Описание архитектуры

Система управления Линк

Содержание

1. Введение **3**
1.1. Описание документа 3
1.2. Термины 4
2. Общие сведения **5**
2.1. Общие сведения и функции 5
3. Модули системы Линк **6**
3.1. Модуль Front 7
3.2. Модуль Backbone 7
3.3. Модуль Roller 8
3.4. Модуль Commander 8
3.5. Модуль MongoDB 8
3.6. Модуль RabbitMQ 9
3.7. Модуль Redis 9
3.8. Модуль LMS 10
4. Детали реализации **11**
4.1. Организация взаимодействия между модулями 11
4.2. Рекомендации по организации виртуализации 12
4.3. **Docker** 12
4.4. **Docker Swarm** 12
4.5. **RabbitMQ** 13
4.6. **MongoDB** 13
5. Развертывание системы Линк **14**
5.1. Установка Линк 14
5.2. Системные требования 14
5.3. Требования к уровню подготовки персонала 15
6. Масштабирование и сценарии развертывания **16**
6.1. Масштабирование 16
6.2. Сценарии развертывания 16
6.2.1. Минимальный / Базовый-сценарий (для небольших развертываний) 17
6.2.2. Расширенный-сценарий (для больших развертываний) 19
6.2.3. Максимальный-сценарий (для больших развертываний) 21
7. Отказоустойчивость **25**
8. Дополнительные возможности **25**
9. Заключение **26**

1. Введение

1.1. Описание документа

Данный документ содержит подробную информацию о Системе управления Линк, ее архитектуре, особенностях реализации и вариантах развертывания.
В документе приведены сведения о назначении, составе и основных функциях Системы управления Линк (далее система Линк, Линк). Рассмотрены модули системы Линк, их назначение, задачи и организация взаимодействия между ними.
Описаны используемые для реализации технологии быстрого развертывания и масштабирования.

1.2. Термины

Docker — программное решение Docker, позволяющее запустить Линк на любом компьютере.

Docker-container — образ-контейнер с собственной виртуальной машиной для запуска программы в Docker. Внутри Docker-контейнера лежит Docker-image.

Docker-image — образ-компонент программы для интеграции в Docker. Имеет тег-версию.

Docker-node — сервер, виртуальная машина, из кластера предоставляющий доступ к своим вычислительным мощностям. На таком сервере могут быть запущены Docker-контейнеры.

Docker-service — служба, совокупность множества одинаковых инкапсулированных Docker-контейнеров, запущенных на один или нескольких Docker-node.

Docker-stack — совокупность Docker-служб, логически объединенных в единую виртуальную сеть.

Docker-repository — репозиторий программного решения.

JSON (JavaScript Object Notation) — стандартный текстовый формат для хранения и передачи структурированных данных.

Виртуальная машина (VM) — программная система, эмулирующая аппаратное обеспечение компьютера и исполняющая программы для guest-платформы (guest — гостевая платформа) на host-платформе, виртуализирующая некоторую платформу и создающая на ней среды, изолированные друг от друга программы и даже операционные системы.

Домен — индивидуальный кластер для организации, внутри которого производится взаимодействие с устройствами, принадлежащими этой организации. Для домена назначается ответственное лицо — администратор домена.

Роутер — роутер iRZ подключается к серверу сбора данных по протоколу Zelda (JSON-format) и передает ему информацию о своем состоянии.

2. Общие сведения

2.1. Общие сведения и функции

Система Линк предназначена для централизованного управления конфигурациями роутеров iRZ и позволяет контролировать состояние сетевого оборудования и инфраструктуры предприятия.

Основные функции системы:

- Контроль состояния группы устройств или каждого роутера iRZ по-отдельности. Пользователю доступна следующая информация: состояние устройства (в сети/не в сети), время последнего ответа, время работы, данные о CPU, LAN, Loopback, маршрутизации и др.
- Централизованное обновление встроенного ПО отдельных роутеров или групп устройств одной аппаратной серии.
- Отправка роутерам команд на перезагрузку и сброс настроек. Встроенный планировщик позволяет задать интервал выполнения команд.
- Создание правил формирования резервных копий устройств и системных журналов.
- Управление конфигурациями роутеров, подключенных к системе.

Все действия в системе производятся удалённо через web-интерфейс из браузера. Сервис доступен по адресу irzlink.irz.net. Клиентская часть уже встроена в ПО роутеров.

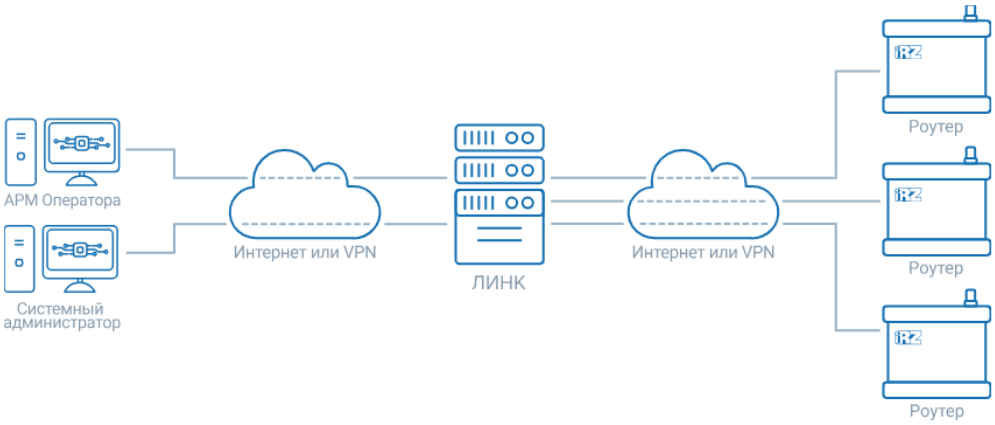


Рис. 1. Принцип организации системы Линк

Благодаря модульной архитектуре функционал системы Линк может быть адаптирован под конкретные бизнес-цели.

Линк построен на основе восьми базовых модулей. Для обеспечения кроссплатформенности и удобства эксплуатации используется платформа контейнеризации — Docker. Каждый модуль работает в отдельном Docker контейнере. Взаимодействие модулей друг с другом осуществляется через виртуальную сеть Docker, что обеспечивает высокую производительность и безопасность.

3. Модули системы Линк

В систему Линк входят следующие модули:

- **Front** — web-интерфейс программного решения;
- **Backbone** — модуль обработки запросов пользователей;
- **Roller** — модуль сбора данных;
- **Redis** — кэш-сервер;
- **RabbitMQ** — программный брокер сообщений;
- **Commander** — модуль обработки регламентных задач, уведомлений и других фоновых операций;
- **MongoDB** — нереляционная СУБД, центральное хранилище данных для Линк;
- **LMS** — модуль управления лицензиями.

