated Computer Time Service). Сервера первого яруса получают данные от эталонных часов. Далее сервера второго яруса синхронизируются с серверами первого яруса, сервера третьего яруса синхронизируются с серверами второго яруса и так далее по аналогии. Всего может быть до 15 ярусов.

NTP-серверы и NTP-клиенты получают данные о времени от серверов яруса 1, хотя на практике NTP-клиентам лучше не делать этого, поскольку тысячи индивидуальных клиентских запросов окажутся слишком большой нагрузкой для серверов яруса 1. Лучше настроить локальный NTP-сервер, который клиенты будут использовать для получения информации о времени.

Стоит отметить, что протокол не устанавливает время на устройстве в чистом виде. Он корректирует время с использованием временного смещения – разницы между временем на NTP-сервере и на внутренних часах.

Протокол NTP совершенствовался не один раз: NTPv1 (1988 г, RFC 1059), NTPv2 (1989 г., RFC1119), NTPv3 (1992 г., RFC1305), NTPv4 (1996 г., RFC2030).

SNTP протокол

Помимо NTP, существует упрощенная версия этого протокола - SNTP (Simple Network Time Protocol). Он реализован для синхронизации времени конечным клиентом, поскольку все преимущества протокола NTP проявляются именно в сети серверов, а для получения показаний конечным пользователем NTP излишне сложен. Поэтому для синхронизации времени конечными компьютерами и серверами был предложен протокол SNTP (SNTPv3: 1992 г., RFC1361 и 1995 г., RFC1769; SNTPv4 включён как подпротокол в NTPv4).

На самом деле SNTP - это не новый протокол, а способ использования NTP-пакетов и NTP-серверов в приложениях, где не требуется высокоточное время, либо оно недостижимо. В этом случае клиент использует только часть информации UDP-пакета NTP-сервера. SNTP-клиент может работать с любыми версиями NTP-серверов, и кроме них - с особыми SNTP-серверами, которые в откликах заполняют только необходимые данные UDP-пакета.

Таким образом, "облегченный" SNTP образует не сеть синхронизирующихся серверов, а пары "клиент-сервер". Любой NTP-сервер является одновременно SNTP-сервером. Клиент, который не передаёт полученное время дальше, может работать как NTP- или SNTP-клиент, в зависимости от условий. Для SNTP, как и для NTP, зарезервирован 123-й UDP-порт.

В большинстве сетевых устройств используется протокол NTP. Разберем некоторые примеры сетевых устройств, которые при своей работе используют данный протокол.

* 1. Виды сетевых устройств

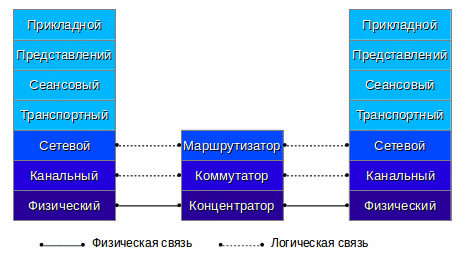


Рисунок 2

Прежде чем разбирать виды сетевых устройств, нужно рассказать про сетевую модель OSI. Данная модель определяет многоуровневое взаимодействие между узлами сети, где каждый уровень представляет набор специфичных функций. Нижние уровни: физический, канальный и сетевой — регламентируют процесс передачи данных как таковой. Сетевые интерфейсы оконечных узлов представляют эти функции, но этого недостаточно, чтобы обеспечить связь между произвольными узлами в локальной сети и, тем более, в Интернет. Причина проста — невозможно установить непосредственные физические связи между всеми узлами. Для снятия этого и прочих ограничений используется дополнительное сетевое оборудование. Основными его типами являются:

