МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование)

02.03.03 Математическое обеспечение и администрирование информационных систем
(код и наименование направления подготовки, специальности)

WEB-дизайн и мультимедиа
(направленность (профиль)/специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему Разработка ПО для выбора оптимального маршрута интернет соединения

Студент Н.В. Хорищенко
(И.О. Фамилия) (личная подпись)

Руководитель С.В. Митин
(ученая степень, звание, И.О. Фамилия)

Консультант М.М. Бажутина
(ученая степень, звание, И.О. Фамилия)

Тольятти 2022

Аннотация

Бакалаврская работа на тему: «Разработка ПО для выбора оптимального маршрута интернет соединения».

Введение определяет основные рассматриваемые направления в рамках работы, ее цели и задачи, преследуемые в рамках рассмотрения вариантов решения.

Структура работы представлена введением, 3 главами, заключением, списком литературы и приложением, содержащее основные фрагменты программного кода.

В первой главе представляется произведенный анализ объекта исследования и разработки, описана модель процесса автоматизации ручного управления в рамках разработки системы для поддержания принятия решения при сетевом администрировании.

Во второй главе описываются итерации подготовительного этапа, направленные на подготовку системы, позволяющую выявить имеющиеся проблемы по линии системы передачи данных, а также произвести пост-анализ произведенной разработки.

Третья глава раскрывает этапы основной реализации, направленные на разработку и дальнейшее внедрение получившегося решения, представлены этапы первоначального тестирования и выявленные направления модернизации алгоритма для улучшения показателей работы.

Стоит отметить, что представляемая работа не имеет аналогов, описанных в открытых источниках и позволяет решить проблему сетевого администрирования вручную, автоматизирует процесс принятия решения о выборе оптимального маршрута соединения.

Работа представлена на 68 страницах с приложением и включает 27 рисунка, 11 таблиц.

Abstract

The title of the bachelor's thesis is Software development for choosing the optimal route of Internet connection.

The aim of the work is develop automatic software for external analysis and management of Internet connections operating in a distributed network to select the optimal route for traffic.

The subject of the research is the the process of automatic control of the available communication channels.

In the first part we outline an analysis of the object of research and development is presented, a model of the manual control automation process is described as part of the development of a system to support decision-making in network administration.

The special focus is worth noting that the presented work has no analogues described in open sources and allows you to solve the problem of network administration manually, automates the decision-making process on choosing the optimal connection route.

The work consists of a 68-page explanatory note including 27 figures, 11 tables, a list of 21 references including 5 foreign sources, 1 appendix and the graphic part on 9 A4 sheets.

Содержание

Введение

В современных реалиях 21 века крупные IT-компании ежедневно сталкиваются с множеством задач по направлениям B2B, B2C, в основе реализации которых лежит постоянный обмен данными между клиент-серверными приложениями, работоспособность которых напрямую зависит от качества применяемых каналов связи.

Современные дорогостоящие технические решения позволяют реализовать различной сложности проекты, а также сети передачи данных, отвечающие стандартам качества и должным уровнем защиты информации от утечки данных. Но, к сожалению, данные решения не всегда могут быть применены в сегменте малого и среднего бизнеса, ввиду их повышенной стоимости по сравнению с более доступным классом устройств, обладающим набором протоколов, но требующих внешних систем для принятия решений в различных складывающихся ситуациях.

В своей выпускной квалификационной работе хотел бы рассмотреть направление по разработке алгоритмов для системы поддержки принятия решения по направлению систем выбора оптимального маршрута интернет соединения [5]. В крупных компаниях, имеющих большой спектр географически-распределенных узлов и офисов, клиентов и заказчиков, встает проблема в содержании имеющейся инфраструктуры в режиме доступности 99.9 %, исключая возможность влияния аварийных ситуаций на общую доступность ресурсов.

В рамках преддипломной практики мне удалось погрузиться более детально в принцип построения крупной корпоративной сети, имеющей множество отходящих соединений, включая в том числе физические соединения с заказчиками и субподряными организациями. В ходе полного погружения в бизнес-структуру удалось выявить актуальные проблемы, пожелания по усовершенствованию алгоритмов управления сетевой

инфраструктурой, стека сервисов и применяемых технологий, которые в свою очередь легки в написание данной работы.

Целью, решаемой в рамках написания работы является разработка автоматического программного обеспечения для внешнего анализа и управления интернет соединениями, функционирующими в распределенной сети для выбора оптимального маршрута прохождения трафика.

При наличии более 2000 активных подключений к ядру сети оперативное ручное управление трафиком не представляется возможным ввиду требуемых длительных временных затрат на анализ ситуации, принятие решения по необходимым переключениям. Применение данного ручного подхода конечно возможен, но в данном случае потребуется расширенный штат квалифицированных специалистов, занимающихся постоянным анализом происходящего.

Описанный подход не может быть оптимальным для активно развивающейся компании, имеющей ряд стартап проектов, требующих больших человеческих ресурсов, которые не могут постоянно отвлекаться на запущенные ранее системы, которые также требуют ручного взаимодействия, в связи с чем возрастает актуальность внедрения данной системы.

Для реализации автоматического алгоритма и ухода от ручного управления в случае возникновения критических ситуаций необходимо выполнение следующих задач:

- Изучить и описать применяемую топологию сети, принципов построения и связей с внешними организациями;
- Проанализировать принцип управления, применяемые протоколы маршрутизации;
- Выявить слабую сторону ручного управления, на основании которой разработать алгоритм автоматизации;
- Организовать экспериментальную работу по внедрению автоматизированной системы управления.

Объектом исследования является процесс автоматического управления имеющимися каналами связи.

Предметом исследования, в свою очередь, является процесс автоматического анализа с последующим выбором маршрута и построения схемы переключения на каналах связи.

Для подготовки анализа и дальнейшего обоснования выбранного метода исследования необходимо подготовить техническую базу мониторинга, основываясь на данные которого можно производить анализ разрабатываемых программных решений и аппаратных реализаций.

В рамках рассматриваемых технологий оперативного мониторинга было принято решение об использовании open-sourse решения Zabbix с распределенным контролем по SNMP протоколу, внешним агентам и Zabbix-Proxy.

Запущенная система мониторинга позволила собрать необходимую для дальнейшего анализа информацию и предоставить графики доступности ресурсов до и после внедрения функционала.

Практическая значимость выполняемой работы заключается в существенном сокращении временных ресурсов необходимых для оперативных переключений и глубокой диагностики в случаях проявления нестабильной работы каналов связи или не отвечающих описанным показателям в условиях договора требований к содержанию каналов связи, а также сокращению необходимого количество оперативного персонала, необходимого для контроля и управления системой. Внедренный алгоритм позволил поднять уровень предоставляемых услуг и снизить время простоя из-за человеческого фактора.

Глава 1 Предпроектное обследование инфраструктуры

1.1. Технико-экономическая характеристика управления информационными системами

1.1.1. Характеристика предприятия

Рынок информационных технологий из года в год расширяет свои границы распространения, применяемые решения стремительнее входят в ежедневный ритм каждого из нас, становясь чем-то обыденным.

История Акционерного общества берет свое начало с 2006 года как инновационная организация, занимающаяся разработкой инновационных решений. Одним из первых решений компании был сервис системы защиты оригинальности, далее с каждым годом компания внедряет дополнительные решения, расширяя свой уровень выхода на Российский рынок.

В настоящее время контакт-центр имеет лицензию ФСТЭК для работы с персональной и конфиденциальной информацией, а бизнес-процессы сертифицированы по ISO9001.

География площадок насчитывает 17 различных городов и включает в себя более 2500 автоматизированных рабочих мест. Ежедневно контактный центр обслуживает более 140 различных проектов заказчиков.

Коллектив компании на постоянной основе изучает рынок программных продуктов и решений, стремясь выявить недостающие решения для бизнеса, стремясь разработать продукты для данной актуализировавшийся сферы.

Система телекоммуникационного центра – программно-аппаратный комплекс узлов, связанных между собой, выполняющих функцию территориально распределенного Центра обработки вызовов (ЦОВ).

Представляет собой клиентно-серверную архитектуру. На базе системы – осуществляется взаимодействие между операторами и клиентам.

Клиентская часть системы представляет собой:

Административную панель управления – предназначена для настройки проектов, сценариев, формирования анкет операторов. Вместе с тем позволяет осуществлять контроль деятельности операторов: осуществлять прослушку звонков, формировать исторические отчеты по работе специалистов, так и осуществлять контроль в режиме реального времени.

Операторское рабочее автоматизированное место (АРМ) – служит инструментом

взаимодействия с клиентом, выполняет функции программного телефона, анкеты (базы знаний).

- анкета встраивается внутри HTML страницы;
- используются Smart поля - изменяется логика на стороне Бекенда;
- анкета может быть отправлена на E-mail;
- может быть настроена для получения внешних обращений от клиентов через API;

Схема взаимодействия программных продуктов и сервисов представлена на рисунке №1.

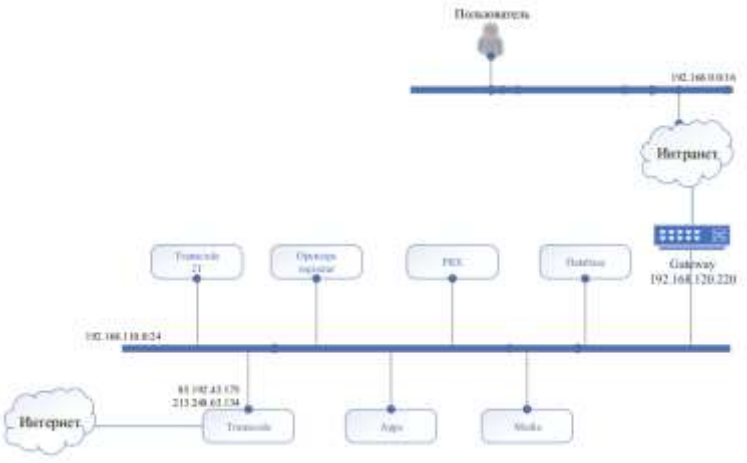

Рисунок 1 - Схема взаимодействия ресурсов

Управление адресным пространством осуществляется путем разделение и выноса сервисов, выполняющих различные функции в отдельные подсети, распределение которых указано в таблице № 1.

Таблица 1 - Краткое распределение адресного пространства для сервисов

Внешний IP адрес	Внутренний IP адрес	Назначение	Примечание
-	192.168.40.246	Производственный сервер приложений	API
-	192.168.40.253 192.168.40.252	Файловый сервер	-
-	192.168.40.116 192.168.40.117 192.168.40.118	Сервер журналирования и отчетов	-
X.X.X.247	192.168.40.108 192.168.40.107	Сервер коммутации медиа-трафика	-
-	192.168.40.116 192.168.40.117 192.168.40.118	Кластер СУБД	-

Точкой входа при взаимодействии пользователя с системой является граничный сервер Nginx (192.168.40.228), который осуществляет проксирование всех запросов на промышленный сервер WEB приложений (192.168.40.246). Данный сервер выдаёт конфигурацию пользовательскому приложению (софтфону) для подключения к media серверам [17] по SIP протоколу. Также данный сервер приложений предоставляет интерфейс работы с системой, как административный, так и пользовательский. При этом данный сервер осуществляет взаимодействие с сопутствующими серверами приложений для обеспечения функционирования бизнес-логики по регистрации данных во время общения по телефону.

Телефонные вызовы в систему поступают из городской телефонной сети через Transcode сервера [1], которые в свою очередь, маршрутизируют вызов на один из Media серверов, к которым подключены конечные абоненты [19].

Сервера журналирования выгружают журналы вызовов с серверов телефонии (media и transcode) по протоколу SSH для последующей локальной обработки.

Для оперативного наблюдения за работой пользователей системы используются WEB приложения, взаимодействующие с БД Redis. Информация из БД используется для финансовых расчетов компании.

Офисная система телефонии являются частью систем. Администрирование системы осуществляется через сервер приложений системы. Коммутация вызовов происходит через сервера транскодинга системы. Сам сервер телефонии представлен всего одним экземпляром, а резервное копирование может осуществляться средствами системы виртуализации.

1.1.2. Краткая характеристика отдела системного администрирования и его видов деятельности

Ключевыми подразделениями проектных решений являются отделы системного администрирования и технической поддержки, работающих воедино с целью поддержания инфраструктуры в постоянной боевой готовности.

Ключевыми задачами смежных отделов является создание отказоустойчивой платформы для размещения разрабатываемых сервисов, применяемых систем, обеспечивая должный уровень SLA доступности.

Отдел системного администрирования берет на себя содержание backend-составляющих, обеспечивает создание и поддержание отказоустойчивых каналов связи с удаленными площадками, в то время когда специалисты технической поддержки выявляют и исправляют ошибки связанные с локальными неисправностями а также подготавливают отладочную информацию для проведения анализа разрабатываемого программных продуктов.

1.2. Моделирование процесса автоматизации управления сетевыми ресурсами

Каналы передачи данных являются основополагающей составляющей в организации, оказывающей телекоммуникационные услуги в различных их направлениях для ряда крупных компаний, в которых для решения вопросов требуются подключение внешних исполнителей для закрытия данного направления деятельности.

В рамках преддипломной практики много времени было уделено изучению используемых ресурсов и применяемых топологий сетевых коммуникаций [6].

Внутренняя IT инфраструктура представлена несколько информационными сетями с разной сетевой адресацией, в рамках адресации 192.168.0.0/16. Информационная сеть пользователей логически отделена от серверной сети, в которой располагаются узлы систем телефонии. Взаимодействие с сетью Интернет осуществляется через несколько узлов, как коммутационных устройств, так и серверов с несколькими сетевыми интерфейсами, имеющими как внешнюю, так и внутреннюю сетевую адресацию. К внутренней IT инфраструктуре имеют подключение внешние пользователи систем телефонии, посредством Remote Access VPN функционирующем на базе OpenVPN.

Для узлов систем телефонии используются виртуальные машины на базе имеющейся среды виртуализации — VMware. Используемые операционные системы серверов систем телефонии включают Debian версии 9 и CentOS версии 6.

Вся телефонная коммутация и взаимодействие между узлами обеспечивается серверами телефонии с использованием протокола SIP / UDP. Системы телефонии включают сервера приложений обеспечивающие работу бизнес логики.

Всё взаимодействие пользователей с системами телефонии, вне зависимости от роли пользователя, осуществляется с рабочей станции посредством браузера и сопутствующего программного обеспечения.