3.1. Модуль Front

Модуль Front- web-сервер, отвечающий за визуальное представление актуальной информации о состоянии устройств. Front обеспечивает интерактивность и удобство использования системы, представляя функциональные возможности в доступной и понятной форме для конечных пользователей. Также модуль осуществляет контроль за доступом пользователей.

Задачи модуля:

- отображение интерактивных элементов (кнопки, таблицы, диаграммы и т.д.) и обеспечение удобного и интуитивно понятного пользовательского интерфейса;
- управление входом пользователей, сессиями и хранением cookie - гарантируя безопасность и конфиденциальность данных;
- получение данных от модуля Backbone и отображение их пользователю, предоставление актуальной информацию о состоянии устройств и системы в целом.

Порт: 80, 443.

Протокол: TCP.

Стек: модуль реализован с использованием JavaScript (JS) и фреймворка Vue.js. Vue.js обеспечивает высокую производительность и отзывчивость веб-интерфейса, делая его комфортным для пользователей.

3.2. Модуль Backbone

Модуль Backbone выступает в роли сервера бэкенда. Он обрабатывает все входящие запросы от модуля Front, инициированные действиями пользователей, выполняет необходимые действия и возвращает результат.

Задачи модуля:

- обработка пользовательских запросов, получаемых от модуля Front в соответствии с заданными правилами;
- обеспечение выполнения действий пользователей, таких как добавление устройств, изменение конфигурации, получение команд и т.д.;
- обновление панели управления новой информацией (отображение актуальных данных о состоянии системы);
- взаимодействие с модулем RabbitMQ, путем направления уведомления о событиях, требующих реакции системы;
- взаимодействие с базой данных MongoDB, для чтения и записи данных об устройствах, конфигурации, событиях и т.д.

Протокол: TCP.

Стек: модуль реализован с использованием Python и Tornado.

3.3. Модуль Roller

Модуль Roller представляет собой сервер сбора данных от сетевых устройств.

Задачи модуля:

- установление связи с сетевыми устройствами, используя протоколы UDP и TCP, для получения данных о состоянии устройств;
- получение телеметрии устройств, включая показания датчиков, информацию о трафике, статусе подключений и т.д.;
- получение обновления конфигурации устройств от пользователей через систему;
- передача файлов между системой и устройствами (обновление прошивки, загрузка конфигурации и т.д.);
- отправка уведомлений модулю RabbitMQ о событиях, связанных с устройствами;
- отправка полученных данных в модуль Backbone.

Порт: 11000.

Протоколы: TCP и UDP.

Стек: модуль реализован с использованием GO.

3.4. Модуль Commander

Модуль Commander — модуль, отвечающий за выполнение фоновых задач, мониторинг событий в системе и отправку уведомлений пользователям.

Задачи модуля:

- планирование и выполнение регламентных задач, таких как сбор логов, создание резервных копий и т.д.;
- мониторинг модуля RabbitMQ на предмет важных событий (например, изменение состояния устройства, превышение лимитов трафика, ошибки системы), и запуск выполнения соответствующих действий;
- удаление устаревшей информации из базы данных для поддержания ее актуальности;
- отправка уведомлений пользователям (по email или через систему оповещений Telegram, Discord) о важных событиях, уведомляя их о проблемах и требуя внимания.

Протокол: TCP.

Стек: модуль реализован с использованием Python и Tornado.

3.5. Модуль MongoDB

Модуль MongoDB обеспечивает функцию центрального хранилища данных. MongoDB — это документно-ориентированная система управления базами данных (СУБД), которая хранит структурированные данные в виде JSON-документов. MongoDB обеспечивает высокую производительность, масштабируемость и отказоустойчивость.

Задачи модуля:

- хранение информации об устройствах, конфигурациях, событиях, пользователях и других данных в JSON-документах, обеспечивая гибкость и эффективность хранения данных;
- обеспечение отказоустойчивости и высокой доступность информации даже в случае сбоя одного из серверов (используется репликация данных на нескольких серверах);
- создание индексов для ускорения поиска и обработки данных для повышения производительности системы.

Порт: 27017.

Протокол: TCP.

3.6. Модуль RabbitMQ

Модуль RabbitMQ — программный брокер сообщений, который используется в распределённых системах для балансировки нагрузки и улучшения обработки асинхронных задач, а также для обеспечения надёжности доставки данных.

Задачи модуля:

- распределение нагрузки: RabbitMQ позволяет распределять задачи между обработчиками, обеспечивая балансировку нагрузки;
- декаплинг компонентов: сервисы могут работать независимо, взаимодействуя через RabbitMQ, что уменьшает их взаимозависимость;
- асинхронность: RabbitMQ позволяет сервисам обрабатывать запросы асинхронно, что способствует более эффективному распределению ресурсов.

Порт: 5672.

3.7. Модуль Redis

Модуль Redis — хранилище данных с открытым исходным кодом, используемое в качестве кэш-сервера (вспомогательной базы данных), который работает в паре с системами управления БД, например, MongoDB.

Задачи модуля:

- кэширование данных, позволяющее существенно снижать нагрузки на СУБД;
- обеспечение быстрого доступа к хранимым данным, что позволяет решить проблему быстрой загрузки мелких, часто обновляемых объектов.

Порт: 6379.

3.8. Модуль LMS

Модуль LMS (License Management System) — это специализированное решение для управления лицензиями, предназначенное осуществлять активацию лицензий и их обновление.

Задачи модуля:

- проверка валидности лицензии (имя сервера, срок действия, количество устройств);
- отключение устройств, подключенных сверх лимита лицензии;
- хранение сетевых ключей;
- раздача лицензий;
- контроль количества подключений к сетевому ключу с лицензией.

Порт: 3189.

4. Детали реализации

4.1. Организация взаимодействия между модулями

На рисунке приведена общая схема взаимодействия между модулями.

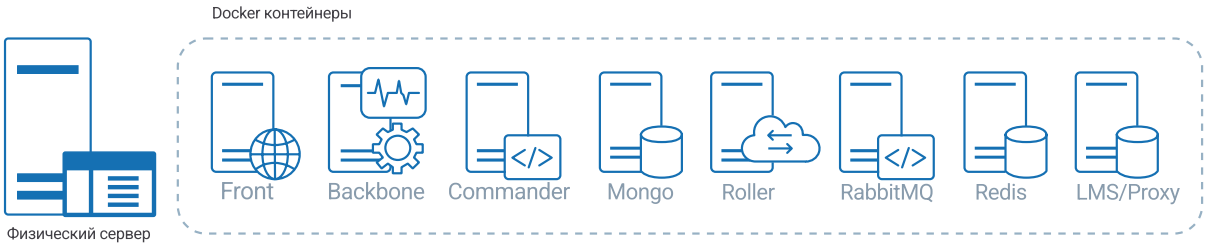


Рис. 2. Схема взаимодействия между модулями

Для обеспечения защиты от несанкционированного доступа и стабильной работы ПО в системе Линк используются изолированные VLAN.
Внешний VLAN используется для доступа к сетям, через которые производится взаимодействие пользователей с модулем Front, роутеров с модулем Roller, а также рассылка уведомлений модулем Commander. Изолированный VLAN используется для коммуникации между модулями Линк. Использование внутренний VLAN позволяет предотвратить несанкционированный доступ к модулям и повышает уровень безопасности межмодульного взаимодействия.