* Концентраторы
* Коммутаторы
* Маршрутизаторы

Концентратор

Концентратор (также хаб от англ. hub — центр) – сетевое устройство, работающее на первом уровне. Основные его задачи связаны с ретрансляцией поступающих данных на все остальные подключенные порты. Никаких действий по обработке данных не производится. Благодаря этому, сеть Ethernet построенная на концентраторах имеет все недостатки общей шины кроме одного: вывод из строя некоторого узла не приводит к остановке работы всей сети. На сегодняшний день концентраторы почти не используются, уступив место более совершенным устройствам – коммутаторам. Единственное преимущество концентратора — низкая стоимость. Оно было актуально лишь в первые годы развития сетей Ethernet. По мере совершенствования и удешевления электронных микропроцессорных компонентов данное преимущество концентратора полностью сошло на нет, так как стоимость вычислительной части коммутаторов и маршрутизаторов составляет лишь малую долю на фоне стоимости разъёмов, разделительных трансформаторов, корпуса и блока питания, общих для концентратора и коммутатора.

Недостатки концентратора являются логическим продолжением недостатков топологии общая шина, а именно — снижение пропускной способности сети по мере увеличения числа узлов. Кроме того, поскольку на физическом уровне узлы не изолированы друг от друга, все они будут работать со скоростью передачи данных самого худшего узла. Например, если в сети присутствуют узлы со скоростью 100 Мбит/с и всего один узел со скоростью 10 Мбит/с, то все узлы будут работать на скорости 10 Мбит/с, даже если узел 10 Мбит/с вообще не проявляет никакой информационной активности. Ещё одним недостатком является вещание сетевого трафика во все порты, что снижает уровень сетевой безопасности и даёт возможность подключения снифферов.



Рисунок 3 сетевой концентратор

Сетевой коммутатор (свитч) – устройство, которое предназначено для соединения нескольких узлов компьютерной сети в пределах одного или нескольких сегментов сети (рисунок 4). Коммутатор хранит в памяти таблицу коммутации, в которой указывается соответствие МАС-адреса узла порту коммутатора. При включении эта таблица пуста, и он работает в режиме обучения. В таком режиме поступающие на какой-либо порт данные передаются на все остальные порты, а МАС-адрес порта-отправителя заносится в таблицу. Если же МАС-адрес хоста-получателя имеется в таблице, данные передаются только получателю. Таким образом, со временем проходящий трафик локализируется.

Существует три способа коммутации. Каждый из них — это комбинация таких параметров, как время ожидания и надёжность передачи.

* С промежуточным хранением (Store and Forward). Коммутатор читает всю информацию в кадре, проверяет его на отсутствие ошибок, выбирает порт коммутации и после этого посылает в него кадр.
* Сквозной (cut-through). Коммутатор считывает в кадре только адрес назначения и после выполняет коммутацию. Этот режим уменьшает задержки при передаче, но в нём нет метода обнаружения ошибок.
* Бесфрагментный (fragment-free) или гибридный . Этот режим является модификацией сквозного режима, который частично решает проблему коллизий. В теории поврежденные кадры (обычно из-за столкновений) часто короче минимального допустимого размера кадра Ethernet, равного 64 байтам. Поэтому в этом режиме коммутатор отбрасывает кадры длиной меньше 64 байт, а все остальные после прочтения первых 64 байт в сквозном режиме передаёт дальше



Рисунок 4 сетевой коммутатор

Маршрутизатор (роутер) – устройство, работающее на третьем уровне и выполняющее функции перенаправления трафика между сетями. Маршрутизатор может связывать разнородные сети различных архитектур. Для принятия решения о пересылке пакетов используется информация о топологии сети и определенные правила, заданные администратором. Обычно маршрутизатор использует адрес получателя и по таблице маршрутизации определяет путь, по которому следует передать данные. Если в таблице такого маршрута нет – пакет отбрасывается.

Существуют и другие способы определения маршрута пересылки пакетов, когда, например, используется адрес отправителя, используемые протоколы верхних уровней и другая информация, содержащаяся в заголовках пакетов сетевого уровня. Нередко маршрутизаторы могут осуществлять трансляцию адресов отправителя и получателя, фильтрацию транзитного потока данных на основе определённых правил с целью ограничения доступа, шифрование/расшифровывание передаваемых данных и т. д.