Из описанного выше принципа построения бизнес-процессов можно сделать вывод что неотъемлемой составляющей всех систем является среда передачи данных [7], связывающая сервисы, удаленные площадки и заказчиков между собой, образуя единый контур инфраструктуры.

Если обратиться к топологии построения сети [4], то данная топология построения сетевой инфраструктуры выполнена по принципу схемы звезда, соединяя удаленные филиалы с единым ядром системы.

В рамках принятого корпоративного стандарта за управление маршрутизацией отвечают решения вендера Mikrotik различных форм-факторов. В настоящее время данное решение является конкурентно способным и выполняет функционал в данном сегменте на ровне решений от Juniper, Cisco и Huawei. В тоже время как за управление коммутацией принято решение от Cisco, применяя различные протоколы физического стекирования оборудования, агрегации каналов для повышения уровня отказоустойчивости и безаварийной наработки ресурсов.

С целью сокращения требуемого времени для оперативной смены применяемой топологии в связи с производственной необходимостью или ставящимися задачами на переконфигурирование оборудования все приходящие каналы от вышестоящих провайдеров и внешних заказчиков приходят на ядро сети в стеки коммутаторов, располагающихся в центрах обработки данных. При наличии нескольких ап-линков каналов одного рода, к примеру, при наличии резервных линий от провайдера они заводятся в единый стек, но в разные физические блоки коммутаторов для защиты от простоя в случаях отказа одного из физического элемента стека, при необходимости данные линии агрегируются по технологии EtherChannel, создавая при этом эффект избыточности.

Вся сетевая инфраструктура находится в четырех центрах обработки данных с классом отказоустойчивости Tier3, что согласно международной классификации является одной из наивысшей, обеспечивающей уровень отказоустойчивости оборудования до 99.9% процентов, что в пересчете на временные ресурсы соответствует максимальному времени недоступности в 23 минуты за 1 месяц в худшем раскладе ситуации, что бывает довольно таки редко.

Связь разрабатываемых сервисов с внешним миром осуществляется через собственную автономную сеть (AS), содержащую в своем объеме 512 Pi адресов, образуя две независимые подсети с маской 24 бита (255.255.255.0),

являющейся минимально принимаемой для анонсирования во внешнюю сеть по Border Gateway Protocol (BGP), выполняющему функцию пограничного маршрутизатора во внешнюю мировую сеть через вышестоящих провайдеров.

В связи с вышеописанным, имея в арсенале множество физических ресурсов, резерв по принципу N+1 невозможно обеспечить должный уровень доступности ресурсов и оказываемых услуг, в связи с чем требуется разработка и внедрение системы выбора оптимального маршрута интернет соединения, которая будет основываться на множестве факторов, анализе и заранее заложенной информации о проведении ремонтных работ для построения таблицы маршрутизации, способствующей снижению времени недоступности и общего простоя предоставляемых ресурсов.

1.3.Постановка задачи на разработку информационной системы для поддержки принятия решения в сфере управления информационными ресурсами

В рамках произведенного анализа, направленного на изучение «боли» отдела системного администрирования были выявлены следующие аспекты функционирования сети.

В текущих условиях построенная сеть выполняет базовый функционал резервирования, так, например, основной шлюз физически резервируется с использованием технологии VRRP, т.е. при выходе одного из устройств маршрутизации роль основного шлюза возьмет на себя резервный блок. В каждый из удаленных офисов или филиалов заведено по два независимых интернет провайдера, поверх которых подняты GRE туннели в двух направлениях подсетей центрального ядра предприятия, тем самым имея независимые маршруты как со стороны удаленного офиса, так и ядра сети на внешних IP адресах AS системы.

Внутренняя маршрутизация выполнена с использованием протокола динамической маршрутизации OSPF [14], в основе которого лежит идеология отслеживания состояния канала. Данный протокол активируется сразу по четырем транковым туннелям, обеспечивая в базовом функционале перестроение маршрутизации без дополнительного участия оператора или иной вспомогательной системы только в случае полного отключения одного упомянутых туннелей, но не представляет функционального применения в случаях наличия существенных потерь и задержек во времени на передачи пакетов информации, что негативно сказывается на качество голосовых соединений, онлайн трансляций информации и иных сервисов.

На данной проблемой было сконцентрировано особое внимание сотрудников в рамках преддипломной практики сотрудниками отдела системного администрирования, что и легло в ключевую основу данной работы.

Разрабатываемая система должна оказывать поддержку принятия решения в сфере управления информационными и сетевыми ресурсами по следующим параметрам:

- Интерактивное отображение общей схемы коммуникации с получением обратной связи от применяемых устройств;
- Автоматическое управление внутренней маршрутизацией при возникновении потерь на каналах связи;
- Наличие интерактивных уведомлений для обслуживающего персонала;
- Наличие возможности автоматической самодиагностики в случаях массовых отказов и аварий на выделенных каналах связи, где применение трассировки не представляется возможным.

1.4. Анализ существующих разработок и обоснование выбора технологии проектирования информационной системы

В рамках детального изучения материалов найти ранее описанные варианты решения данной проблемы не представилось возможным, ввиду отсутствия аналогичных решений в открытых источниках или идей и заготовок для дальнейшего развития.

В связи с массовым применением оборудования Латвийского производителя – Mikrtoik надо были изучены возможные варианты внешнего управления операционной системой RouterOS,представленные в таблице № 2.

Таблица 2 - Способы взаимодействия с Mikrotik

Метод управления	Особенности применения	Возможность применения для автоматизации	Метод связи
COM-Port	Физическая реализация подключения, для управления необходимо внешнее устройство	+	Внешний COM-Port
Winbox	Графический интерфейс управления Router OS	-	8291 / TCP
SSH	Сетевой протокол удаленного консольного управления, стандартизированный RFC 4251 Возможна организация авторизации по сертификату, повышая уровень защищенного соединения	+	22 / TCP
Telnet	Сетевой протокол удаленного консольного управления, стандартизированный RFC 854 в отличии от SSH не имеет никаких алгоритмов шифрования передаваемых команд	+	23 / TCP
API	Протокол открытого сетевого взаимодействия, приведенный к единому стандарту вызовов функций операционной системы	+	8728 / TCP

Проанализировав возможные методы организации управления устройствами Mikrotik в рамках решения поставленной задачей будет оптимальным применение способа связи по API, применяя библиотеки для языка программирования Python, такие как RouterOS_API и тому подобные, в связи с наличием больших преимуществ по сравнению с SSH, т.к. имеется стандартизированный синтаксис коммуникации и структурированный метод получения результата.

Вывод к главе 1

Стандартный функционал оборудования имеет довольно таки обширный перечень реализуемых на нем решений, соответствующего уровня для сегмента среднего и малого бизнеса, но для повышения уровня отказоустойчивости и решения нетривиальных решений требуется внедрение системы оценки состояния сети и содействия в принятии решения при выборе оптимального маршрута интернет соединения.

Глава 2 Разработка и реализация проектных решений

В рамках описанной системной инфраструктуры были представлены основные элементы, участвующие в построении телекоммуникационной сети. В представляемой главе работы будет представлен итерации реализации системы.

2.1 Создание системы мониторинга состояния сетевой инфраструктуры

Для оценки текущего состояния сетевой инфраструктуры с целью определения итогов разрабатываемого функционала необходимо внедрение системы мониторинга инфраструктуры, являющейся неотъемлемой частью итогового функционала.

В качестве штатной инсталляции было принято решение о внедрении системы мониторинга Zabbix [3]. Первоначальное знакомство с этой системой удалось провести в рамках производственной практики, так что уже с более глубокой детализацией появилась возможность подойти к этому вопросу.

В рамках изученной инфраструктуры удалось выяснить что на территории компании существует несколько развернутых инсталляций Zabbix, но их общая сущностью не предоставляет должно уровня детализации бизнес-структуры, в связи с чем было принято решение о создании новой с последующим переносом прежних триггеров на новый сервер.

Инсталляция располагается на системе виртуализации VMWare с операционной системой Debian 11. Скриншот, представленный на рисунке №2 отображает применяемые средства виртуализации.

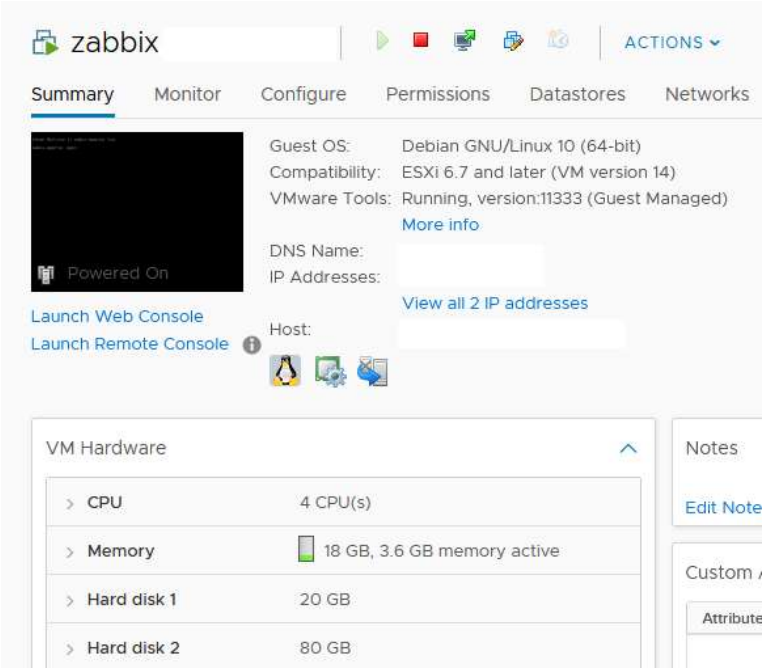

Рисунок 2 – Характеристика виртуальной машины для сервера Zabbix

Ресурсы, выделенные на сервер: CPU – 4 ядра, RAM – 18 Gb, HDD – 100 GB, что позволяет в настоящее время охватить мониторингом всю инфраструктуру, но при необходимости можно расширить с использованием системы виртуализации.

Сервер Zabbix [9] находится в изолированном сегменте сети, с осуществлением доступа к Web-интерфейсу на основании LDAP-авторизации и ACL списка IP адресов.

Для оценки состояния доступности корпоративных сегментов и сервисов на мониторинге должны находиться:

- Центральные узлы сетевой коммутации;
- Узлы удаленных хостов внешних офисов;
- Сервисы разрабатываемых и применяемых приложений;
- Физические сервера;
-Доступность операционной системы кластеризации системы виртуализации;

- Интерфейсы внешнего управления серверными стойками iLO;

- Узлы, отвечающие за анонсирование автономной сети (AS);

- Физическая инфраструктура СКС между Центрами обработки данных.

Для реализации возможности разнесенного мониторинга возможно внедрение внешних Zabbix Proxy, физически располагающихся в географически распределенных местах или сегментах сети.

На этапе рассмотрения вариантов реализации было принято решение о разворачивании инфраструктуры в несколько этапов:

- Актуализация схемы коммутации в формате документа Microsoft Visio;

- Добавление хостов в систему мониторинга;

- Разработка интерактивных карт по отдельным систематизированным направлениям;

- Группировка отдельных карт в обобщенные Dashboards одинаковой сущности;

- Настройка доступа к ресурсам мониторинга по распределенным группам доступа;

- Настройка уведомлений по ответственным подразделениям.

И так, приступая к итоговой реализации необходимо подготовить ряд шаблонов мониторинга. Основным применяемым оборудованием является – Cisco, Mikrotik [8]. В сообществе Zabbix доступны следующие реализации шаблонов для мониторинга по SNMP протоколу.

Zabbix Community доступно в открытом доступе по ссылке - https://share.zabbix.com/ , основной задачей которого является систематизация и обобщение ранее разработанных решений, доступных в свободном доступе.

2.1.1 Добавление хостов

Ключевые параметры, необходимые для организации SNMP мониторинга:

- IP / DNS имя конечного хоста;
- Версия SNMP протокола, значение SNMP community;
- Порт SNMP протокола;
- Применяемый шаблон мониторинга.

Основные действия для управления хостами осуществляются в разделе Configuration – Hosts

Для единочтения наименования [16] хостов было принято решение об использовании в host name сокращенное название города (офиса), в котором находится оборудование, его функциональная роль, номер или буквенное обозначение релевантности, представленные в таблице №3.

Таблица 3 - Варианты трактования для систематизации.

Составная часть	Пример сокращения
Город расположения оборудования	Тольятти Фрунзе – TLT-Fr Тольятти Баныкина – TLT-Ban Казань – KZN Центр обработки данных M9 – M9 Центр обработки данных MP4 – MP4
Выполняемая функция	Маршрутизатор – GW Свитч – SW Сетевое хранилище – NAS Сервер – SRV

Продолжение таблицы № 3

Порядковая систематизация	Для маршрутизаторов при наличии только двух применяется: Master / Slave Для всего остального оборудования – 01, 02, 03 и т.д.

Все указанные наименования в системе мониторинга должны строго соответствовать названиям с Racktables – системе учета физических кабельных подключений, использования адресного пространства подсетей.

Каждый хост должен относится к функциональной группе для возможности последующей систематизации и массового изменения конфигураций оборудования – поле Groups.

Применяемый шаблон соотносится в соответствующей вкладке – Templates.

Проверить корректность введенных данных можно на основании индикатора доступности хоста, при отсутствии связи с определенным ресурсов будет выделен красной пиктограммой с предоставляемой минимальной диагностикой причины невозможности соединения с хостом, представленные на рисунке 3.

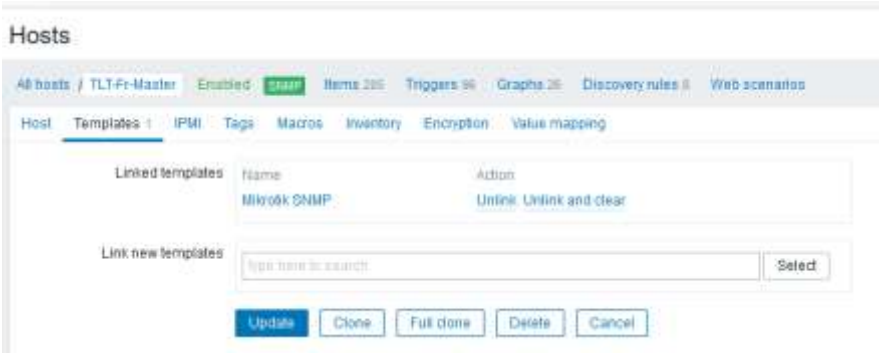

Рисунок 3 – Карточка устройства в системе мониторинга

2.1.2 Методы организации мониторинга

Протокол Simple Network Management Protocol, далее по тексту – SNMP, работает на прикладном уровне L7 сетевой модели OSI и встроен во все сетевые операционные системы, характеризующийся основным применением портов управления – 161, 162 UDP, но при необходимости которые могут быть переконфигурированы на иные с целью защиты предоставляемой информации в дополнении к применению нестандартного значения SNMP community и применяемого в третьей версии SNMP вариантов шифрования с использованием заранее заданного ключа шифрования или применяемого TLS сертификата, обеспечивающего Point-To-Point шифрования, минимизирующего перехват трафика.