Модуль Front информирует модуль Backbone о действиях, произведенных пользователем.

Модуль Backbone обрабатывает пользовательские запросы от модуля Front и направляет сведения о них модулям MongoDB и RabbitMQ.

Модуль Roller осуществляет взаимодействие с устройствами по протоколу TCP (для обновления прошивки) и протоколу UDP (для передачи информации о состоянии устройств). Информацию об изменении состояния устройств Roller передает в MongoDB и модулю RabbitMQ, который передает ее Backbone для дальнейшей трансляции на web-интерфейс пользователя.

Модуль Redis обрабатывает запросы от модуля Roller, проверяя наличие запрашиваемых данных. Если они есть, то модулю Roller будет дан ответ из Redis, а не напрямую из основной базы данных MongoDB. Если данных нет, то Roller будет запрашивать их у MongoDB, а также сообщит эти данные Redis.

Модуль Commander отвечает за выполнение фоновых задач и мониторинг событий в системе. Модуль анализирует события. В случае если о событии необходимо проинформировать, модуль обращается с запросом к MongoDB. От MongoDB модуль Commander получает информацию о списке адресов email или аккаунтов Telegram для рассылки уведомлений.

4.2. Рекомендации по организации виртуализации

Каждый сервер с системой Линк может быть виртуализирован с помощью технологий виртуализации, таких как VMware или Hyper-V, что позволяет гибко распределять ресурсы и управлять серверами.

Каждая виртуальная машина, на которой установлены модули Линк, должна иметь как минимум два сетевых адаптера:

- один адаптер для доступа к публичной сети, через который пользователи подключаются к модулю Front, устройства подключаются к модулю Roller, а модуль Commander рассылает уведомления;
- второй адаптер для внутренней коммуникации между модулями Линк.

Использование VLAN также позволяет гибко управлять сетевыми ресурсами и распределять трафик между различными сегментами сети.

4.3. Docker

Docker — это платформа контейнеризации, которая позволяет упаковывать приложения и их зависимости в изолированные контейнеры. Такой подход значительно повышает удобство развертывания, управления и масштабирования приложений.

Развертывание и эксплуатация системы Линк на платформе Docker обеспечивает:

- изоляцию: контейнеры изолируют приложения друг от друга, что исключает конфликты зависимостей;
- переносимость: контейнеры можно запускать на любой платформе, где установлен Docker, что делает систему независимой от инфраструктуры заказчика;
- удобство развертывания: Docker позволяет легко развертывать и обновлять приложения, что упрощает обслуживание системы;
- образы: Docker использует образы (images) для создания контейнеров. образы содержат все необходимые файлы и конфигурации для запуска приложения;
- удобство управления ресурсами: Docker позволяет управлять ресурсами, которые доступны контейнерам, например, памятью, CPU и дисковым пространством;
- сеть: Docker обеспечивает виртуальную сеть для контейнеров, что позволяет им общаться друг с другом и с внешними системами.

4.4. Docker Swarm

Docker Swarm — представляет собой средство для кластеризации и оркестрации контейнеров. Docker Swarm позволяет управлять сгруппированными в кластер ресурсами (серверами Docker и т.д.) как единой системой.

Docker Swarm обеспечивает:

- кластеризацию: создает кластеры из нескольких серверов с Docker, что позволяет масштабировать приложения и обеспечивать высокую доступность сервисов;
- оркестрацию: управляет развертыванием, масштабированием и обновлением контейнеров в кластере;
- распределение нагрузки: обеспечивает равномерное распределение нагрузки между серверами в кластере, что повышает производительность и отказоустойчивость приложений;

- управление службами: позволяет определять и управлять службами (services) в кластере, что обеспечивает автоматическое восстановление после отказов и масштабирование приложений.

4.5. RabbitMQ

RabbitMQ — программный брокер сообщений, который используется в распределённых системах для обмена сообщениями между различными компонентами системы.

RabbitMQ обеспечивает:

- надёжное хранение сообщений: сообщения сохраняются на диске во избежание их утраты при сбое системы;
- масштабируемость: возможность горизонтального масштабирования, добавление новых узлов без остановки системы;
- мониторинг и управление: встроенные инструменты для мониторинга и управления производительностью и состоянием очередей.

4.6. MongoDB

MongoDB — это документоориентированная нереляционная СУБД. В MongoDB данные хранятся в формате JSON-подобных документов. MongoDB обеспечивает:

- гибкость: позволяет хранить данные различной структуры, что удобно для работы с телеметрией от сетевого оборудования;
- непрерывную работу системы: реализует репликацию данных на нескольких серверах с помощью репликационных наборов (Replica Sets). Каждый репликационный набор состоит из одного главного сервера (Primary) и одного или нескольких вторичных серверов (Secondary). Главный сервер принимает все записи и реплицирует их на вторичные серверы. Если главный сервер выходит из строя, один из вторичных серверов становится главным, обеспечивая непрерывную работу системы;
- объем хранимых данных: легко масштабируется горизонтально, что позволяет наращивать объем данных и производительность системы;
- быстрый поиск: позволяет создавать индексы для ускорения поиска данных. Индексы позволяют быстро находить нужные документы, что увеличивает производительность системы;
- управление доступом: предоставляет гибкие механизмы управления доступом, позволяя ограничивать доступ к данным для разных пользователей и групп;
- масштабируемость: для увеличения масштабируемости MongoDB поддерживает шардинг. Шардинг разделяет данные на несколько частей (шардов), которые располагаются на разных серверах. Это повышает производительность и масштабируемость системы, позволяя обрабатывать большие объемы данных.

5. Развертывание системы Линк

5.1. Установка Линк

Система Линк — решение, построенное по принципу «клиент-сервер». Клиентская часть уже встроена в прошивку роутеров, и никаких дополнительных действий здесь не требуется. Серверная часть может быть установлена на любом сервере, входящем в клиентскую сеть, или находиться в облаке.

При использовании облачного решения система управления предоставляется как сервис (SaaS), что не требует затрат на внедрение и конфигурирование.

Развертывание коробочного решения Линк на инфраструктуре заказчика подробно описано в документе «Линк. Инструкция по развертыванию коробочного решения».

5.2. Системные требования

Системные требования к аппаратно-программным средствам для установки Линк зависят от количества контейнеров, устройств и физических серверов.

Минимальные системные требования к хостовому серверу для базовых сценариев развертывания приведены в таблице ниже. О сценариях развертывания подробнее в разделе Сценарии развертывания.