Рисунок 5

* 1. Обзор аналогичных программных решений

В ходе предварительных исследований был проведен анализ существующих решений с функционалом, требуемым от разрабатываемого программного модуля. При анализе учитывался не только необходимый функционал: настройка параметров для синхронизации времени на устройствах по протоколу NTP, но и смежные возможности, облегчающие использование решения; возможности компании разработчика по актуализации функционала, и его расширению. Характеристики рассмотренных решений в сравнении с ПМ НКСС представлены в табл. 1.

Таблица 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Аналоги  Критерии | MikroTik RouterOS | Quagga | Junos OS | Cisco IOS | C-Терра Шлюз |
| Возможность настройки NTP | Есть | Нет | Есть | Есть | Есть |
| Способ настройки NTP конфигурации | С помощью конфигурационных файлов и команд | Нет | С помощью команд | С помощью команд | С помощью команд |
| Возможность сохранения логов NTP в отдельный файл | Нет | Нет | Нет | Нет | Да |
| Необходимость загрузки дополнительных файлов | Есть | Нет | Нет | Нет | Нет |
| Наличие пользовательского интерфейса | Есть | Нет | Есть | Есть | Нет |

**MikroTik RouterOS**

MikroTik— латвийский производитель сетевого оборудования. Компания разрабатывает и продает проводное и беспроводное сетевое оборудование, в частности маршрутизаторы, сетевые коммутаторы (свитчи), точки доступа, а также программное обеспечение — операционные системы и вспомогательное ПО. Компания была основана в 1996 году с целью продажи оборудования на развивающихся рынках.

Одним из продуктов MikroTik является RouterOS — сетевая операционная система на базе Linux. RouterOS предназначена для установки на маршрутизаторы MikroTik RouterBoard. Также данная система может быть установлена на ПК, превращая его в маршрутизатор с функциями брандмауэра, VPN-сервера/клиента, QoS, точки доступа и другими. Система также может служить в качестве Captive-портала на основе системы беспроводного доступа. Также существует специальная версия RouterOS, именуемая Cloud Hosted Router и предназначенная для облачных виртуальных машин.

Операционная система имеет несколько уровней лицензий с возрастающим числом функций. Кроме того, существует программное обеспечение под названием Winbox, которое предоставляет графический интерфейс для настройки RouterOS. Доступ к устройствам под управлением RouterOS возможен также через Веб интерфейс, FTP, Telnet, и SSH. Существует также API, позволяющий создавать специализированные приложения для управления и мониторинга.

RouterOS поддерживает множество сервисов и протоколов, которые могут быть использованы средними или крупными провайдерами — таких, как OSPF, BGP, VPLS/MPLS. RouterOS — достаточно гибкая система, и очень хорошо поддерживается Mikrotik, как в рамках форума и предоставления различных Wiki-материалов, так и специализированных примеров конфигураций.

RouterOS обеспечивает поддержку практически всех сетевых интерфейсов на ядре Linux. Из беспроводных чипсетов поддерживаются решения на основе Atheros и Prism (по состоянию RouterOS версии 3.x). Mikrotik также работает над модернизацией программного обеспечения, которая обеспечит полную совместимость устройств и ПО Mikrotik с набирающими популярность сетевыми технологиями, такими как IPv6.

RouterOS предоставляет системному администратору графический интерфейс (WinBox) для наглядной и быстрой настройки файервола, маршрутизации и управления QoS. В том числе, в интерфейсе WinBox практически полностью реализована функциональность Linux-утилит iptables, iproute2, управление трафиком и QoS на основе алгоритма HTB.

**Quagga**

Quagga — пакет свободного программного обеспечения, поддерживающий протоколы динамической маршрутизации IP. Компьютер с установленным и сконфигурированным пакетом Quagga становится способен использовать любые из нижеследующих протоколов динамической маршрутизации:

* Routing Information Protocol (RIP): v1, v2, v3;
* Open Shortest Path First (OSPF): v2, v3;
* Border Gateway Protocol (BGP): v4;
* Intermediate System to Intermediate System (IS-IS);
* Protocol Independent Multicast (PIM, только PIM-SSM).

