ПОДХОДЫ И ПРИНЦИПЫ К ПРОЕКТИРОВАНИЮ АРХИТЕКТУР СОВРЕМЕННОГО ПО (НА ПРИМЕРЕ ПАК «РУБИКОН NOVA»)

О СЕБЕ

Хмелёв Игорь Геннадьевич

- Инженер
- Более10 лет в IT и кибербезопасности
- Занимал различные позиции при разработке программного обеспечения
- Активно принимаю участие в CTF, конференциях и профильных мероприятиях
- Team Lead и архитектор в продукте класса NGFW

СИСТЕМА И АРХИТЕКТУРА

СИСТЕМА

Система – совокупность взаимосвязанных элементов, организованных для достижения поставленных целей

Свойства системы:

- Иерархичность – система имеет структуру
- Интеграция – элементы системы объединены общими целями и функционируют согласовано
- Эмерджентность – система всегда больше чем сумма ее элементов
- Устойчивость – сохранение состояние при воздействии внешних факторов
- Изолированность – система может быть открытой или закрытой
- Саморегулируемость – автоматическое изменение состояния
- Обратная связь – получение информации от объекта управления

АРХИТЕКТУРА

Для сложных систем нужно создавать архитектуру

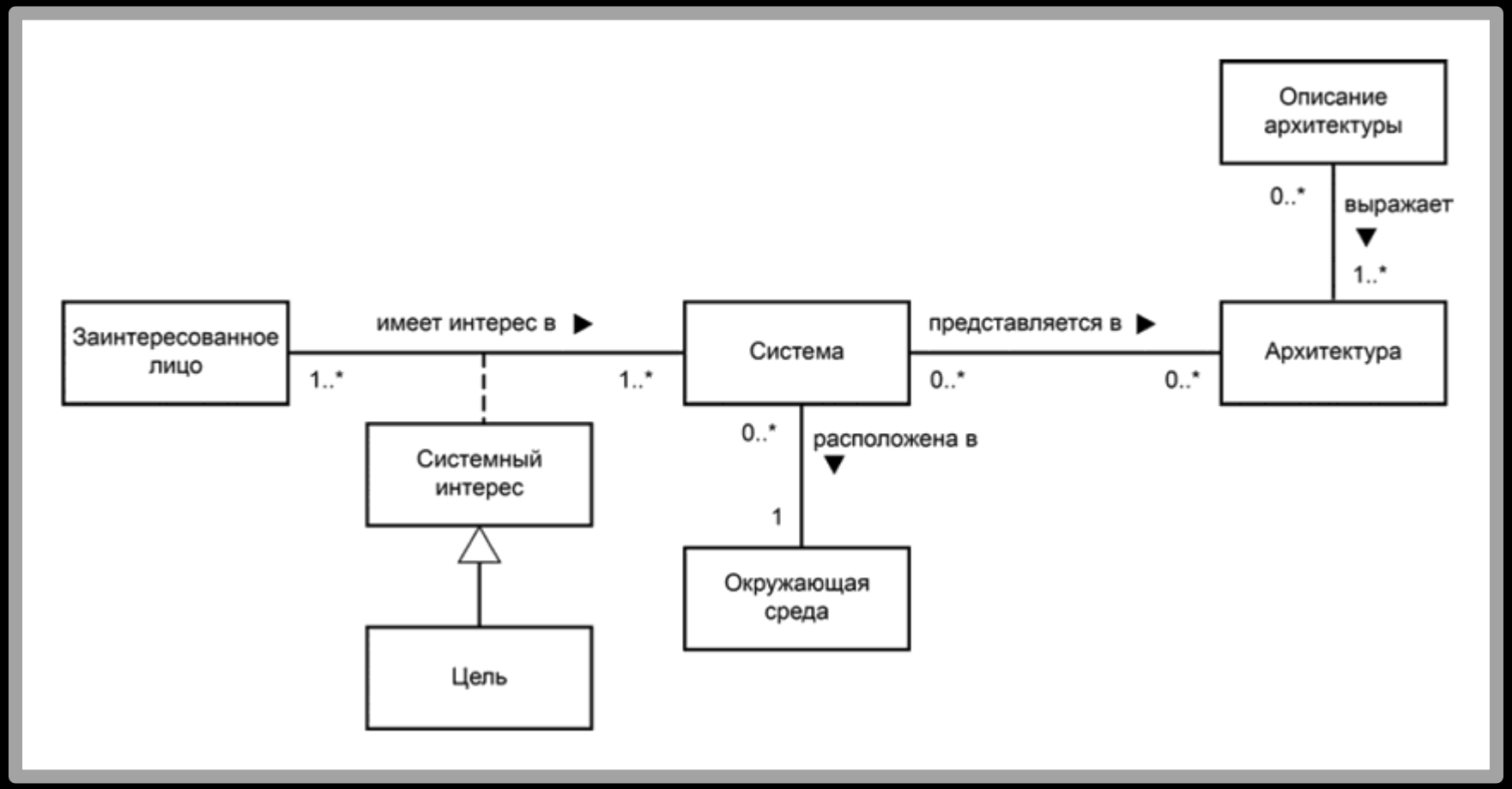
Архитектура – свойства системы, воплощенной в ее элементах
Описание архитектуры – выражение архитектуры