Основной задачей протокола SNMP является предоставление информации диагностической информации о состоянии оборудования, проходящего трафика через интерфейсы, их физический статус и состояние триггеров отвечающих за статус физических компонентов оборудования без необходимости прямого подключения на оборудования и использования дополнительных команд, предоставляя такие данные как состояние встроенных блоков питания, установленных блоков охлаждения, датчиков температуры и иных показателей.

Система мониторинга Zabbix активно развивается сообществом, выпуская в свет ежегодно поддерживающиеся Stable версии программного продукта с возможностью безболезненного обновления на вышестоящую текущую или мажорную версию выпуска без потери данных, что повышает возможность применения современных реализаций функционала, идти в ногу со временем реализуя пожелания сообщества.

В связи с множеством постоянно внедряемых функций для графического функционала (GUI) Zabbix [21] добавление однотипных хостов возможно с использованием функции клонирования, задавая только новые имена хостам и соответствующий способ подключениям к ним.

В рамках подготовки технологической платформы для диагностики состояния сети было включено в мониторинг следующее количество показателей, систематизированные значения которых отображены в таблице 4.

Таблица 4 - Системные показатели мониторинга

Системное значение	Общее количество	Доступно в настоящее время	Отключенные параметры
Hosts - общее количество хостов	285	266	19
Items - значение количества показателей, находящихся на мониторинге	44806	42834	1972
Triggers - показатели мониторинга, имеющих показатели доступности	29277	28335	942

2.1.3 Распределенный мониторинг

В связи с повышенной нагрузкой на Zabbix Server [18] и повышенным значением метрик, находящихся в очереди на обработку данных было принято решение о запуске дополнительных Zabbix Proxy, позволяющих снять нагрузку, предоставить распределенный географический мониторинг и возможность записи метрик в момент отсутствия связи с основным сервером Zabbix для проведения пост-анализа произошедших аварийных ситуаций с различных частей корпоративной сети для сужения поиска неисправного участка или узла сети. Рассматриваемая во время реализации проекта нагрузки представляется на рисунке № 4.

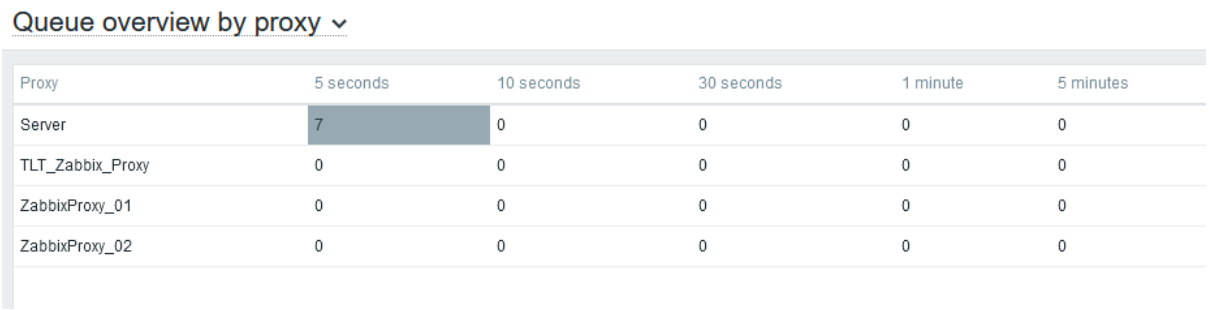

Рисунок 4 - Очередь нагрузки на узлы опроса и мониторинга

Скриншот раздела Administration – Queue – Overview by proxy, характеризующий показатели состояния очереди опроса кластера мониторинга [20], исключающее возможность пропуска аварийных ситуаций в связи с нахождением контролируемой метрики в очереди на опрос, что может негативно сказаться на время обнаружения открытой проблемы.

Представление описаний узлов, участвующих в опросе сетевой инфраструктуры отображается на таблице № 5.

Таблица 5 - Применяемые распределенные узлы опроса

Наименование	Место нахождения	Функциональное применение
Zabbix Server	Хост ESXI-04, Москва Центр обработки данных MP-4	Основной узел мониторинга, мониторинг через основной сегмент сети AS
ZabbixProxy_01	Хост ESXI-01, Москва Центр обработки данных MP-4	Снятие нагрузки с основного Zabbix Server, мониторинг через основной сегмент сети AS
ZabbixProxy_02	Хост ESXI-21, Москва Центр обработки данных M9	Снятие нагрузки с основного Zabbix Server, мониторинг через вспомогательный сегмент сети AS для оценки доступности удаленных ресурсов поверх иного вышестоящего провайдера
TLT_Zabbix_Proxy	RaspberryPi 4, офис в городе Тольятти	Мониторинг ресурсов с удаленного офиса для контроля корректности доступности корпоративной сети вне контура центров обработки данных

При первоначальном изучении корпоративной внешней автономной сети (AS) было выявлено что имеющийся набор Pi адресации анонсируется вышестоящим интернет-провайдерам единым блоком (/23), что при возникновении аварийных ситуаций не предоставляет никакой гибкости в управлении трафиком и исключает возможность незаметного перестроения BGP маршрутизации без потерь пакетов и возникающих ситуаций простоя ресурсов инфраструктуры. В связи с чем, в рамках обсуждения с архитектором сети, было принято решение о разделении AS на два сегмента с маской 255.255.255.0 (/24), что способствует более гибкому управлению трафиком [10] и возможности перехода на резервный узел связи на стороне ЦОД в случае возникновения проблем с доступностью основного.

После успешного ввода в эксплуатацию мониторинга появилась единая платформа [2], объединяющая все корпоративные узлы в едином месте, предоставляя интерактивное представление и возможность сбора статистики для последующей рефлексии и предоставления вариантов изменений, предотвращая возникновение проблем в будущем, наглядный вариант представления которых отображается на рисунке № 5.

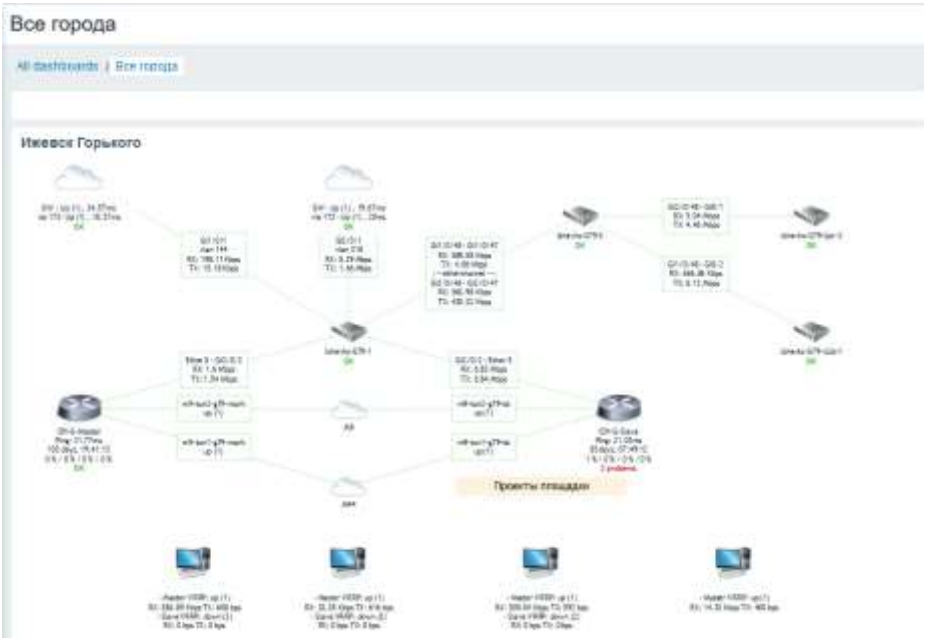

Рисунок 5 - Формат предоставления данных по состоянию удаленных городов

Для контроля и выявления информации по автономной системы был подготовлен расклад автономной системы (AS), наглядно продемонстрированной на рисунке № 6.

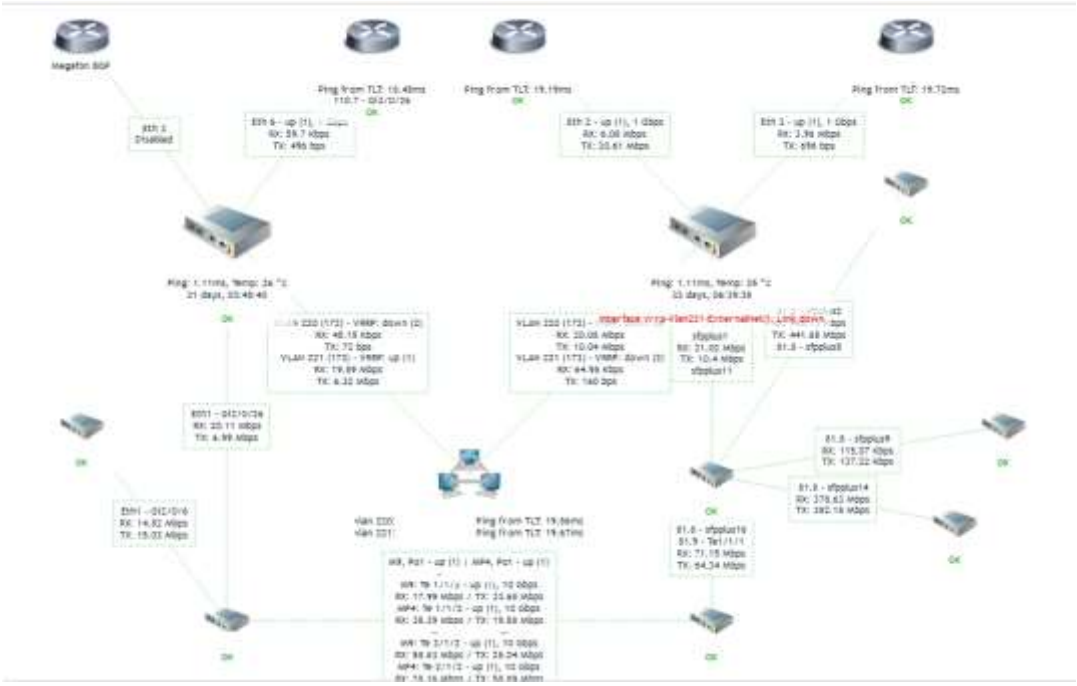

Рисунок 6 - Формат предоставления данных
об анонсировании внешней AS

2.2. Анализ поступающей информации

Оперативное решение возникающих проблем является неотъемлемой задачей отдела системного администрирования. Появившаяся система мониторинга с интервалом от 5 секунд до 1 минуты, в зависимости от значимости ресурса, производит сбор метрик и последующий анализ в режиме реального времени для определения отклонений и возникающих проблем.

Основное внимание нацелена на повышение значений следующих показателей [12]:

- Общая доступность ресурса;
- Доступность удаленного ресурса через канал связи со временем ответа не более 80 секунд.

Произведя анализ возникающих проблемных ситуаций можно разделить на следующие:

1. Параметры соединения: в удаленном городе два независимых канала связи, два физических маршрутизатора, внутренняя маршрутизация с использованием протокола OSPF [17].

Проблема: полный выход основного канала связи

Диагностика: внешний IP не отвечает на ICMP запросы с различных узлов

Возможные проблемы: недостаточная ширина канала резервного соединения, возможность наличия статических записей маршрутизации через основного провайдера, необходимость привлечения оперативного персонала для контроля ситуации

Решение: Автоматическое перестроение таблицы маршрутизации [13] по протоколу OSPF, трафик переходит на резервный канал связи до восстановления основного

Представление проблемы**:** отображено на рисунке №7

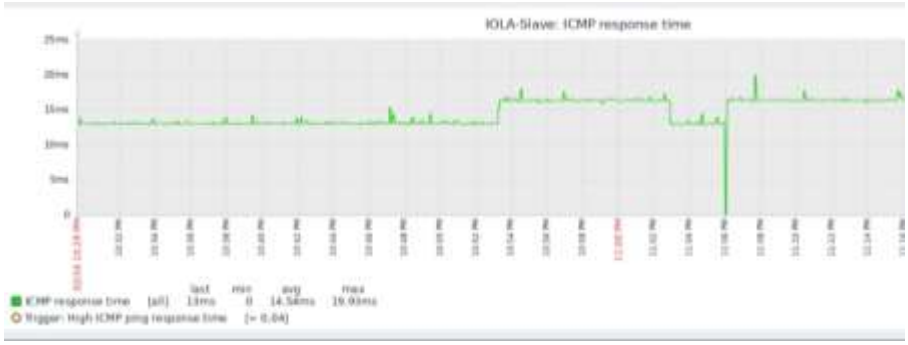

Рисунок 7 – представление проблемы

2. Параметры соединения: в удаленном городе два независимых канала связи, два физических маршрутизатора, внутренняя маршрутизация с использованием протокола OSPF.

Проблема: нестабильность работы основного канала связи

Диагностика: внешний IP не отвечает на часть ICMP запросов / отвечает с повышенным временем ответа. Ситуаций фиксируется поверх основного и / или резервного узла аггрегации.

Возможные проблемы: возможен частый запуск процедуры перестроения таблицы внутренней маршрутизации, сопровождающийся на это время временной полной недоступностью ресурсов. В случае неполного восстановления основного канала маршрутизация перестроится обратно через основной канал связи, и далее при очередном проблемном временном интервале трафик перейдет через полную недоступность на резерв. Ситуация может перейти в цикличный режим, для исправления требуется привлечение оперативного персонала.

Решение: Ручная смена приоритета каналов, понижение приоритета у нестабильно работающего основного канала связи до локализации проблемы со стороны провайдеров.