Таблица 1. Минимальные системные требования

Сценарий развертывания	Минимальный	Базовый	Расширенный	Максимальный
Кол-во устройств под управлением	до 100	до 200	до 500	более 500
Процессор	2Ghz, 2 Core	2Ghz, 2 Core	2Ghz, 4 Core	2Ghz, 8 Core
Оперативная память	8 Gb	8 Gb	8 Gb	16 Gb
Дисковое пространство	от 50 Gb	от 50 Gb	от 100 Gb	от 500 Gb
Тип дисков	HDD	HDD	SSD	SSD

Рекомендуемые системные требования к хостовому серверу для базовых сценариев развертывания приведены в таблице ниже.

Таблица 2. Рекомендуемые системные требования

Сценарий развертывания	Минимальный	Базовый	Расширенный	Максимальный
Кол-во устройств под управлением	до 100	до 200	до 500	более 500
Процессор	2Ghz, 4 Core	2Ghz, 4 Core	2Ghz, 8 Core	2Ghz, 16 Core
Оперативная память	12 Gb	12 Gb	16 Gb	32 Gb
Дисковое пространство	от 100 G	от 100 Gb	от 300 Gb	от 1 Tb
Тип дисков	SSD	SSD	SSD	SSD

Для большей гибкости модули Линк можно разделить и разместить на отдельных рабочих станциях.

5.3. Требования к уровню подготовки персонала

Для эксплуатации, установки и настройки программных решений Линк персонал эксплуатирующей организации должен обладать следующими компетенциями:

- администрирование и сопровождение Unix подобных систем;
- установки, настройки и поддержки Docker, Docker Swarm;
- базовые знания сетевых и web-технологий.

6. Масштабирование и сценарии развертывания

6.1. Масштабирование

Масштабирование Линк предусматривает физическое распределение задач на несколько серверов, логически объединённых в кластер. Для этого используется инструмент кластеризации и оркестрации контейнеров Docker Swarm.
Docker позволяет объединять в кластер группу серверов, когда их подсистемы Docker работают вместе в режиме Swarm. Созданный таким образом кластер называется Swarm. Все машины, подключенные к Swarm, называются нодами (nodes).
Swarm состоит из двух типов нод: нода-менеджер и рабочая-нода. Каждый Swarm инициализируется с ноды менеджера и все команды Docker для управления и мониторинга swarm должны выполняться с ноды-менеджера.
Менеджер также отвечает за распределение контейнеров между всеми нодами. Совокупность всех модулей в swarm носит название stack. Правила распределения и взаимодействия между контейнерами описываются в конфигурационном файле.
Для упрощения конфигурирования рекомендуется для каждой Docker-node указывать набор тэгов (labels).

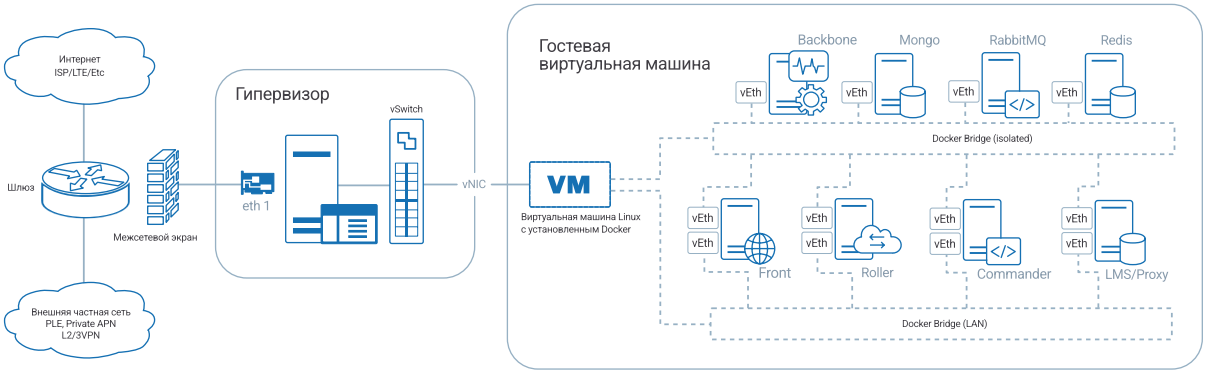
6.2. Сценарии развертывания

В зависимости от количества устройств, находящихся под управлением, и задействованного серверного оборудования, в системе Линк предусмотрено несколько основных сценария развертывания:

- Минимальный;
- Базовый;
- Расширенный;
- Максимальный.

6.2.1. Минимальный / Базовый-сценарий (для небольших развертываний)

На рисунке представлена структурная схема Линк для сценария развертывания Минимальный / Базовый. Под управлением находится до 200 устройств.



Настройка:

- все модули Линк размещаются на одном сервере.

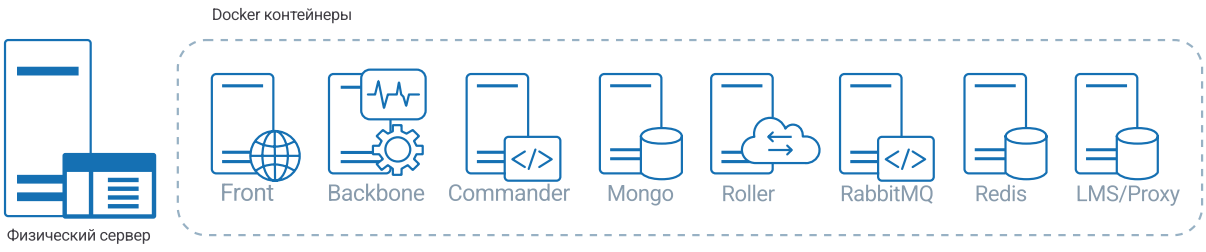
Преимущества:

- простота настройки, так как все компоненты установлены на одном сервере;
- низкая стоимость, так как требуется всего один сервер.