Задачи описания архитектуры:

- Определение компонентов системы
- Определение взаимодействия между компонентами как внутри системы, так и с внешними системами
- Создание структуры системы с учетом требований
- Создание описания системы, базы знаний, подходов и решений
- Управление сложностью проекта

ГОСТ Р 57100-2016/ISO/IEC/IEEE 42010:2011
Системная и программная инженерия. Описание архитектуры

ОПИСАНИЕ АРХИТЕКТУРЫ



ГОСТ Р 57100-2016/ISO/IEC/IEEE 42010:2011
Системная и программная инженерия. Описание архитектуры

ВИДЫ АРХИТЕКТУР ПО

Монолитная архитектура – вся функциональность реализуется внутри одного процесса

Многослойная архитектура – система делится на слои, каждый из которых выполняет свою роль

Сервис-ориентированная архитектура (SOA) – система делится на независимые сервисы, каждый из которых реализует заданный функционал

Микросервисная архитектура – система делится на набор независимых сервисов, каждый из которых реализует ограниченный функционал

Дополнительные подходы: Композитная, Georaспределенная (SDN)

ГРАНИЦЫ СИСТЕМЫ

При создании сложных систем создаются слои абстракции, между которыми осуществляются взаимодействия

Необходимо точно определять где проходят границы системы

API

API (Application Programming Interface) – набор правил и инструментов, позволяющих различным программам взаимодействовать друг с другом

Примеры:

- POSIX – взаимодействие между ПО и ОС
- File – данные передаются посредством файлов
- Socket – передача данных через сетевой стек
- PIPE – передача данных между процессами
- RPC (gRPC, XML RPC, JSON-RPC) – удалённый вызов процедур
- REST, SOAP, GraphQL, WebSocket – передача данных используемые в веб-технологиях
- Message Queue – обмен данными через очереди сообщений

Прочее интерфейсы взаимодействия:

- SATA, USB, PCI-E, RS-232, 802.3, 802.11 – аппаратные технологии для передачи данных
- TCP/IP – стек сетевых протоколов

ЗАИМСТВОВАННЫЕ КОМПОНЕНТЫ

При создании ПО используются заимствованные компоненты.

Вопрос:
Можно ли обойтись без заимствованных компонентов?