Представление проблемы: отображено на рисунке №8

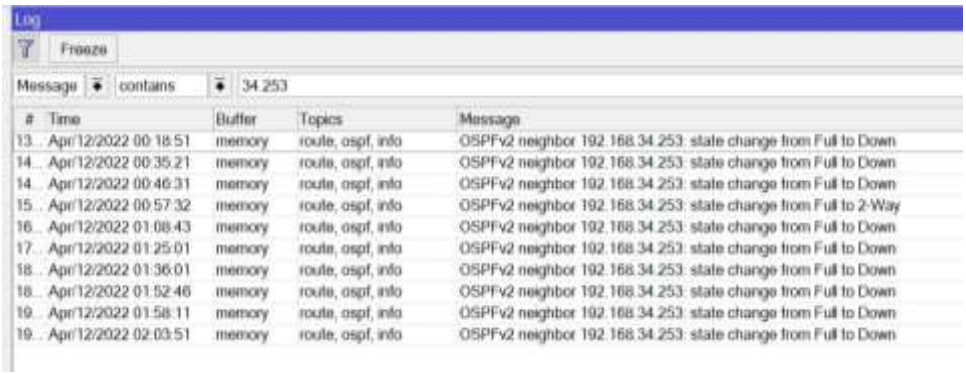

Рисунок 8 – представление проблемы

3. Параметры соединения: в удаленном городе два независимых канала связи, два физических маршрутизатора, внутренняя маршрутизация с использованием протокола OSPF.

Проблема: работа основного канала связи с повышенным временем ответа / проявляющимися потерями пакетов

Диагностика: внешний IP отвечает на ICMP с повышенным временем ответа / фиксируется потеря пакетов. Ситуаций фиксируется поверх основного и / или резервного узла агрегации. Постановка удаленного терминирующего узла на детальный мониторинг с использованием сервиса IP SLA на оборудовании решения Cisco.

Возможные проблемы: возможно проявление в виде искаженного голоса IP-телефонии, появления эха во время разговора, пропадание части слов.

Решение: Требуется ручная смена приоритета каналов, понижение приоритета у основного канала связи доступного с повышенным временем ответа до локализации проблемы со стороны провайдеров.

Представление проблемы: отображено на рисунках №9, 10

Рисунок 9 - Ситуация фиксации массовой потери пакетов

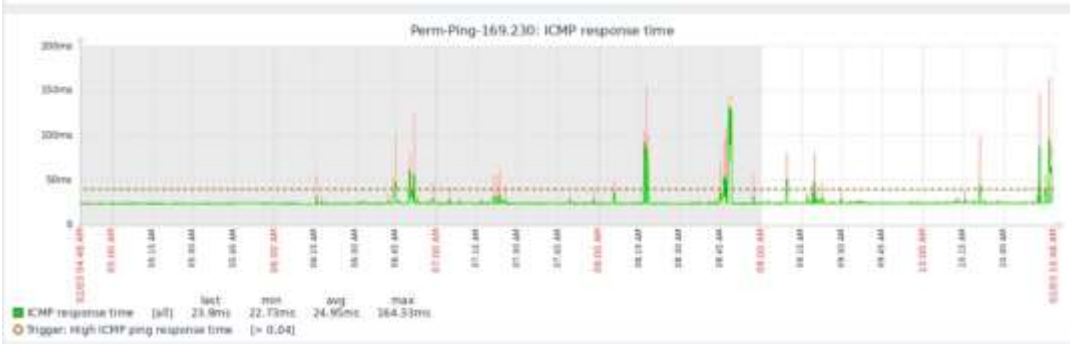

Рисунок 10 - Ситуация фиксации повышенного времени ответа на ICMP-запрос

4. Параметры соединения: в удаленном городе один канал связи, один физический маршрутизатор, внутренняя маршрутизация с использованием протокола OSPF поверх двух туннелей до терминирующих узлов GRE-туннелей в ЦОД, выходящих во всемирную сеть через разных провайдеров.

Проблема: канал связи недоступен поверх основного туннеля

Диагностика: внешний IP адрес не отвечает на ICMP запрос.

Возможные проблемы:

1) До разделения AS на два сегмента развитие ситуации не имело положительного исхода ввиду полной недоступности канала связи

2) После разделения AS на два сегмента ситуация недоступность удаленного ресурса может наблюдать поверх только лишь одной части AS, когда в другом сегменте ресурс будет штатно доступным.

Решение:

1) Необходимо немедленное вмешательство инженерного состава для составления обращения в адрес интернет-провайдера для проведения диагностики причины возникновения неисправности и предприятия мер для локализации аварии.

2) Принудительный перевод исходящего и входящего трафика на туннель поверх иного сегмента, где проблема не наблюдается

Представление проблемы: отображено на рисунке №11

Рисунок 11 – представление варианта проблемы

Удаленный ресурс поверх одного сегмента недоступен, в тоже время на мониторинге через иной сегмент доступен, но с повышенным пингом, в меньшей степени сказывающимся на доступность ресурса вцелом. Данный график, согласно производимым наблюдениям, соответствует моменту

перестроения BGP соединения на нашей стороне с вышестоящим интернет-провайдером или чисто в сети вышестоящего интернет-провайдера.

Представленные основные четыре исхода развития событий описывают основные проблемы появление которых сказывается в негативную сторону при организации каналов связи у удаленными филиалами поверх всемирной сети Интернет.

2.3. Протокол динамической маршрутизации OSPF

В крупной распределенной сети при наличии нескольких вариантов прохождения трафика для уменьшения зависимости от ручных переключений оперативным персоналом применение статической маршрутизации является неприемлемой и не соответствует современным применяемым технологиям.

В таком случае для организации сети применяется один из протоколов маршрутизации, отвечающий за построение таблицы внутренней маршрутизации на основе технологии отслеживания состояния канала (link-state technology), дополнительно при вычислении закладывает значение стоимости соединения канала для определения более ревалентного в случаях если значение одного канала выше другого, а также при равных значениях может балансировать нагрузку между интерфейсами с равными коэффициентами значений. Описанная схема отображена на рисунке №12.

Рисунок 12 - Развертка построения маршрутизации в разрезе одного удаленного офиса

В качестве терминирующих узлов со стороны Центров обработки данных для приходящих удаленных каналов связи применяются облачные решения Mikrotik Cloud Hosted Router, располагающиеся внутри виртуальной инфраструктуры с возможность миграции в случае выхода из строя физического сервера, отвечающего за выполнение процессов Mikrotik CHR.

Применяемая инфраструктура базируется на принципах защиты от парализования деятельности ввиду отказа одного узла, в связи с этим применяется принцип N+1 для возможности временного вывода из строя одной ноды сети на время планового обслуживания, процедуры обновления прошивок операционной системы RouterOS без сказывания на показатели доступности ресурсов.

В базовой версии решений, основанных на использовании RouterOS нет встроенных пакетов для обеспечения маршрутизации по таким протоколам как OSPF, BGP нет. Для их установки необходимо загрузить дополнительные

пакеты, в зависимости от применяемой физической платформы, с сайта производителя, разместить их в физической памяти устройства, произвести его перезагрузку, после чего в графической платформе управления Winbox появится блок Routing и иные, в зависимости от комплектности устанавливаемых расширений.

Основное управление осуществляется в разделе Routing – OSPF, при переходе в который открывается возможность тонкой настройки выбранного протокола. Для добавления удаленного узла в общекорпоративную сеть необходимо поднять туннельное соединение, связывающее ЦОД с офисом. Для решения данной задачи можно подойти одним из следующих способов, представленных в таблице № 6.

Таблица 6 - сравнение возможных путей организации каналов связи

Тип канала связи	Достоинство	Недостатки
Выделенный канал связи - L2 (аренда L2-VPN канала)	Невозможность вмешательства из внешней инфраструктуры, никак не связанной с корпоративным сегментом	Повышенная стоимость предоставляемого канала Сложность при диагностике неисправностей, связанных с проявляющейся потерей пакетов, повышенным временем ответа удаленной точки
Канал связи – L3 (GRE, IPSec)	Канал связи работает поверх обычного интернет соединения, может быть гибко перестроен без участия провайдера Не требует дополнительной оплаты за соединение помимо абонентской платы	Возможность подверженности от внешних факторов (DDoS-атаки, блокировка трафика) Прямая зависимость от вышестоящих интернет-провайдеров Требуемые дополнительные вычислительные ресурсы для защиты передаваемых данных

Организовав канал связи одним из удобных способов можно приступать к настройке маршрутизации с использованием OSPF на устройствах Mikrotik.

После добавления туннельного соединения необходимо произвести подготовительную работу на стороне подключаемого узла. Добавление нового узла должно сопровождаться:

- Указанием Router ID – номер удаленного устройства, соответствующий одному из реальных IP адресов;

- Заданием In и Out фильтров, отвечающих за отсечение принимаемых и отправляемых маршрутов внутри корпоративной сети.

Вариант управления через консоль Winbox представлен на рисунках № 13,14.

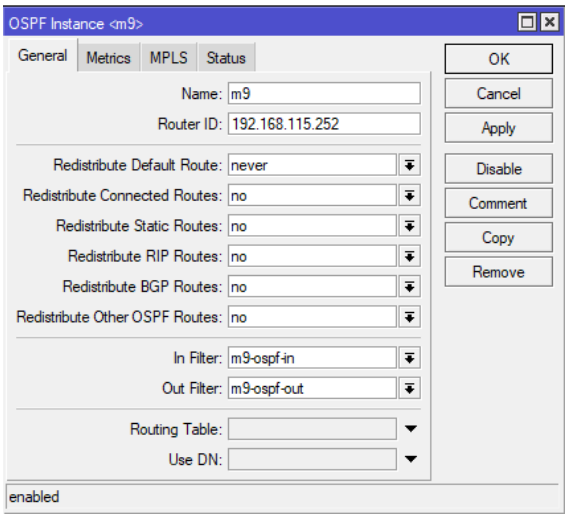

Рисунок 13 - Окно управления Instances (раздел Routing – OSPF – Instances)

Произведя указанные действия в разделе Instances перемещаемся во вкладку Networks, требующую от нас значений адресных пространств, подлежащих к внутреннему анонсированию, привязанных к определенной Area (соответствующей зоне)

Рисунок 14 - Окно управления Networks (раздел Routing – OSPF – Networks)

Добавление необходимо производить с указанием адреса сети и применяемой маски подсети.

Адресное пространство IPv4 представляет собой несколько выделенных наборов адресных пространств, стандартизированных стандартом международного классификатора - RFC6890.

К сетям, предназначенных для внутренней маршрутизации относятся следующие, упомянутые в таблице 7.

Таблица 7 - Распределения адресов для внутренней маршрутизации

Начало диапазона	Конец диапазона	Обобщенная маска
10.0.0.0	10.255.255.255	255.0.0.0 (/8)
172.16.0.0	172.31.255.255	255.240.0.0 (/12)
192.168.0.0	192.168.255.255	255.255.0.0 (/16)

Правилом хорошего тона при проектировании информационных сетей передачи данных является единых подход к систематизации данных, их едином месте учета, ведении своевременном заполнении в рамках поднимаемых новых соединений и ввода подсетей в эксплуатацию. Для реализации данного направления подходит решение – Racktables, позволяющее помимо управления корпоративным адресным пространством дополнительно вести кабельный журнал, учитывать места монтирования серверного оборудования для последующего избегания проблем с накладкой сетевых пространств и возникновения связанных с этим проблем.

Не отклоняясь от принятых корпоративных стандартов добавим в Racktables вновь добавляемую подсеть 172.18.33.0/26, указав в свойствах понятное имя сети, привязав соответствующий Tag расположения адресного пространства. Вариант учета подсети представлен на рисунке № 15.

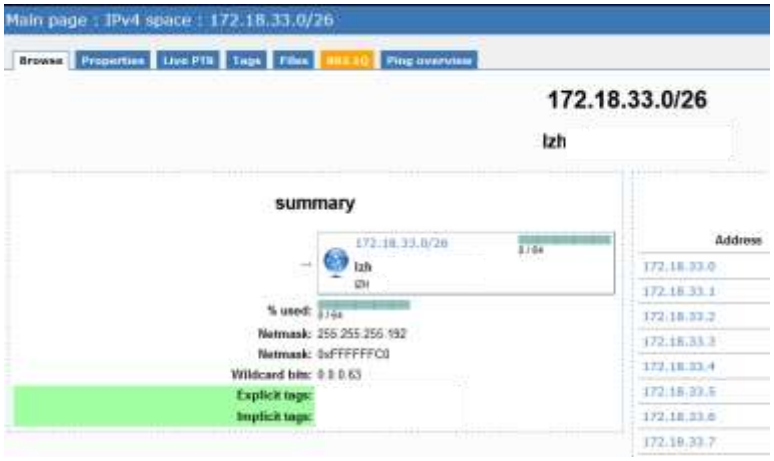

Рисунок 15 - Указание параметров сети в Racktables

Как можно заметить, после успешного добавления подсети она встала в порядковое место общей внутрикорпоративной системы, визуально отображая свободные для распределения участки сети. В случае обнаружения проблемы с дублированием адресного пространства система автоматически уведомит сотрудника и предложит варианты решения проблемы, вариант распределения адресного пространства отображен на рисунке № 16.

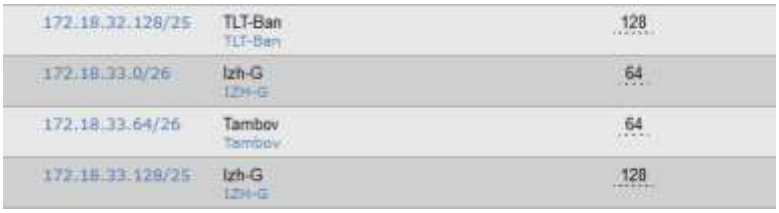

Рисунок 16 - Скриншот Racktables. IPv4 space

После подготовительной работы представляется возможность для продолжения процедуры разворачивания динамической маршрутизации. Выбрав необходимый для добавления физический транспортный интерфейс, задаем ему значение Cost, т.е. указываем приоритет данного канала, применяемый метод аутентификации на данном канале (Authentication), заранее заданный ключ при использовании аутентификации (Authentication Key),

снимаем флаг Passive, отключающий распространение маршрутизации на данный интерфейс, вариант в интерфейсе указан на рисунке №17.