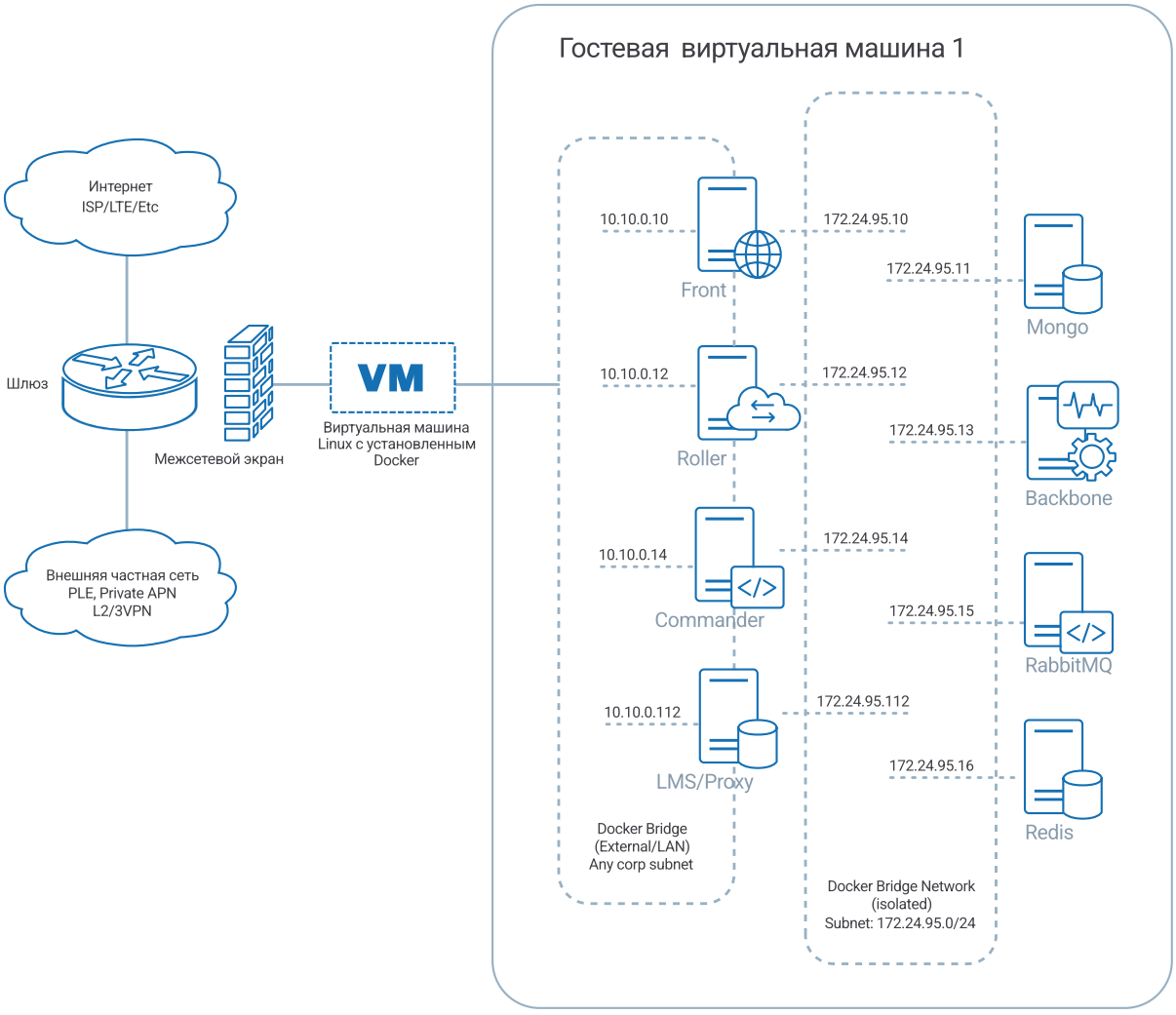
Ограничения:

- ограниченная масштабируемость, так как система ограничена мощностью одного сервера;
- сниженная отказоустойчивость, так как отказ одного сервера приведет к простою всей системы.

Схема взаимодействия между модулями для сценария развертывания Минимальный / Базовый представлена на рисунке ниже.

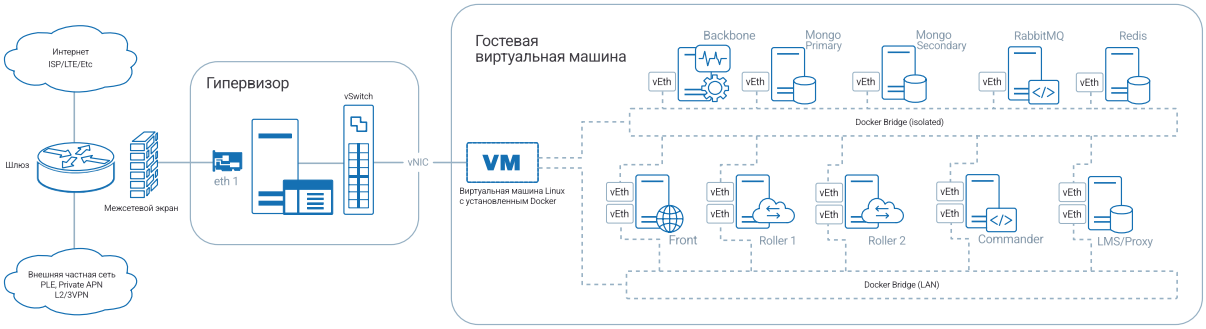


На рисунке ниже приведен пример схемы логического взаимодействия с указанием параметров настройки.


Гостевая виртуальная машина 1
Интернет
ISP/LTE/Etc
Шлюз
Межсетевой экран
VM
Виртуальная машина
Linux с установленным
Docker
Внешняя частная сеть
PLE, Private APN
L2/3VPN
10.10.0.10
Front
172.24.95.10
172.24.95.11
Mongo
10.10.0.12
Roller
172.24.95.12
172.24.95.13
Backbone
10.10.0.14
Commander
172.24.95.14
172.24.95.15
RabbitMQ
10.10.0.112
LMS/Proxy
172.24.95.112
172.24.95.16
Redis
Docker Bridge
(External/LAN)
Any corp subnet
Docker Bridge Network
(isolated)
Subnet: 172.24.95.0/24

6.2.2. Расширенный-сценарий (для больших развертываний)

На рисунке представлена структурная схема Линк для расширенного-сценария развертывания. Под управлением находится от 100 до 500 устройств.



Для данного сценария предполагается, что все модули Линк размещаются на одном сервере. При этом для критически важных компонентов (MongoDB и Roller) применяется N+1 избыточность. Роутер взаимодействуют с тем или другим экземпляром модуля Roller в зависимости от текущей нагрузки на модуль (обеспечивается балансировщиком).

Настройка:

- Все модули Линк размещаются на одном сервере;
- Использование N+1 избыточности для критически важных компонентов.

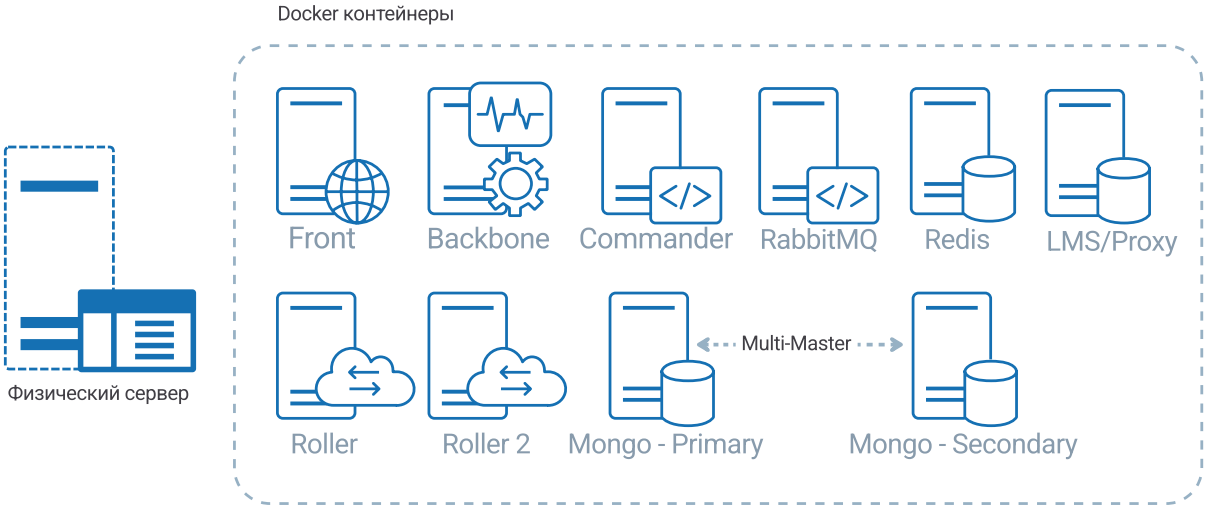
Преимущества:

- простота настройки, так как все компоненты установлены на одном сервере;
- применяется балансировка нагрузки для модуля Roller, чтобы распределить запросы между его различными экземплярами;
- используется репликация MongoDB для обеспечения доступности данных в случае сбоя в работе основной реплики MongoDB;
- низкая стоимость, так как требуется всего один сервер.

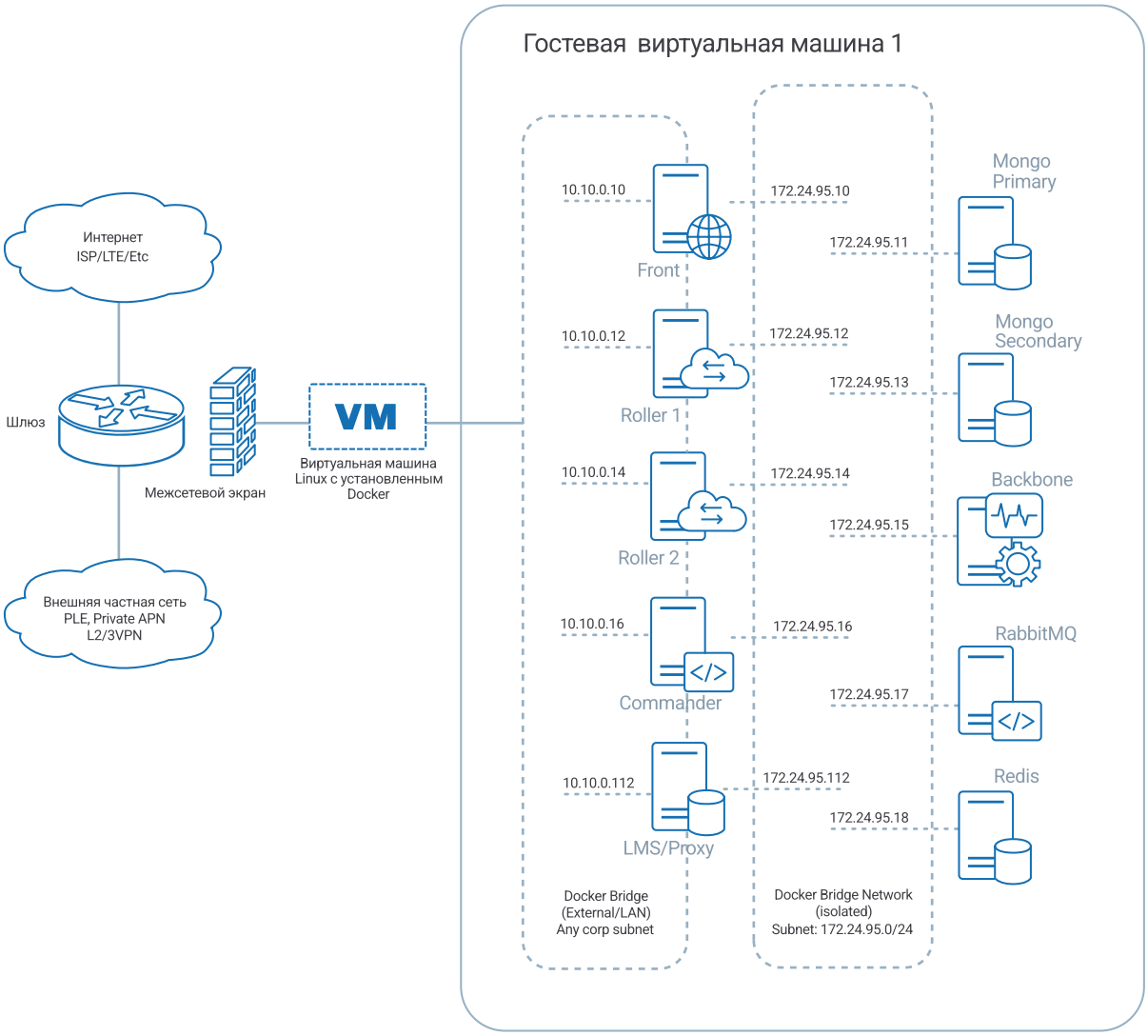
Ограничения:

- ограниченная масштабируемость, так как система ограничена мощностью одного сервера;
- сниженная отказоустойчивость, так как отказ одного сервера приведет к простою всей системы.

Схема взаимодействия между модулями для Расширенного-сценария развертывания представлена на рисунке ниже.