Пакет Quagga может быть установлен на UNIX-подобные операционные системы. Quagga — это усовершенствованная версия GNU Zebra, компьютерной программы, развитие которой остановилось в 2005 году.

Свое название продукт получил от животного. Квагга (Quaga) — это подвид зебры, обитавший в южной Африке. В отличие от самого животного, истреблённого в конце XIX‐ого века, проект Quagga пережил ныне умерший проект GNU Zebra. Последняя стабильная версия Zebra (0.95a) датирована 2005-09-08, большинство BGP маршрутизаторов, которые использовали GNU Zebra, перешли на Quagga.

Quagga состоит из базового ядра (core daemon) zebra, выполняющего роль промежуточного уровня абстракции (abstraction layer) ядра ОС, и предоставляющего Zserv API клиентам по протоколу TCP. Клиентами Zserv выступают службы (демоны):

* ospfd (протокол OSPFv2);
* ripd (протокол RIP v1, V2);
* ospf6d (протокол OSPFv3 IPv6);
* ripngd (протокол RIPng IPv6);
* bgpd (протокол BGPv4+, включая поддержку multicast и IPv6));
* isisd (протокол IS-IS);
* pimd (протокол PIM, пока только PIM-SSM).

Библиотека Quagga существенно облегчает разработку дополнительных модулей, позволяя всем её службам использовать унифицированный способ конфигурации и управления.

**JUNOS**

JUNOS — операционная система, используемая в оборудовании компании Juniper Networks. Создана на основе 4-й ветки свободной ОС FreeBSD. Главный конкурент — операционная система IOS-XR компании Cisco Systems.

В JUNOS существует возможность установки дополнительного ПО, которое распространяется в виде пакетов, подписанных соответствующим сертификатом Juniper Networks. Система управления пакетами JUNOS также унаследована от FreeBSD.

Пользовательское окружение представляет собой полноценную рабочую среду с набором классических (для FreeBSD) утилит. Однако внесение изменений в конфигурацию допускается только при помощи специальной утилиты — «cli». Этой же утилитой осуществляется управление всеми ASIC. Серии оборудования Juniper Networks, работающие под управлением JUNOS:

* маршрутизаторы — M, MX, T
* сервисные шлюзы — SRX
* сервисные маршрутизаторы — J
* коммутаторы — EX
* коммутаторы для ЦОД – QFX

Командный интерфейс JUNOS позволяет как выполнять команды, так и вводить конфигурацию. Изменения в конфигурации не применяются до выполнения команды commit (предусмотрена возможность автоматического отката конфигурации на предыдущую, если она не была подтверждена в течение заданного времени). Поддерживается история версий (до 50 версий), к которым может производиться откат. Как и большинство других интерфейсов командной строки маршрутизаторов, поддерживает автодополнение по уникальной комбинации первых символов команды (например, sh int fe-1/1/1 ex будет расшифрованно как show interface fe-1/1/1 extensive). Автодополнение происходит не при интерпретации команды, а при нажатии пробела. Поддерживается контекстная справка (вызов — знак вопроса).

Конфигурация представляет собой директивы конфигурирования той или иной подсистемы. Директивы могут содержать вложенные элементы, описывающие настройку отдельных компонент. Например, конфигурация интерфейса ethernet может иметь вложенные настройки для отдельных подсетей, которые, в свою очередь, могут иметь вложенные настройки для различных протоколов (например, ip4 и ip6).

Оконечные узлы конфигурации (не содержащие вложенных элементов) заканчиваются точкой с запятой, содержащие вложенные элементы задают их с помощью фигурных скобок (точка с запятой в этом случае не ставится). Конфигурирование может осуществляться либо вводом готового текстового блока, либо использованием сокращённых директив с помощью команды set.

**Cisco IOS**

Cisco IOS (от англ. Internetwork Operating System — Межсетевая Операционная Система) — программное обеспечение, используемое в маршрутизаторах и сетевых коммутаторах Cisco. Cisco IOS является многозадачной операционной системой, выполняющей функции сетевой организации, маршрутизации, коммутации и передачи данных.