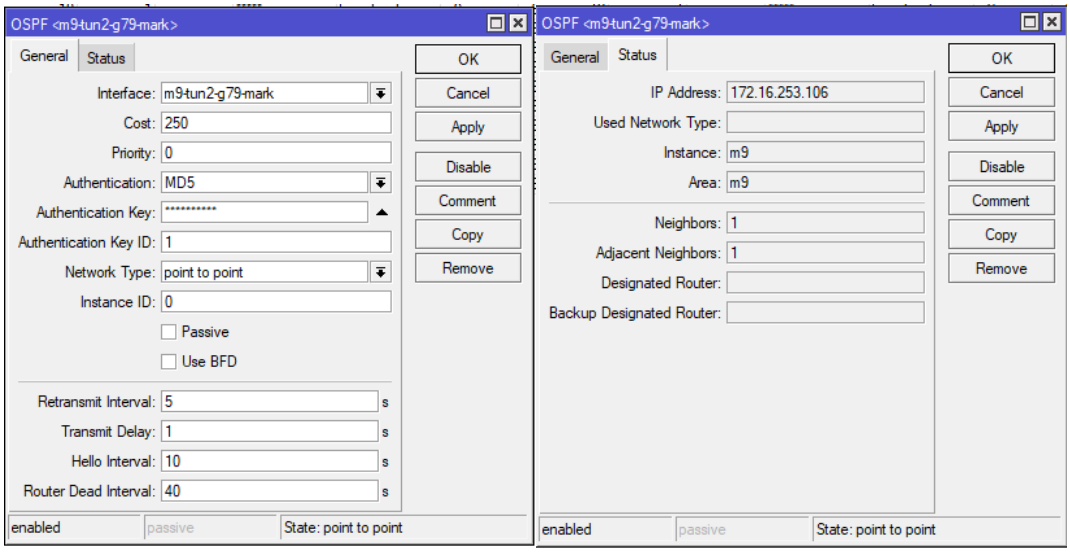

Рисунок 17 - Скриншот раздела Routing – OSPF – Interfaces

После успешного поднятия сессии OSPF в соответствующей разделе статуса появится информация о состоянии канала, его Neighbords (соседним подключениям), а самым ключевым показателем является появлением приходящих маршрутов в разделе IP – Routes с указанием интерфейса по которому приходит маршрут. Наглядное представление на рисунке №18.

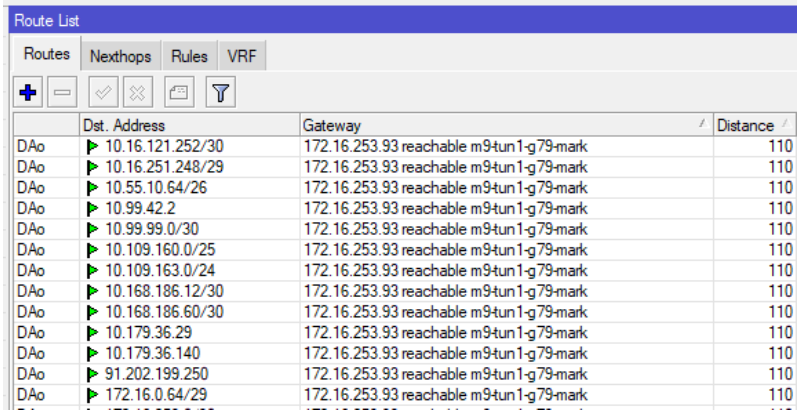

Рисунок 18 - Скриншот раздела IP – Routes

Как уже было сказано ранее, при поднятии OSPF соединения указываются фильтры, отвечающие за фильтрацию принимаемым и отправляемым подсетям. Правилом хорошего тона является корректное указание Out-Filters в сторону центрального узла, жестко указав в него только применяемые в данной ноде сети, для исключения возможности трансляции в корпоративную сеть адресных пространств случайно попавших в маршрутизацию от вышестоящего провайдера по так заведомо называемых Passive интерфейсам, а со стороны центрального узла маршрутизации соответственного In-Filters, отвечающим за принимаемые префиксы. В случаях если в удаленном офисе не требуется доступ в полный корпоративный сегмент можно заложить применение технологии с указанием полного набора фильтрации In/Out Filters.

Применив указанную схему на полный перечень удаленных офисов получилась единообразная сетевая инфраструктура по принципиальной схеме «звезда», наглядно отображенной на схеме в разрезе на Рисунок 12 - Развертка построения маршрутизации в разрезе одного удаленного офиса.

2.3 Постановка проблемы

Описанные проблемы в разделе №2.2 представляемой работы представляют обзор на основные возникающие ситуации в рамках эксплуатации крупной распределенной сети. При прохождении трафика в сети может происходить множество совершаемых действий над ним для организации структуры его передачи, уменьшению возникающих издержек и возможных потерь пакетов. Решение большинства проблем лежит в постоянном наблюдении за средой передачи информации, анализу развития ситуации и принятии решения о необходимости произведения переключений. В связи с предоставлением компанией сервисов и услуг в круглосуточном режиме без выходных и приостановки процесса для обслуживания сети необходим оперативно – дежурный персонал, отвечающий за произведение оперативной работы в случаях возникновения проблемных ситуаций.

Разрабатываемая программа должна оказывать содействие и принятие решений при возникновении ситуаций, подходящий под критический уровень, а также:

- Иметь возможность запуска в ручном / автоматическом режиме работы;
- Являться неотъемлемой интегрированной составляющей системы мониторинга;
- Осуществлять защиту от негативного воздействия в случае массовой аварии, т.е. иметь блокировку от запуска в случае массового инцидента;
- Производить логирование всех производимых действий с целью возможности обращения к этим данным при разборе случившегося инцидента;
- Иметь возможность осуществления комплексного изучения обстановки перед запуском выдаваемых команд на управление;
- Осуществлять оперативный персонал о произведенных переключениях, их основной сути.

Итоговый вид должен представлять собой цельный алгоритм, разделенный на модульную структуру, для возможности выполнения каждого блока отдельно.

Глава 3 Практическая реализация программного продукта

3.1 Применяемый метод реализации

3.1.1 Подготовка оборудования Mikrotik к внешнему управлению

Как было сказано в первых главах работы, широко представленная на мировом рынке в различных вариантах исполнения операционная система на базе Router OS имеет множество вариантов внешнего управления, отвечающих тем или иным актуальным требованиям основным был выбран интерфейс API и язык программирования – Python.

В рамках подготовительного этапа [11] необходимо обеспечить доступ с рабочего места, предназначенного для разработки продукта, а также с центрального сервера Zabbix к порту управления – 8728 TCP.

Выполним подготовительную работу по включению данного сервиса и открытию порта для доступа, в случае применения нормально-закрытого Firewall. Метод управления отображен в таблице №8.

Таблица 8 - Подготовка Firewall к работе

Терминальная команда	Выполняемое действие
/ip service set api address=192.168.10.250, 192.168.56.100 disabled=no	Активация IP интерфейса с доступностью только по двум управляющим адресам
/ip firewall filter add action=accept chain=input dst-port=8728 protocol=tcp src-address=192.168.10.250 add action=accept chain=input dst-port=8728 protocol=tcp src-address=192.168.56.100	Открытие TCP порта 8728 в Firewall в цепочке Input, отвечающей за трафик идущий на сам маршрутизатор

Для первоначальной проверки доступности порта можно использовать приложение Putty, выполнив telnet запрос по IP адресу устройства и соответствующему порту, в случае корректного прохождения трафика будет открыто консольное окно с приглашением ввести команду, представленное на рисунке №19.

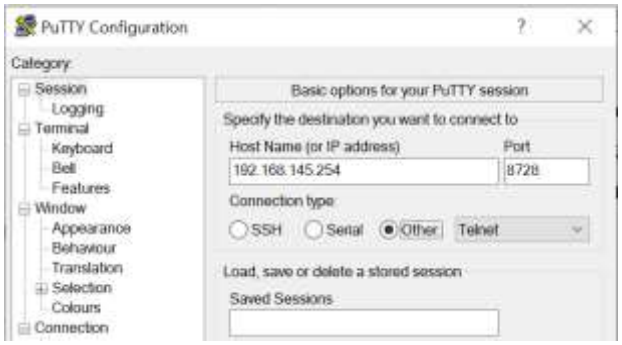

Рисунок 19 - Интерфейс приложения PuTTY,
активно применяемого при сетевом администрировании.

В случае недоступности порта к первоначальной диагностики можно отнести выполнение действий, указанных на 20 рисунке:

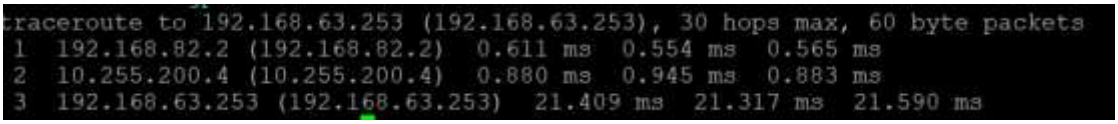

Рисунок 20 - Выполнение трассировки прохождения ICMP
пакетов до хоста

Вариант выполнения трассировки с указанием порта назначения представлен на рисунке 21:

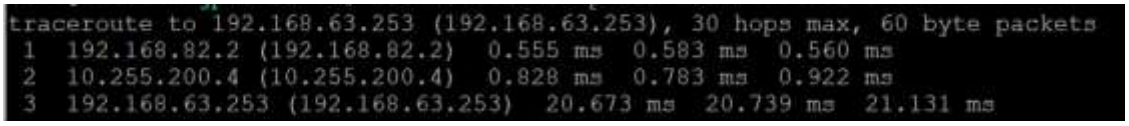

Рисунок 21 - Вывод команды: traceroute -T -p 212 192.168.63.253

Подготовку к работе множества устройств, имеющих доступный порт SSH интерфейса, базовую настройку можно выполнить с использованием

сервиса Ansible, позволяющего выполнить однотипную задачи по заранее описанному интерфейсу.

3.1.2 Создание компонента установления соединения API – Python

В качестве основной библиотеки для Python по направлению Mikrotik была выбрана routeros_api, являющейся разработанной сообществом Mikrotik. Для установления соединения с управляемым устройством необходимо указать сведения об IP адресе, логине, пароле, порту подключения.

Как выяснилось позже, из-за применения различных версий прошивок операционной системы Router OS, имеющих существенные отличия в алгоритмах проведения авторизации по протоколу API было принято решение о введении проверки на установления соединения с передачей пароля с зашифрованном виде, при невозможности запускается дополнительная итерация проверки с передачей пароля в том виде «как есть», за что отвечает дополнительная команда при вызове инициализации соединения plaintext_login=True. При невозможности установления связи с удаленным устройством формируется соответствующее сообщение с занесением в блок уведомления и лог-файла.

3.2 Создание основного блока управления

3.2.1 Работа над используемыми интерфейсами

В текущей топологии управления внутренней маршрутизацией для изменения направления движения трафика необходимо производить изменения в соответствующем разделе Routing – OSPF - Interfaces, изменяя значение Cost на интерфейсе, чей приоритет необходимо изменить, отображенный на рисунке №22 в интерфейсе Winbox.

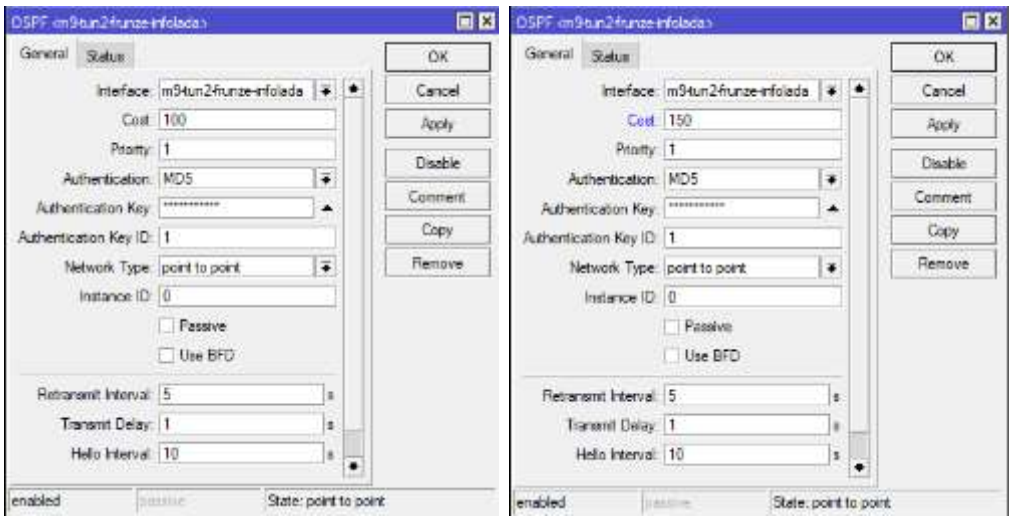

Рисунок 22 - Изменение значения в графическом интерфейсе Winbox.

Для преступления к организации блока алгоритма по изменению основного направления движения трафика необходимо определить какой из существующих интерфейсов является основным, имеется ли активный резервный канал связи. В случае произведения данного действия в ручном режиме необходимо открыть 4 (четыре) интерфейса:

- Основной терминирующий узел в ЦОД;
- Резервный терминирующий узел в ЦОД;
- Основной маршрутизатор удаленного офиса;
- Резервный маршрутизатор удаленного офиса (при наличии).

С целью оптимизации работы кода, при массовом управлении несколькими удаленными филиалами, для избежания дублирования выполняемых действий блок запроса интерфейсов по пунктам 1-2 вынесен в отдельную функцию и запускается один раз при запуске скрипта для формирования массива RoutingOSPFInterface_DC - активных интерфейсов OSPF на стороне Центра обработки данных. Рассмотрим реализацию запроса подробнее.

Для управления устройствами после первоначального установления связи с ними объявляем соответствующие функции, в рамках взаимодействия с

которыми будут производиться последующие действия и команды, так было принято об использовании функций API_TUN1 и API_TUN2.

В связи с крупной географической распределенностью и множеством применяемых ЦОД заведение физических каналов связи по единой коммутации не всегда представляется возможным, в связи с чем в скрипт закладываются дополнительные подпрограммы для решения данных задач. За указание на использование нестандартной схемы коммутации со стороны ЦОД отвечает переменная DC_One_Mikrotik, где к примеру применяется только один агрегирующий узел на стороне ядра сети.

