**Отчёт обучающегося по практике**

Студент: Федотов Алексей Александрович Группа: ПИН-41

Направление: 09.03.04 Программная инженерия

Образовательная программа: Программные технологии распределенной обработки информации

### Вид и тип практики: Учебная практика - практика по получению первичных профессиональных умений и навыков, в т.ч. первичных умений и навыков научно-исследовательской деятельности)

Весенний семестр 2021 учебного года

Место прохождения практики: ООО «С-Терра СиЭсПи»

Учебная практика проходила на предприятии ООО «С-Терра СиЭсПи» в отделе агента, мобильных решений и систем управления. Этот отдел занимается разработкой, тестированием и сопровождением программных продуктов кампании.

Продукция компании «С-Терра СиЭсПи» используется как в государственных учреждениях, так и в коммерческих организациях в соответствии с отраслевыми стандартами и требованиями по защите информации, в том числе:

* защите конфиденциальной информации органов государственной власти в соответствии с СТР‑К, а также совместным приказом ФСБ и ФСТЭК от 31 августа 2010 года №416/489
* защите подключения информационных систем государственных органов к Интернет в соответствии с Указом Президента РФ от 17 марта 2008 года №351, Постановлением Правительства РФ от 18 мая 2009 года №424, Приказом ФСО от 7 августа 2009 года №487
* защите персональных данных в соответствии с методическими рекомендациями ФСБ по защите персональных данных, а также в соответствии с отраслевыми стандартами Банка России, НАУФОР, НАПФ, операторов связи и др.
* защите объектов инфраструктуры в соответствии нормативными документами ФСТЭК России по защите ключевых систем информационной инфраструктуры
* защите систем управления технологическими процессами (АСУ ТП)
* защите крупных территориально-распределенных сетей и т.п.

В ООО «С-Терра СиЭсПи» была поставлена задача по разработке программного модуля, обеспечивающего упрощение конфигурирования сетевого сервиса, использующегося в компании.

# Исследовательский раздел

Все сетевые устройства имеют внутренние часы. Они инициализируются при загрузке системы, затем время уже поддерживается с помощью регулярных прерываний от таймера, так что они 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 логов, ПО и т. Д.

Рассмотрим некоторые протоколы, которые используются для синхронизации времени на разных устройствах.

# Протоколы синхронизации времени

Daytime и Time

Самые первые попытки синхронизировать время начали предприниматься в 1983 году. Именно тогда появился документ RFC 867, в котором описывался протокол DAYTIME. Немногим позже, в 1985 году появился документ RFC 868, описывающий протокол TIME. Первый из них предназначался для людей, второй же для компьютера.

По протоколу DAYTIME нет строгого стандарта вывода данных, считается что человек, увидевший выведенную строку сразу поймет какое число и который час. Поэтому он не предназначен для обработки компьютером.

В то же время, протокол TIME нужен для обмена времени между компьютерами. Разберем принцип его работы. К серверу подключается какой-либо компьютер. Этот сервер отсылает ему пакет по протоколу UDP. В данном пакете содержится одно 32-битное число. Это число показывает количество секунд, прошедших с первого января 1900 года. Это число переполнится через 136 лет, а значит, в 2036 году этот протокол перестанет работать.

NTP – протокол

Из всего вышеперечисленного легко понять, что протоколы DAYTIME и TIME не способны обеспечивать необходимую для нас точность синхронизации, т. к. не учитывают время задержки, за время которой данные доходят от сервера клиенту. Чтобы решить эту проблему, в 1985 году американский компьютерный инженер Дэвид Л. Миллс из университета Делавэра, разработал новый протокол. Этот протокол назвали NTP(Network Time Protocol), или в переводе на русский язык – протокол синхронизации времени. Этот протокол был описан в документе RFC 958.

Новый протокол использует так называемый алгоритм Марзулло, названный в честь Кейта Марзулло из университета Калифорнии в Сан-Диего. В четвертой версии протокола (NTPv4) есть возможность достигать точности вплоть до 10 мс, включая такой фактор, как учет времени передачи данных.

Как уже сказано выше, данный протокол применяется для синхронизации времени между устройствами. Для своей работы он применяет еще один протокол, предназначенный для передачи данных – UDP.

Все сервера, работающие по протоколу NTP находятся в некой иерархической сети (см. рис. 1). Каждый новый уровень такой сети называется stratum, или же ярусом. На самом верхнем уровне (stratum0) находятся высокоточные часы, их еще называют эталонными. В качестве таких часов используются сигнал GPS или службы ACTS. Серверы первого яруса могут получать данные от этих часов. Сервера второго, в свою очередь. Могут получать данные от серверов первого яруса. Сервера третьего от второго и далее по аналогии. Всего в такой иерархии может быть до 15 ярусов.

Как NTP-серверы, так и NTP-клиенты получают данные от серверов первого яруса, хотя по факту лучше клиентам так не делать. Все дело в том, что огромное количество клиентских запросов могут оказаться слишком большой нагрузкой для серверов из stratum1. Будет гораздо удобнее, если настроить локальный сервер, к которому и будут подключаться клиенты.

Важно отметить, что этот протокол не настраивает текущее время на устройстве. Принцип действия заключается в том, что протокол корректирует время устройства с использованием разницы между временем на сервере и на внутренних часах.

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

SNTP протокол

Существует также и упрощенная версия протокола NTP. Она называется SNTP (Simple Network Time Protocol) и описана в документе RFC 4330. Этот протокол был создан для того, чтобы синхронизировать время в тех системах, где не трубется высокая точность.

Все основные плюсы NTP-протокола проявляются только в описанной выше иерархической сети. Для того, чтобы отправлять показания конечным пользователям, этот NTP оказался слишком сложным. Чтобы решить такую проблему был создан SNTP.

Отметим, SNTP не является каким-то новым протоколом. Он скорее является способом использования пакетов и серверов протокола NTP в приложениях, где не требуется высокая точность. Клиент будет использовать только какую-то часть информационного пакета, который был полученн от сервера.

Стоит сразу сказать, что SNTP – это не новый протокол. Это скорее способ использования NTP-пакетов и NTP-серверов в таких приложениях, где не особо требуется очень высокая точность, либо если такая точность просто недостижима. В таком случае клиент использует только часть информации пакета, полученного от сервера. SNTP может работать с любыми NTP-серверами, а еще с особенными SNTP-серверами. Их особенностью является то, что в своих откликах они будут заполнять только самые необходимые данные пакета.

Упрощенный протокол не образует сеть серверов. Вместо этого он образует пары клиент-сервер. Любой NTP-сервер одновременно является еще и SNTP-сервером. Клиент, который не передает полученное с сервера время далее, может работать и как NTP-клиент, и как SNTP-клиент. Все зависит от условий. Именно для протоколов SNTP и NTP зарезервирован 123 UDP порт.

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

# 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

Сравнение аналогов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 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 | Графический интерфейс | Консоль | Графический интерфейс | Графический интерфейс | Консоль |

# 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 практике

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

Руководитель практики от МИЭТ / Касимов Р. А./

Руководитель практики   
от организации / Харитонов Р. Л./

Обучающийся / Федотов А.А./