В Cisco IOS есть специфичный интерфейс командной строки (command line interface, CLI), который был скопирован многими другими сетевыми продуктами. Интерфейс IOS предлагает набор многословных команд, согласно выбранному режиму и уровню привилегий пользователя. Global configuration mode предоставляет возможность для изменения настроек системы и сетевых интерфейсов.

Всем командам приписывается определённый уровень привилегий от 0 до 15, и к ним могут обратиться только пользователи с соответствующим уровнем привилегий. Через командный интерфейс можно определить доступные команды для каждого уровня привилегий.

**С-Терра Шлюз**

ПАК «С-Терра Шлюз» является программно-аппаратным средством защиты сетей, подсетей, офисов и самого шлюза от несанкционированного доступа.

ПАК «С-Терра Шлюз» используется для защиты от несанкционированного доступа к информации ограниченного доступа, не содержащей сведений, составляющих государственную тайну. ПАК «С-Терра Шлюз» может применяться:

* в значимых объектах критической информационной инфраструктуры (КИИ) до 1 категории включительно;
* в государственных информационных системах (ГИС) до 1 класса защищенности включительно;
* в автоматизированных системах управления производственными и технологическими процессами (АСУТП) до 1 класса защищенности включительно;
* в информационных системах персональных данных (ИСПДн), обеспечивающих 1, 2, 3 и 4 уровни защищенности персональных данных;
* в информационных системах общего пользования (ИСОП) II класса.

ПАК «С-Терра Шлюз» предназначен для работы на аппаратных платформах в архитектуре Intel (x86-64 совместимых) универсального назначения, отвечающих следующим минимальным требованиям: имеющих от 1 процессора, от 2 Гб ОЗУ, от 3 сетевых интерфейсов, подключаемых к внешним сетям:

* Аквариус T30S100DC, T30S001DC, T40S102DF-B;
* LN-S, LN-Si, LN-M, LN-L, LN-XL;
* АТБ-АТОМ-1.

ПАК «С-Терра Шлюз» работает под управлением операционной системы Debian Linux 9 с установленными последними обновлениями безопасности.

* 1. Требования к ПМ НКСС.

ПМ НКСС должен обеспечивать выполнение следующих функций:

* Правильное распознавание введенных команд
* При вводе неверной команды должно выводиться сообщение об ошибке с указанием места ошибки
* Должна быть возможность настройки конфигурации в качестве NTP-клиента
* Должна быть возможность настройки конфигурации в качестве NTP-сервера
* Должна быть возможность указания NTP-серверов, с которыми проводится синхронизация
* Должна быть возможность с просмотра статуса текущих настроек с помощью определенных команд
* Должна быть предусмотрена возможность сохранения логов в отдельный лог-файл
* Все возвращаемые данные должны выводиться обратно на командную строку
  1. Постановка цели и задач

Цель разработки:

Создать программный модуль, позволяющий упростить настройку конфигурации NTP сервисов на сетевых устройствах.

**Задачи:**

* Исследование предметной области
* Сравнительный анализ существующих решений
* Выбор платформы для реализации модуля
* Выбор языка и среды разработки
* Разработка схемы данных ПМ НКСС
* Разработка схемы алгоритма ПМ НКСС
* Программная реализация ПМ НКСС
* Тестирование и отладка модуля
* Разработка руководства оператора

Предполагаемый алгоритм решения

ПМ получает команду и необходимые данные из консоли. На основе введенной команды выполняются действия по изменению настроек конфигурации или формировании сообщения о текущих настройках NTP.

Если в настройках указан файл для логирования, то в этот файл сохраняются логи работы NTP-сервиса.

# Выводы по разделу

В исследовательском разделе была обоснована актуальность разработки ПМ НКСС. Исследована предметная область и проведен сравнительный анализ решений для настройки конфигурации NTP на сетевых устройствах.

# Конструкторский раздел

* 1. Выбор языка и среды программирования
  2. связь программного модуля с другими компонентами