Согласно синтаксиса по использованию библиотеки routeros_api [15], для выполнения запроса на выборку информации используется атрибут get_resource с указанием раздела, значение информации откуда требуется получить. Обратимся к одному из Mikrotik, находящемуся в ЦОД для получения информации: API_TUN1.get_resource('/routing/ospf/interface').get(), полученная информация заносится в массив RoutingOSPFInterface_MP4_RAW, в названии которого префикс “_RAW” отвечает за то, что эти данные взяты из исходного набора выборки и не прошли фильтрацию для исключения данных, не использующихся при работе алгоритма. Выполнив точно такой же запрос по отношению ко второму узлу агрегации получим набор данных, произведем перегонку массивов для создания итогового RoutingOSPFInterface_DC, включающего в себя только OSPF интерфейсы со статусом point-to-point и являющимися включенными на момент запроса. На языке Python данный цикл имеет следующий вид как на рисунке 23:

Рисунок 23 - Листинг кода

Как можно заметить из представленной выдержки кода, из исходных массивов в оперативный RoutingOSPFInterface_DC заносятся значения о названии узла откуда они были импортированы, ID интерфейса на момент

запроса, имени интерфейса и текущее его значение Cost – это минимально необходимый набор значений, требующийся для оперирования и обеспечения логической связанности скрипта.

На этом этапе работа над формированием массива по интерфейсам в ЦОД завершена, при создании результирующего массива по применяемым интерфейсам на ответной стороне применим аналогичный подход, только за исключением того, что перед процедурой запуска перегонки массивов в итоговый, очистим его от возможных прежних ранее записанных в него значений, не относящихся к выполняемой вышестоящей итерации.

3.2.2 Контроль валидности данных

С целью определения корректности поступившей входной информации в тело скрипта необходимо произвести валидацию по следующим направлениям:

- Определение основного и резервного интерфейса в разрезе удаленного офиса;
- Определение наличия «ответной» для основных определенных интерфейсов с целью проверки на возможность произведения переключения в рамках текущего набора данных.

Отправной точкой для запуска скрипта является процедура определения основного и резервного интерфейсов OSPF, работа над переключением которых будет запущена при необходимости выполнения переключений. В сетевом администрировании в различных применяемых протоколах определение так называемых Master и Backup значений осуществляется на основании значения «веса» того или итого интерфейса, пира или опции, но существенным отличием в протоколах является порядок определения, так для одних Master – является значение, имеющее минимальный показатель, когда в тоже время в ином протоколе наблюдается реверсивная политика, где для того чтобы применяемый интерфейс считался системой как основной (Master) – его

значение должно иметь наибольший приоритет. В рамках рассмотрения протокола динамической маршрутизации OSPF, основанный на определение минимальной стоимости соединения, Master интерфейс должен иметь минимальное значение, что подпадает под определение применяемой топологии интерфейса по минимальной стоимости маршрута.

Для упрощения понимания топологии итогового скрипта и возможности быстрого вхождения в используемые функции, переменные и массивы без необходимости длительного изучения алгоритма было принято решение о выводе информации о Master и Backup интерфейсе в соответствующие переменные, являющимися глобально-объявленными, указанными в таблице №9.

Таблица 9 - Описание применяемых переменных

Переменная	Определяемое значение
OSPF_Master_Cost	Значение Cost Master / Backup OSPF Interfaces
OSPF_Backup_Cost	
OSPF_Master_ID	Массив значения параметров описанных для Master / Backup OSPF Interfaces
OSPF_Backup_ID	

Для выборки соответствующих значений воспользуемся соответствующим алгоритмом выборки, пройдясь по массиву OSPF_GRE, содержащему информацию об актуальных интерфейсах. Для нахождения минимальных значений зададим искомым переменным максимально-возможное значением и в ходе прохождения в цикле будем сравнивать является ли представляемое значение Cost в данном шаге по сравнению с текущим минимальным значением, если оно является меньшим по сравнению с содержимым в переменной OSPF_Master_Cost, то присвоим значение данной переменной соответствующему текущему, в случае если сравниваемое значение при сравнении с наименьшим не проходит проверку на сравнение, то

в этом случае запускается вторая проверка на сравнение значения с текущим содержащимся в OSPF_Backup_Cost, т.е. осуществляется проверка на соответствие Backup интерфейсу. По итогам работы данного блока заполняются значения соответствующие основному интерфейсу, через который в настоящее время осуществляется движение трафика и резервного, на который в случае полного выхода из строя основного интерфейса произойдет переключение трафика. По итогам формирования данных значений в блок формирования уведомлений заносятся соответствующее записи для вывода информации в итоговое уведомление о работе скрипта, также данная информация записывается в файл лога.

Если проследить за вложенной в скрипт логикой, то в настоящее время имеется минимальный набор значений, на основании которых можно принять решении о возможности нахождения иного транспортного канала, что выполняется в функции скрипта AnalyzeCostOSPF, производящая в своем теле проверку на наличие второго канала связи. Из вышеописанного блока данную проверку можно осуществить путем сравнения значений (OSPF_Backup_Cost > 9989 or OSPF_Master_Cost > 9999), данный блок обеспечивает проверку на корректность входящие данных.

Согласно топологии OSPF, каждый из участников процессов, отвечающих за прохождение трафика, определяет направление трафика для своего узла в отношении к вышестоящему узлу на основании стоимости соединения маршрута, в связи с тем, чтобы не допустить прохождение входящего трафика по одному интерфейсу, а исходящего по другому необходимо производить зеркальные переключения с двух сторон участников процесса OSPF, а также не допускать произведение некорректных переключений или переключений, выполненных только на некоторую половину процесса. Для осуществления данной проверки необходимо проверить наличия сопоставляемых интерфейсов являющимися основными в филиале интерфейсам в Центре обработки данных. Этот процесс рассмотрен в функции Interface_At_DC, на вход в которую поступает информация об

искомом имени интерфейса. Проверка осуществляется для Master / Backup интерфейсов и является неотъемлемой частью работы скрипта для защиты от некорректной работы.

Убедившись в наличии минимального набора входных данных можно приступать к основному блоку, отвечающему за процедуру переключения. Текстовое описание данной процедуры звучит как поочередная замена значений Cost интерфейсов. Отобразив макет переключения в таблице может иметь примерный формат зафиксированный в таблице №10.

Таблица 10 - Графическое отображение плана переключений

Входной статус интерфейса	Входное значение	Промежуточное значение	Итоговое значение	Итоговый статус интерфейса
Master	150	150	200	Backup
Backup	200	149	150	Master

Из представленной логики работы основные изменения происходят в три логических этапа, опираясь на входящие значения Master-интерфейса, производя изменения на Backup интерфейсе на значение, соответствующее меньшему чем актуальное на Master интерфейсе для смены логической значимости применяемых каналов связи.

Инициализация процесса перестроения идет из вышеописанной функции AnalyzeCostOSPF, путем передачи в соответствующий блок ChangedCostOSPF значений gw – название физического устройства, на котором запускается процесс смены значений, id – текущее порядковое значение ID интерфейса, interface – понятное название интерфейса и cost – присваиваемое значение интерфейсу.

Функция ChangedCostOSPF является операционной для итоговой отправки интерфейсу команды на производимые переключения, при вызове которой осуществляется проверка на блокировку исполнительных команд по времени прошлого запуска, что являлось в свою очередь одним из требуемых

условий в техническом задании на разрабатываемый скрипт. Данная блокировка достигается считыванием информации о дате и времени прошлого производимого переключения и проверка разницы во времени с текущим значением времени, при произведении прошлого переключения после прошествии более 5 минут, то дается разрешение на переключение, в противном случае устанавливается блокировка и команда на исполнение не переключение не приводится в действие с соответствующим выводом уведомления для оперативного персонала.

В теле функции, в зависимости от поступившего на вход значения параметра, gw происходит новая попытка установления связи с устройством в цикле с тремя попытками на инициализацию с целью повышения уровня результативности выполнения скрипта, поскольку его запуск в свою очередь связан с возникающими проблемами на каналах связи. Для передачи управляемых команд на агрегирующие узлы в ЦОД осуществляется по ранее установленным соединениям, поскольку шанс отсутствия связи с ними сводится к минимальному.

Для произведения переключения необходимо получить массив интерфейсов для осуществления переключений, далее удаленного управления применяется связи по id и передаче необходимых новых команд. Часть функции представлена на листинге кода в рисунке 24.

Рисунок 24 - Листинг кода функции ChangedCostOSPF

Как можно заметить из представленного листинга кода после первоначального установления связи с устройством скрипт заносит в свои переменные GW_Slave_Plaint и GW_Master_Plaint информацию о требуемом способе передачи пароля при авторизации, что в свою очередь сокращается время и количество попыток некорректного соединения.

На этом основной функционал работы скрипта завершается и нерассмотренными блоками остались – блок формирования итогового аналитического отчета по произведенным шагам скрипта и статусе их выполнения. При разработке данного блока встала задача получения оперативной итоговой информации в удобном виде для персонала и при возможности без дополнительной оплаты за отправляемые данные, но в тоже время обеспечивая должный уровень защиты отправляемой информации. В качестве рассматриваемых приложений в данном блоке выбор остановился на двух решениях – использовании Rocket.Chat или Telegram.

Rocket.Chat является одним из OpenSourse решений для внутрикорпоративной коммуникации, разворачиваемый на автономном сервере организации что в свою очередь передает возможность для полного контроля над хранимыми данными, имеет различные варианты интеграций с внешними системами управления и транспортных каналов и вторым рассматриваемым продуктом является Telegram, отличающийся от иных мессенджеров открытой системой API для внешних систем выбор пал на использование его в качестве основного, поскольку ряд дополнительных систем оперативных уведомлений построено на его основе.

3.3. Тестирование разработанного продукта

В связи с высокой нагрузкой на сетевые ресурсы и зависимости критических бизнес-процессов от сети передачи данных тестирование было назначено на ночное время в удаленном офисе, завершающем свою работу в 22:00 МСК.

Для запуска выбора зоны, над которой будет производиться переключение на вход скрипта подается значение переменной place, соответствующее выбранной зоне, после чего на основании входных данных происходит сопоставление IP адресов устройств маршрутизации,

расположенных на удаленной площадке и, при наличии, особенности схемы проведения переключения.

В качестве ключевого показателя для оценки получившегося решения будет принято значение количества потерянных пакетов при произведении переключения, что в свою очередь напрямую отображает будут ли сказываться данные производимые действия на общее состояние канала связи.

Наблюдение будет осуществляться с двух позиций: г.Москва, ближайшая сеть к ядру сети и с г.Тольятти, присоединенному путем OpenVPN соединения. Для наблюдения будет использована штатная утилита PING и программа WinMTR, производящая наблюдение путем трассировки до конечного узла, что сможет наглядно показать что переключение произошло и были связанные с ним на этот момент потери пакетов зафиксированных на рисунке 25.

Рисунок 25 - График времени ответа на ICMP запрос во время переключения.

Анализируя представленный материал, можно сделать вывод что разработанный скрипт выполняет свое прямое предназначение и в случае аварийной ситуации сможет в автоматическом режиме произвести переключение, но в тоже время при использовании данного решения в рамках штатной оперативной работы для вывода определенного канала из эксплуатации не представляется возможным в полной мере в связи с возникающими потерями связи во время перестроения таблицы маршрутизации в четырех узлах агрегации сети, в связи с чем необходимо произвести пост-анализ ситуации, направленный на выдвижение рабочего плана по изменению структуры переключения.

Производя анализ, можно сделать вывод что данного рода проблема имеет место в связи с практически одновременным запуском процессов перестроения таблиц динамической маршрутизации на всех узлах сети, что воспрепятствует нормальному прохождению трафика в это время, поскольку устройства не имеют представления о корректном направлении прохождения трафика. В основах применяемого протокола динамической маршрутизации OSPF лежит оценка стоимости маршрута, т.е. при изменении стоимости на одном из интерфейсов, находящихся в единой backbone зоны.

В связи с чем для уменьшения влияния производимого переключения изменим логику работы блока ChangedCostOSPF, производя переключения только по отношению к Slave маршрутизатору удаленного офиса и резервному ядру сети в дата центре. Данная схема переключений представлена в наглядном виде в таблице №11.

Таблица 11 - Графическое отображение измененного плана переключений

Входной статус интерфейса	Входное значение	Итоговое значение	Итоговый статус интерфейса
Основной интерфейс - Master	150	150	Основной интерфейс - Backup
Резервный интерфейс - Backup	200	100	Резервный интерфейс - Master
Основной интерфейс - Backup	150	150	Основной интерфейс - Master
Резервный интерфейс - Master	100	200	Резервный интерфейс - Backup

Произведя смену логику работы указанного блока произведем последующее тестирование в тех же условиях. Результаты отображены на 26 рисунке.

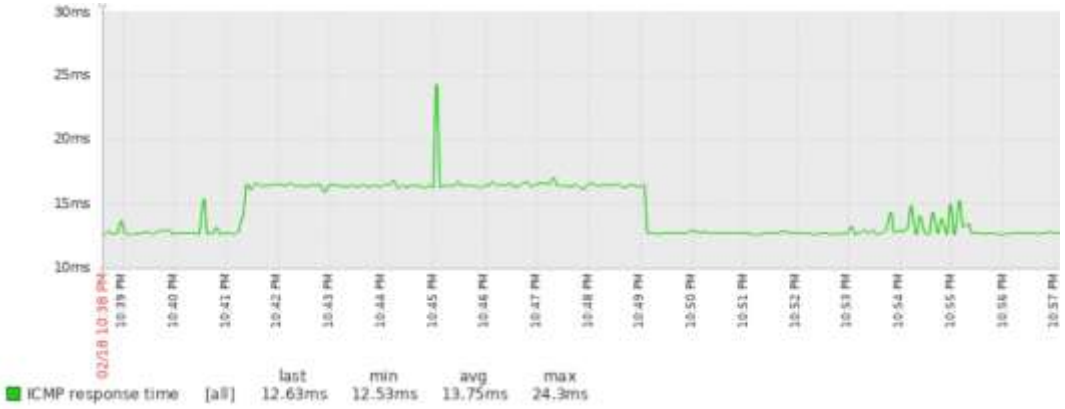

Рисунок 26 - График времени ответа на ICMP запрос во время переключения после изменения логики работы.

Произведенные изменения положительно сказались на общую картину, в момент запуска процесса перестроения маршрутизации трафик ходит штатно, без моментов связанных с потерей связи, тоже самое происходит и во время обратного перестроения. Обратимся к информации, представленной на рисунке № 27, собираемой утилитой MTR для того чтобы удостовериться о выполненном переключении и переводе трафика.

```
Keys:  Help   Display mode   Restart statistics   Order of fields   quit
                                 Packets               Pings
Host                          Loss%   Snt   Last   Avg  Best  Wrst StDev
1. 192.168.82.2                0.0%  1313    0.6   0.6   0.6   5.7   0.2
2. 10.255.200.4                0.0%  1313    0.9   0.7   0.7   5.8   0.3
   10.255.200.3
3. 172.16.253.77               0.0%  1313   16.5  12.6  12.5  22.2   0.6
   172.16.253.194
   192.168.13.253
4. 192.168.12.131              0.0%  1313   16.9  13.1  12.9  19.3   0.6
```

Рисунок 27 - Трассировка прохождения трафика.