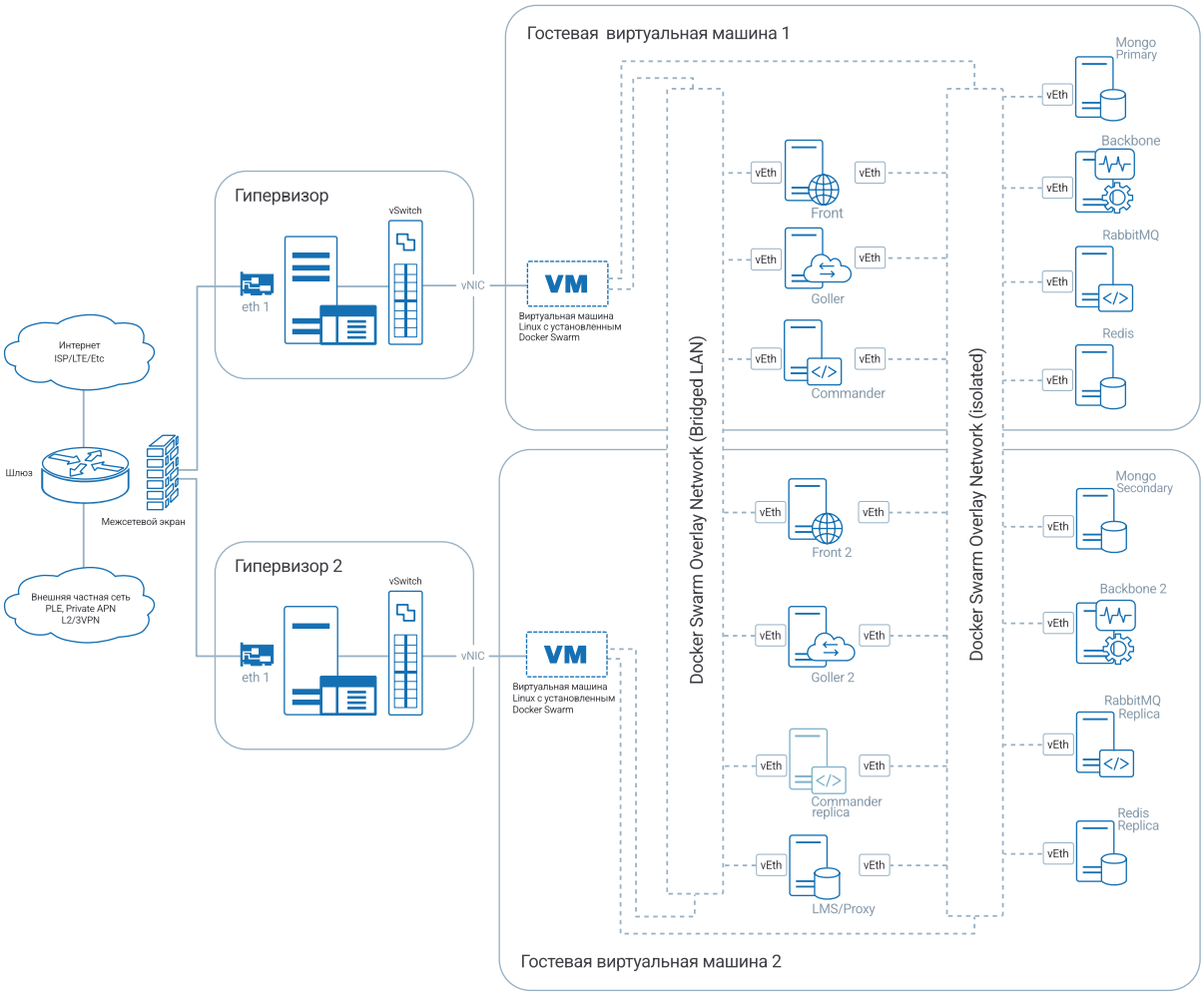
На рисунке ниже приведен пример схемы логического взаимодействия с указанием параметров настройки.



6.2.3. Максимальный-сценарий (для больших развертываний)

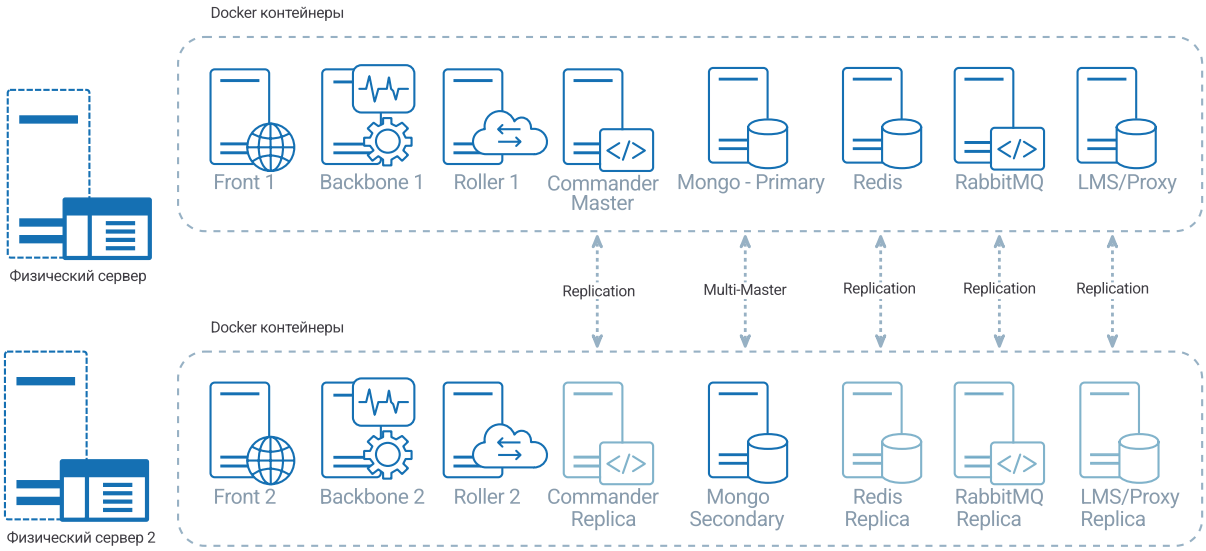
Для обеспечения отказоустойчивости Линк важно распределить не только нагрузку, но и дублировать критичные компоненты системы. Такой подход позволяет обеспечить непрерывную работу в нештатных ситуациях при сбое или отказе оборудования, программного обеспечения или неисправности инфраструктурных составляющих системы.

На рисунке представлена структурная схема Линк для Максимального-сценария развертывания. Под управлением находится более 500 устройств.



В данной конфигурации на каждом сервере запущено по одному экземпляру модулей Backbone, Roller, Front.

Модули Commander, Redis, RabbitMQ, LMS расположены на каждом сервере, но запущены только на одном. Реплики этих модулей на втором сервере будут запущены только в случае сбоя в работе Master-реплики.
Схема взаимодействия между модулями для сценария развертывания Максимального-сценария представлена на рисунке ниже.



Для повышения отказоустойчивости путем репликации данных, в СУБД MongoDB предусмотрен штатный механизм репликации - Mongo ReplicaSet.
После настройки репликации СУБД на несколько нод, в соответствии с начальной конфигурацией, одна из реплик БД становится основной (PRIMARY) и имеет приоритет 1. Модули Backbone и Roller обращаются с запросами на запись именно к этой реплике СУБД, с запросом на чтение из базы модули могут обратиться к любой из доступных актуальных реплик СУБД. При отключении ноды с основной репликой MongoDB статус другой реплики MongoDB будет изменен на PRIMARY.

Настройка:

- модули системы Линк размещаются на нескольких серверах, чтобы распределить нагрузку и повысить производительность;
- применяется балансировка нагрузки для модулей Front, Backbone и Roller, чтобы распределить входящие запросы между различными экземплярами модулей, обеспечивая высокую доступность;
- используется репликационный набор MongoDB, в котором данные дублируются на нескольких серверах, чтобы повысить отказоустойчивость и обеспечить доступность данных в случае сбоя одного из серверов;
- оркестратор Docker Swarm используется для автоматического управления развертыванием, масштабированием и управлением контейнерами, что упрощает администрирование и обеспечивает эффективное использование ресурсов.