Заключение

Выпускная квалификационная работа направлена на решение актуальной проблемы с выбором оптимального маршрута интернет соединения, позволяя автоматизировать данный процесс в случае возникновения нештатных на узлах передачи данных.

В ходе выполнения работы был решен комплекс поставленных задач:

- Изучена и описана применяемая топология сети, принципы построения и связи с внешними организациями;
- Проанализирован принцип управления, применяемые протоколы внутренней и внешней маршрутизации;
- Внедрена система мониторинга и сбора событий Zabbix;
- Разработан алгоритм внедрения автоматизации;
- Организована экспериментальная работа по внедрению автоматизированной системы управления маршрутизацией.

Получившийся программный продукт, реализованный на языке программирования Python, прошел тестирование и внедрен в систему мониторинга для интеграции автоматического режима управления и выдачи исполнительных команд на ряд сетевых устройств, участвующих в маршрутизации передаваемого трафика.

Внедрение системы позволило перейти на автоматизированный принцип управления каналами передачи данных для проведения оперативного анализа сетевой доступности ресурсов и последующего выбора оптимального маршрута и снижения простоя ресурсов из-за некорректной работы каналов связи.

Подводя итог, можно сделать вывод, что все поставленные задачи были достигнуты, а результатом работы является программное решение, программный код которого представлен в приложении А.

Список используемой литературы

1. Гольштейн Б. С., Пинчук А. В., Суховицкий А. Л. IP-телефония (третье издание). М.: Радио и связь, 2006. – 336 с.: ил. ISBN: 5-256-01585-0
2. Госкаров Д. В., Интеллектуальные информационные системы: учебное пособие / Д. В. Госкаров – Москва: 2009. – 805 с
3. Далле Вакке, А. Zabbix. Практическое руководство / Андреа Далле Вакке ; пер. с англ. А.Н. Киселева. - Москва : ДМК Пресс, 2017. - 356 с. - ISBN 978-5-97060-462-5. - Текст : электронный.
4. Косарев, В. А. Локальные вычислительные сети : учебное пособие / В. А. Косарев, А. А. Игнаткин. - Москва : ИД МИСиС, 2018. - 149 с. - Текст : электронный.
5. Коцюба И.Ю., Чунаев А.В., Шиков А.Н. Основы проектирования информационных систем. Учебное пособие. – СПб: Университет ИТМО, 2015. – 206 с.
6. Нотации моделирования бизнес-процессов [Электронный ресурс]. – Режим доступа: https://www.businessstudio.ru/products/business_studio/ (дата обращения: 05.02.2022)
7. Олифер В.Г. Компьютерные сети. Принципы, технологии, протоколы - Учебник для ВУЗов, 2-е издание/ В.Г. Олифер, Н. А Олифер. СПб.: Питер, 2004 - 864с., ил.
8. Официальный сайт «Mikrotik» [Электронный ресурс]. – Режим доступа: https://mikrotik.com/
9. Официальный сайт «Zabbix» [Электронный ресурс]. – Режим доступа: https://www.zabbix.com/documentation/current/ru/manual
10. Ретана А. Принципы проектирования корпоративных IP сетей [Текст] / А. Ретана, Д. Слайс, Р. Уайт, пер. с анг. – М.: «Вильяс», 2012. –368 с.
11. Семенов А.Б. Проектирование и расчет структурированных кабельных систем и их компонентов [Текст] / А.Б. Семенов и др. – М.: ДМК Пресс, 2014. – 416 с.

12. Asterisk [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Asterisk (дата обращения: 11.03.2022).
13. OSPF [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/OSPF (дата обращения: 11.03.2022)
14. OSPF Mikrotik [Электронный ресурс]. – Режим доступа: https://itproffi.ru/ospf-mikrotik-polnaya-instruktsiya-po-nastrojki-ospf-na-mikrotik/ (дата обращения: 14.03.2022)
15. RouterOS-api [Электронный ресурс]. – Режим доступа: https://pypi.org/project/RouterOS-api/ (дата обращения: 14.03.2022)
16. Agarwal B.B., Gupta M., Tayal S.P. Software Engineering And Testing: An Introduction – Jones & Bartlett Publishers, 2009. – 515 p. – ISBN-10 1934015555; ISBN-13 978-1934015551.
17. Alpaev Gennadiy. Software Testing Automation Tips – Apress, 2017. – 50 p.
18. Andrea Dalle Vacche, Masrering Zabbix, second edition, 2019. -278 p. – ISBN – 9781785289262.
19. Davidson J., Peters J., Bhatia M. Voice over IP fundamentals. – Cisco press, 2006.
20. Nathan Liefting, Brian van Baekel. Zabbix 5 IT Infrastructure Monitoring Cookbook: Explore the New Features of Zabbix 5 for Designing, Building, and Maintaining Your Zabbix Setup, 2021. – 346 p. – ISBN- 1800202237, 9781800202238.
21. Nathan Liefting, Zabbix 6 IT Infrastructure Monitoring Cookbook: Explore the new features of Zabbix 6 for designing, building, and maintaining your Zabbix setup, 2nd Edition - Simon and Schuster, 2022. – 528 pages. - ISBN-13: 978-180324691, ISBN-10: 180324691X.

Приложение А

Фрагмент программного кода

```
import routeros_api
import time
import datetime
import sys
import configparser
import requests

Test_Run = True

config = configparser.ConfigParser()
config.read(fr'{sys.path[0]}\config.ini', encoding="utf-8")
TimeOfLastStart = datetime.datetime.strptime(config['OSPF_Robot']['TimeOfLastStart'] ,  '%Y-%m-%d %H:%M:%S.%f')
TimeBlockForNext = int(config['OSPF_Robot']['TimeBlockForNext']) * 60   # Время в минутах на блокировку следующего запуска
TlgAlertList = [79000] # Массив сотрудников, получающих уведомления

Tlg_Token = "ID628:AAFU_p7token"
TlgAlertMess = [] # Массив текущего уведомления
Alert_CC = str(sys.argv[1]) # Город - инициатор
#print(Alert_CC)

GW_Master_Plaint = False
GW_Slave_Plaint = False
GW_Master_IP=""
GW_Slave_IP = ""
API_TUN1=""
API_TUN2=""

RoutingOSPFInterface_DC = [] # Перечень всех нужных интерфейсов OSPF в ДЦ
OSPF_Interface_At_DC = [] # Нахождение запрашиваемого интерфейса в ДЦ

API_login = "admin" # API Mikrotik КЦ
API_Pass = "pass" # API Mikrotik КЦ
API_TUN_login = "admin" # API TUN Mikrotik КЦ
API_TUN_Pass = "pass" # API TUN Mikrotik КЦ
TUN1_MP4_IP = "192.168.40.121"
TUN1_MP4_login = "admin" # API Login TUN_1 Mikrotik ДЦ
TUN1_MP4_Pass = "pass" # API Pass TUN_1 Mikrotik ДЦ
TUN2_M9_IP = "192.168.40.122"
file = "OSPF_Robot_"+datetime.datetime.today().strftime("%Y%m%d")+".txt"
log_file = open(file, "a")
log_file.write("\n-------INIT "+str(datetime.datetime.today())+"------")

def ConfStart():
```

```
  global TimeOfLastStart
   global TlgAlertMess
   TimeNow = datetime.datetime.now()

   TimeInterval = TimeNow - TimeOfLastStart
   TimeInterval = int(TimeInterval.total_seconds())
   print(TimeInterval)
   if TimeInterval > TimeBlockForNext: # 5 min = 300 s
      #print("ConfStart - OK")
      TlgAlertMess.append("\u23E9 Разрешение переключения: есть")
      return True
   else:
      print("ConfStart - False")
      TlgAlertMess.append("❌Блокировка! Прошлый старт в "+str(TimeOfLastStart))
      return False

def Tlg_Send_Notif():
   global TlgAlertList, TlgAlertMess
   TlgAlertMess = '\n'.join(TlgAlertMess)
   #print(TlgAlertMess)

   if Test_Run == True:
      # Уведомления приходят только n.horishenko во время теста
      TlgAlertList = [799]  # Массив сотрудников, получающих уведомления ,=
   for it_sotr in range(len(TlgAlertList)):
      url = "https://api.telegram.org/bot"
      channel_id = TlgAlertList[it_sotr]
      url += Tlg_Token
      method = url + "/sendMessage"
      mode = "HTML"
      #text = text_of_notif()
      r = requests.post(method, data={
          "chat_id": channel_id,
          "text": TlgAlertMess,
          "parse_mode": "HTML",
           })
      if r.status_code == 200:
         log_file.write("\n"+str(datetime.datetime.today())+" Уведомление успешно отправлено для "+
str(channel_id))
      else:
         log_file.write("\n"+str(datetime.datetime.today())+" Error send to "+str(channel_id))

def text_of_notif():    global ticket_id,ticket_info,ticket_subject,ticket_date,ticket_theme
```

```
global ticket_mass,ticket_office,ticket_project
    out_text = []
    out_text.append("<b>"+ticket_subject+"</b>")
    link = """<a href=\"https:// lc.lsupport.ru /Ticket/Display.html?id="""+ticket_id+"""\">Заявка
#"""+ticket_id+""" в RT</a>"""
    out_text.append(link)
    out_text.append("Проект: " + ticket_project)
    out_text.append("Площадка: "+ticket_office)
    out_text.append("Массовый инцендент: " + ticket_mass)
    out_text.append(ticket_info)
    #out_text += "\n" + ticket_subject
    #out_text += "\n <a href='https://lc.lsupport.ru/Ticket/Display.html?id='+ticket_id+'>Заявка в RT</a>"
    out_text = '\n'.join(out_text)
    print(out_text)
    return out_text


def ErrorUpdate():
    print("Error while update")

def ChangedCostOSPF(gw, id, interface, cost):
    global GW_Master_IP, GW_Slave_IP, API_login, API_Pass, GW_Master_Plaint, GW_Slave_Plaint
    global API_TUN1, API_TUN2
    global config
    print("GW_Slave_IP = "+str(GW_Slave_IP))
    #print("API_TUN2 = " + str(API_TUN2))
    config['OSPF_Robot']['TimeOfLastStart'] = str(datetime.datetime.now())
    _isSending = True
    countSend = 0
    while _isSending:
        if countSend > 3:
            print("Потеряна связь с Mikrotik... "+ str(gw))
            TlgAlertMess.append("Потеряна связь с "+ str(gw))
            _isSending = False
        else:
            try:
                print("Связь с Mikrotik..."+ str(gw))

                if gw == "master":
                    GW_Master = routeros_api.RouterOsApiPool(GW_Master_IP, username=API_login,
password=API_Pass, plaintext_login=GW_Master_Plaint)
                    API_Master = GW_Master.get_api()
                    interfaces = API_Master.get_resource('/routing/ospf/interface')
                    interfaces.set(id=id, cost=str(cost))
                if gw == "slave":
                    GW_Slave = routeros_api.RouterOsApiPool(GW_Slave_IP, username=API_login,
password=API_Pass, plaintext_login=GW_Slave_Plaint)
                    API_Slave = GW_Slave.get_api()
                    interfaces = API_Slave.get_resource('/routing/ospf/interface')
                    interfaces.set(id=id, cost=str(cost))
```

```
            if gw == "MP4-TUN1":
                interfaces = API_TUN1.get_resource('/routing/ospf/interface')
                interfaces.set(id=id, cost=str(cost))

            if gw == "M9-TUN2":
                interfaces = API_TUN2.get_resource('/routing/ospf/interface')
                interfaces.set(id=id, cost=str(cost))

            print("ChangedCostOSPF on ", gw, "at interface: ", interface, "by cost = ", cost)
            TlgAlertMess.append("Интерфейс " + str(interface) + " на " + str(gw) + " cost = " + str(cost))
            _isSending = False
        except Exception:
            print("Попытка связи с "+str(gw)+" неуспешна")
            TlgAlertMess.append("Попытка связи с "+str(gw)+" неуспешна")
            time.sleep(5)
            countSend += 1


def AnalyzeCostOSPF():
    global OSPF_Master_ID, OSPF_Backup_ID, OSPF_Master_Cost, OSPF_Backup_Cost
    global OSPF_Interface_At_DC
    # M  150   ->   150     ->  200
    # S  200   ->   149     ->  150
    #print("AnalyzeCostOSPF")
    #print(OSPF_Backup_ID[0]['gw'])
    if (OSPF_Backup_Cost > 9989 or OSPF_Master_Cost > 9999):
        TlgAlertMess.append("<b>В КЦ нет или есть только один OSPF интерфейс</b>")
    else:
        if (Interface_At_DC(interface=OSPF_Backup_ID[0]['interface']) and
Interface_At_DC(interface=OSPF_Master_ID[0]['interface']) and Test_Run == False):

            if OSPF_Master_Cost == 150:
                # Сейчас маршрутизация идет через ОСНОВНОЙ туннель
                # Понижаем значение Cost на РЕЗЕРВНОМ туннеле до 100
                ChangedCostOSPF(gw=OSPF_Backup_ID[0]['gw'], id=OSPF_Backup_ID[0]['id'],
interface=OSPF_Backup_ID[0]['interface'], cost=100)
                Interface_At_DC(interface=OSPF_Backup_ID[0]['interface'])
                ChangedCostOSPF(gw=OSPF_Interface_At_DC[0]['gw'], id=OSPF_Interface_At_DC[0]['id'],
interface=OSPF_Interface_At_DC[0]['interface'], cost=100)

            if OSPF_Master_Cost == 100:
                # Сейчас маршруты через РЕЗЕРВНЫЙ туннель
                # Поэтому повышаем значение именно на РЕЗЕРВНОМ туннеле на 200
                ChangedCostOSPF(gw=OSPF_Master_ID[0]['gw'], id=OSPF_Master_ID[0]['id'],
interface=OSPF_Master_ID[0]['interface'], cost=200)
                Interface_At_DC(interface=OSPF_Master_ID[0]['interface'])
                ChangedCostOSPF(gw=OSPF_Interface_At_DC[0]['gw'], id=OSPF_Interface_At_DC[0]['id'],
```

```
interface=OSPF_Interface_At_DC[0]['interface'], cost=200)

        """
        Блок переключения до модернизации
        ChangedCostOSPF(gw=OSPF_Backup_ID[0]['gw'], id=OSPF_Backup_ID[0]['id'],
interface=OSPF_Backup_ID[0]['interface'], cost=OSPF_Master_Cost-1)
        Interface_At_DC(interface=OSPF_Backup_ID[0]['interface'])
        ChangedCostOSPF(gw=OSPF_Interface_At_DC[0]['gw'], id=OSPF_Interface_At_DC[0]['id'],
interface=OSPF_Interface_At_DC[0]['interface'], cost=OSPF_Master_Cost-1)

        ChangedCostOSPF(gw=OSPF_Master_ID[0]['gw'], id=OSPF_Master_ID[0]['id'],
interface=OSPF_Master_ID[0]['interface'], cost=200)
        Interface_At_DC(interface=OSPF_Master_ID[0]['interface'])
        ChangedCostOSPF(gw=OSPF_Interface_At_DC[0]['gw'], id=OSPF_Interface_At_DC[0]['id'],
interface=OSPF_Interface_At_DC[0]['interface'], cost=200)

        #time.sleep(5)
        ChangedCostOSPF(gw=OSPF_Backup_ID[0]['gw'], id=OSPF_Backup_ID[0]['id'],
interface=OSPF_Backup_ID[0]['interface'], cost=150)
        Interface_At_DC(interface=OSPF_Backup_ID[0]['interface'])
        ChangedCostOSPF(gw=OSPF_Interface_At_DC[0]['gw'], id=OSPF_Interface_At_DC[0]['id'],
interface=OSPF_Interface_At_DC[0]['interface'], cost=150)
        """



        TlgAlertMess.append("\u2705<b>Перестройка OSPF выполнена успешно</b>")
        TlgAlertMess.append("Актуальная схема OSPF: ")
        ConnectingToCC(place=Alert_CC)
    else:
        TlgAlertMess.append("\u274C<b>Ошибка перестроения OSPF! </b>")

def Interface_At_DC(interface):
  global  RoutingOSPFInterface_DC
  global OSPF_Interface_At_DC
  OSPF_Interface_At_DC = [] # Нахождение запрашиваемого интерфейса в ДЦ
  Result_Search = False
  for Interface in RoutingOSPFInterface_DC:
    #print(Interface['interface'])
    if Interface['interface'] == interface:
      #print("This interface at DC: ", str(Interface['gw']), " ID: ", str(Interface['id']), " with cost: ",
str(Interface['cost']))
      OSPF_Interface_At_DC.append({'gw': Interface['gw'],'id': Interface['id'],'interface':
Interface['interface'],'cost': Interface['cost']})
      Result_Search = True
  if Result_Search == False:
    TlgAlertMess.append("\u274C<b>Ошибка сопоставления КЦ-ДЦ: </b>"+str(interface))
  return Result_Search


def GetOSPF_DC():  # Получение списка всех туннелей в ДЦ
```

```
global RoutingOSPFInterface_DC
   global TUN1_MP4_IP, TUN2_M9_IP, API_TUN_login, API_TUN_Pass, TUN1_MP4_login,
TUN1_MP4_Pass
   global API_TUN1, API_TUN2
   #         MP4:    tun - 1         |     M9:   tun - 2

   try:
      TUN1_MP4 = routeros_api.RouterOsApiPool(TUN1_MP4_IP, username=TUN1_MP4_login,
password=TUN1_MP4_Pass)
      API_TUN1 = TUN1_MP4.get_api()
      print("API_TUN1 Success here - 1")
   except Exception:
      print("API_TUN1 Crashed here - 1 except")

      try:
         TUN1_MP4 = routeros_api.RouterOsApiPool(TUN1_MP4_IP, username=TUN1_MP4_login,
password=TUN1_MP4_Pass,plaintext_login=True)
         API_TUN1 = TUN1_MP4.get_api()
         print("API_TUN1 Success here - 2")
      except Exception:
         TlgAlertMess.append("<b>API Error: </b> TUN1_MP4")
         print("Cannot connect to TUN1_MP4")

   #time.sleep(1)

   if DC_One_Mikrotik == False:
      # На случай аггрегации в ином, нестандартном узле без второго плеча
      try:
         TUN2_M9 = routeros_api.RouterOsApiPool(TUN2_M9_IP, username=API_TUN_login,
password=API_TUN_Pass)
         API_TUN2 = TUN2_M9.get_api()
         #print("API_TUN2 Success here - 1")
      except Exception:
         #print("API_TUN2 Crashed here - 1 except")
         try:
            TUN2_M9 = routeros_api.RouterOsApiPool(TUN2_M9_IP, username=API_TUN_login,
password=API_TUN_Pass, plaintext_login=True)
            API_TUN2 = TUN2_M9.get_api()
            #print("API_TUN2 Success here - 2")
         except Exception:
            TlgAlertMess.append("<b>API Error: </b> TUN2_M9")
            print("Cannot connect to TUN2_M9")

   RoutingOSPFInterface_MP4_RAW = API_TUN1.get_resource('/routing/ospf/interface').get()
   #time.sleep(1)

   if DC_One_Mikrotik == False:
      RoutingOSPFInterface_M9_RAW = API_TUN2.get_resource('/routing/ospf/interface').get()

   for Interface in RoutingOSPFInterface_MP4_RAW:
      try:
```

```
if Interface['state'] == "point-to-point" and Interface['disabled'] == "false":
            RoutingOSPFInterface_DC.append({'gw': "MP4-TUN1",'id': Interface['id'], 'interface':
Interface['interface'], 'cost': int(Interface['cost'])})
        except:
            #print("Interface at MP4 - except")
            pass


    if DC_One_Mikrotik == False:
        for Interface in RoutingOSPFInterface_M9_RAW:
            try:
                if Interface['state'] == "point-to-point" and Interface['disabled'] == "false":
                    RoutingOSPFInterface_DC.append({'gw': "M9-TUN2",'id': Interface['id'], 'interface':
Interface['interface'], 'cost': int(Interface['cost'])})
            except:
                #print("Interface at M9 - except")
                pass
    #print("Len of RAW_MP4: ", str(len(RoutingOSPFInterface_MP4_RAW)))
    #print("Len of RAW_M9: ", str(len(RoutingOSPFInterface_M9_RAW)))
    print("Interfaces.Len of DC: ", str(len(RoutingOSPFInterface_DC)))

def ConnectingToCC(place):
    global OSPF_Master_Cost, OSPF_Backup_Cost, OSPF_Master_ID, OSPF_Backup_ID
    global API_Master, API_Slave, API_login, API_Pass
    global TlgAlertMess
    global GW_Slave_IP, GW_Master_IP
    global GW_Master_Plaint, GW_Slave_Plaint


    def ConnectingStart():
        global OSPF_Master_Cost, OSPF_Backup_Cost, OSPF_Master_ID, OSPF_Backup_ID
        global API_Master, API_Slave, API_login, API_Pass
        global TlgAlertMess
        global GW_Master_Plaint, GW_Slave_Plaint
        #print("Start ConnectingStart")

        try:
            GW_Master = routeros_api.RouterOsApiPool(GW_Master_IP, username=API_login,
password=API_Pass)
            GW_Master_Plaint = False

            API_Master = GW_Master.get_api()
        except:
            try:
                GW_Master = routeros_api.RouterOsApiPool(GW_Master_IP, username=API_login,
password=API_Pass, plaintext_login=True)
                GW_Master_Plaint = True
```

```
    API_Master = GW_Master.get_api()
        except:
            TlgAlertMess.append("<b>API Error: </b> GW_Master")
            print("Cannot connect to GW_Master")

        #time.sleep(1)

        if GW_Slave_IP != "0.0.0.0":
            try:
                GW_Slave = routeros_api.RouterOsApiPool(GW_Slave_IP, username=API_login,
password=API_Pass)
                GW_Slave_Plaint = False
                API_Slave = GW_Slave.get_api()
            except:
                try:
                    GW_Slave = routeros_api.RouterOsApiPool(GW_Slave_IP, username=API_login,
password=API_Pass, plaintext_login=True)
                    GW_Slave_Plaint = True
                    API_Slave = GW_Slave.get_api()
                except:
                    TlgAlertMess.append("<b>API Error: </b> GW_Slave")
                    print("Cannot connect to GW_Slave")


        try:
            RoutingOSPFInterface_Master = API_Master.get_resource('/routing/ospf/interface').get()
        except:
            TlgAlertMess.append("<b>API Error: </b> Не возможно получить список OSPF Interfaces на
Master")
            print("Не возможно получить список OSPF Interfaces на Master")

        #time.sleep(1)

        try:
            RoutingOSPFInterface_Slave = API_Slave.get_resource('/routing/ospf/interface').get()
        except:
            if GW_Slave_IP != "0.0.0.0":
                TlgAlertMess.append("<b>API Error: </b> Не возможно получить список OSPF Interfaces на
Slave")
                print("Невозможно получить список OSPF Interfaces на Slave")
            else:
                print("Невозможно получить список OSPF Interfaces на Slave, т.к. в КЦ всего 1 Mikrotik")
        #print(RoutingOSPFInterface_Master)
        #print(RoutingOSPFInterface_Slave)
            OSPF = []  # Перечень всех активных интерфейсов в КЦ
        try:
            for Interface in RoutingOSPFInterface_Master:
                try:
                    if Interface['state'] == "point-to-point" and Interface['disabled'] == "false":
                        OSPF.append({'gw': "master", 'id': Interface['id'], 'interface': Interface['interface'],
                                'cost': int(Interface['cost'])})
```

```
            except:
                # print("Interface at master - except")
                pass
        except:
            TlgAlertMess.append("<b>API Error: </b> Пустой массив OSPF Interfaces на Master")
            print("Пустой массив OSPF Interfaces на Master")
            pass


        if GW_Slave_IP != "0.0.0.0":
            for Interface in RoutingOSPFInterface_Slave:
                try:
                    if Interface['state'] == "point-to-point" and Interface['disabled'] == "false":
                        OSPF.append({'gw': "slave", 'id': Interface['id'], 'interface': Interface['interface'],
                                     'cost': int(Interface['cost'])})
                except:
                    # print("Interface at slave - except")
                    pass

        #print("Total_GW_OSPF: ", OSPF)
        OSPF_Master_Cost = 9990
        OSPF_Backup_Cost = 9999
        OSPF_Master_ID = []  # GW  | interface
        OSPF_Backup_ID = []  # GW  | interface
        for OSPF_GRE in range(len(OSPF)):
            # print(OSPF[OSPF_GRE])
            if OSPF[OSPF_GRE]['cost'] < OSPF_Master_Cost:
                # print("cost < OSPF_Master_Cost")
                # Перед этим OSPF_Backup_Cost присвоить значение OSPF_Master_Cost
                OSPF_Backup_Cost = OSPF_Master_Cost
                OSPF_Backup_ID = []
                OSPF_Backup_ID = OSPF_Master_ID
                # OSPF_Backup_ID.append({'gw': OSPF[OSPF_GRE]['gw'], 'interface':
OSPF[OSPF_GRE]['interface']})
                # Тут надо OSPF_Master_Cost присвоить значение OSPF[OSPF_GRE]['cost']
                OSPF_Master_Cost = OSPF[OSPF_GRE]['cost']
                OSPF_Master_ID = []
                OSPF_Master_ID.append({'gw': OSPF[OSPF_GRE]['gw'], 'interface':
OSPF[OSPF_GRE]['interface'], 'id': OSPF[OSPF_GRE]['id']})
            elif OSPF[OSPF_GRE]['cost'] < OSPF_Backup_Cost:
                # Это точно не Master, но ближе к Slave
                # print("Else cost < OSPF_Master_Cost")
                OSPF_Backup_Cost = OSPF[OSPF_GRE]['cost']
                OSPF_Backup_ID = []
                OSPF_Backup_ID.append({'gw': OSPF[OSPF_GRE]['gw'], 'interface':
OSPF[OSPF_GRE]['interface'], 'id': OSPF[OSPF_GRE]['id']})

        print("OSPF_Master: ", OSPF_Master_ID)
        print("OSPF_Master: ", OSPF_Master_Cost)
```

```
#Interface_At_DC(interface=str(OSPF_Master_ID[0]['interface']))

        print("OSPF_Backup: ", OSPF_Backup_ID)
        print("OSPF_Backup: ", OSPF_Backup_Cost)
        #Interface_At_DC(interface=str(OSPF_Backup_ID[0]['interface']))

        try:
            TlgAlertMess.append("CC_M: "+str(OSPF_Master_ID[0]['interface'])+" at
"+str(OSPF_Master_ID[0]['gw'])+" "+str(OSPF_Master_Cost))
            TlgAlertMess.append("CC_B: "+str(OSPF_Backup_ID[0]['interface'])+" at
"+str(OSPF_Backup_ID[0]['gw'])+" "+str(OSPF_Backup_Cost))
        except:
            TlgAlertMess.append("<b>Error: </b> Пустой массив OSPF Interfaces")
            print("Пустой массив OSPF Interfaces")
            pass


    if place == "Nkamsk":
        GW_Master_IP="192.168.40.253"
        GW_Slave_IP = "192.168.40.252"
        ConnectingStart()
        return True
    else:
        print("Error of place name")
        return False
if __name__ == '__main__':
    TlgAlertMess.append("□<b>OSPF </b>"+str(Alert_CC))
    TlgAlertMess.append("🕑"+str(datetime.datetime.today().strftime("%Y-%m-%d-%H.%M.%S")))

    if Alert_CC == "NNovg":
        # Блок инициализации Нижнего Новгорода в связи с нестандартной схемой включения
        TUN1_MP4_IP="192.168.41.2"
        TUN1_MP4_login="admin"
        TUN1_MP4_Pass="pass"
        DC_One_Mikrotik = True
    else:
        DC_One_Mikrotik = False
    GetOSPF_DC()
    ConfStart()
    print("TimeOfLastStart ",TimeOfLastStart)

    if ConnectingToCC(place=Alert_CC) == True:
        #print("Before start AnalyzeCostOSPF")
        AnalyzeCostOSPF()
    Tlg_Send_Notif()
    #GW_Master.disconnect()
    #GW_Slave.disconnect()
    with open(fr'{sys.path[0]}\config.ini', 'w') as configfile: # Применение изменений в config
        config.write(configfile)
```