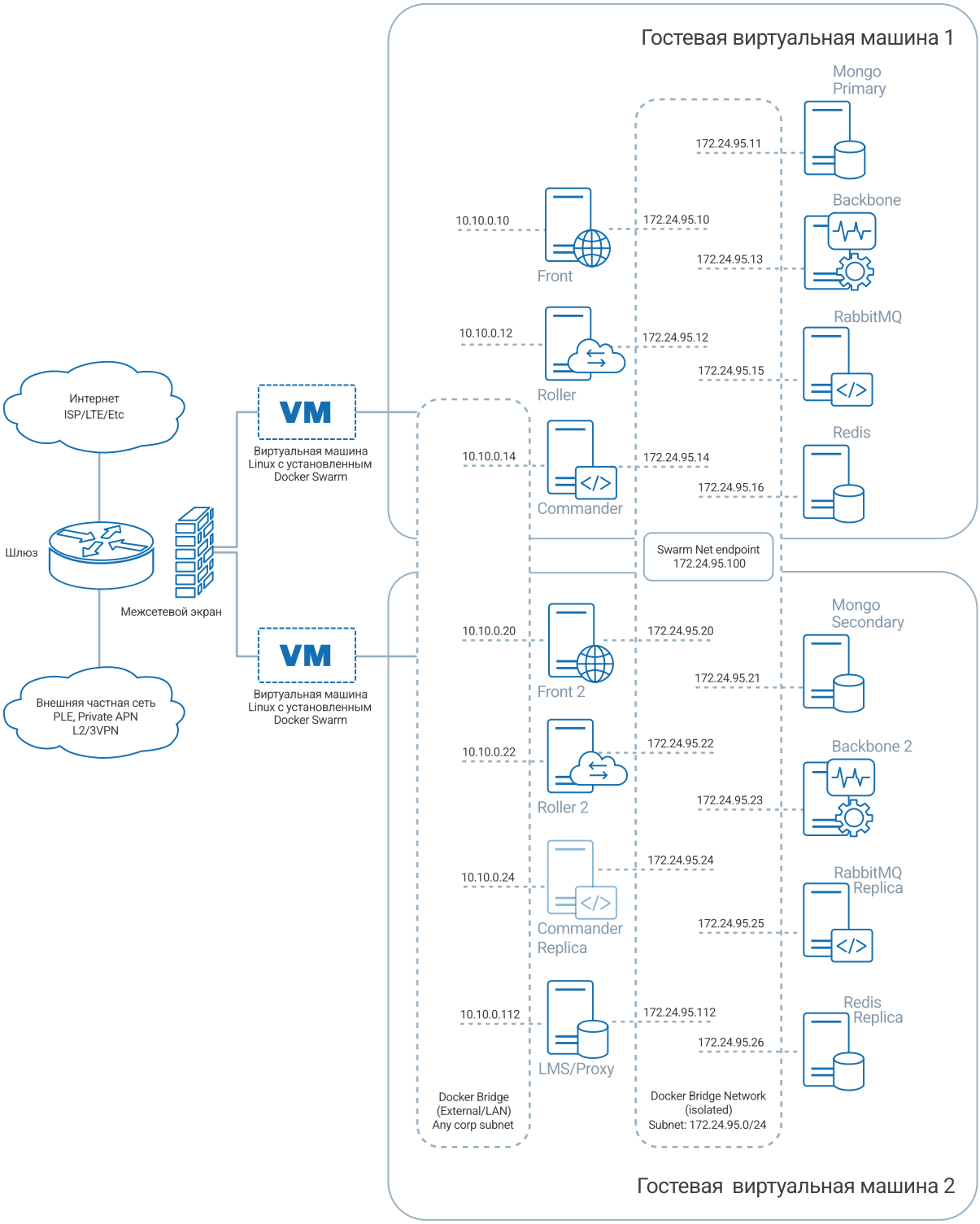
Преимущества:

- повышенная масштабируемость, так как система может быть расширена за счет добавления новых серверов;
- высокая доступность, так как система продолжает работать даже в случае отказа одного из серверов;
- улучшенная отказоустойчивость, так как данные дублируются, а компоненты реплицируются;

Ограничения:

- повышенная сложность, так как требуется настройка и управление несколькими серверами;
- потенциально более высокая стоимость, так как требуется больше серверов и ресурсов.

На рисунке ниже приведен пример схемы логического взаимодействия с указанием параметров настройки.


Гостевая виртуальная машина 1
Mongo Primary
172.24.95.11
Backbone
10.10.0.10
172.24.95.10
172.24.95.13
Front
RabbitMQ
10.10.0.12
172.24.95.12
172.24.95.15
Roller
Redis
10.10.0.14
172.24.95.14
172.24.95.16
Commander
Интернет
ISP/LTE/Etc
VM
Виртуальная машина Linux с установленным Docker Swarm
Шлюз
Межсетевой экран
Swarm Net endpoint
172.24.95.100
Mongo Secondary
10.10.0.20
172.24.95.20
172.24.95.21
Front 2
Backbone 2
10.10.0.22
172.24.95.22
172.24.95.23
Roller 2
RabbitMQ Replica
10.10.0.24
172.24.95.24
172.24.95.25
Commander Replica
Redis Replica
10.10.0.112
172.24.95.112
172.24.95.26
LMS/Proxy
Внешняя частная сеть
PLE, Private APN
L2/3VPN
VM
Виртуальная машина Linux с установленным Docker Swarm
Docker Bridge (External/LAN) Any corp subnet
Docker Bridge Network (isolated) Subnet: 172.24.95.0/24
Гостевая виртуальная машина 2

7. Отказоустойчивость

Отказоустойчивость системы Линк обеспечивается следующими механизмами:

- репликация MongoDB: MongoDB реплицирует данные между несколькими серверами, что гарантирует согласованность и доступность информации даже в случае отказа одного из серверов;
- избыточные компоненты: Ключевые компоненты, такие как Backbone и Roller, могут быть реплицированы на разных серверах, что обеспечивает резервное копирование в случае отказа одного из серверов;
- Docker Swarm: Docker Swarm автоматически управляет доступностью контейнеров и динамически перезапускает неисправные контейнеры на исправных узлах, обеспечивая непрерывную работу системы;
- балансировка нагрузки: Балансировщик нагрузки распределяет запросы между несколькими экземплярами Front и Roller.

8. Дополнительные возможности

Интеграция с системами мониторинга
Линк может быть интегрирован с другими системами мониторинга (например, Nagios, Zabbix), что позволяет объединять данные из разных источников и создавать единую картину состояния сетевой инфраструктуры.

Анализ данных
Линк может использовать встроенные инструменты анализа данных или интегрироваться с внешними системами аналитики, что позволяет выявлять тренды, аномалии и потенциальные проблемы в работе сетевых устройств.

Автоматизация задач
Линк позволяет автоматизировать задачи управления сетевыми устройствами, например: обновление прошивки, настройку конфигурации и выполнение других операций.

9. Заключение

Линк — это отказоустойчивая и масштабируемая система управления и мониторинга сетевого оборудования, позволяющая реализовать широкий спектр сценариев развёртывания. Ее модульная архитектура, использование Docker и Docker Swarm, а также репликация базы данных MongoDB обеспечивают высокую доступность, производительность и безопасность системы.