ЗАИМСТВОВАННЫЕ КОМПОНЕНТЫ

Заимствованные компоненты:

- Оборудование (CPU, GPU, NIC)
- Драйвера
- Операционные системы
- Языки программирования
- Компиляторы, интерпретаторы, среды сборки/развертывания (CI/CD) и тестирования
- Библиотеки
- Готовые фреймворки
- Готовые системы (базы данных, веб-сервера, реверс прокси)
- Интеграции (Сетевой стэк TCP/IP, протоколы)

Итоговая стоимость продукта + разработка аналога (целесообразно?)

АРХИТЕКТУРА

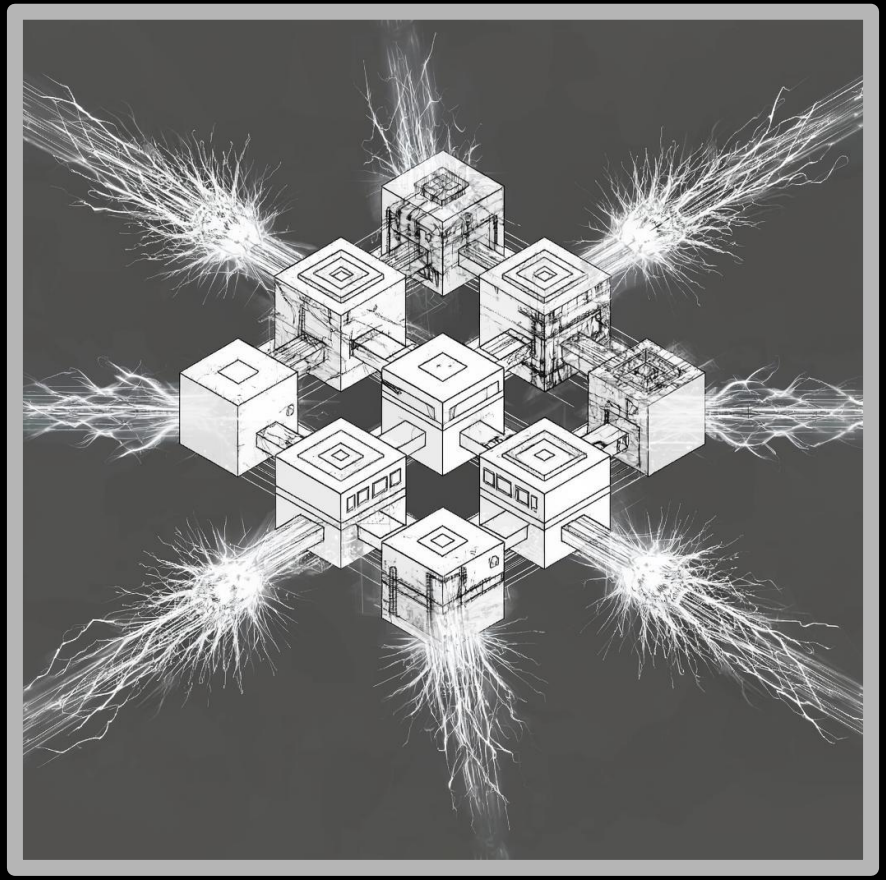
СЕРЕБРЕННАЯ ПУЛЯ

Есть ли универсальное решение при создании архитектуры?

В архитектуре ПО не существует единой “серебренной пули”

- Сложность современных систем
  - Множество слоев абстракции
  - Каждый слой имеет свою логику работы и взаимодействия
  - Движение в сторону технологической сингулярности
- Разнообразие доменных областей
  - В каждой области различные задачи и требования
  - Разнообразие архитектурных подходов и технологий
  - Различные стандарты и регламенты в разных компаниях и областях

АРХИТЕКТУРА И БЕЗОПАСНОСТЬ



АРХИТЕКТУРА И БЕЗОПАСНОСТЬ

Ошибки в архитектуре, влияющих на безопасность системы:

- Нарушение принципа минимальных привилегий
  - Слабый контроль доступа
  - Избыточные права доступа
  - Отсутствие сегментации доступа (разделение ответственности)
  - Открытые границы системы
- Небезопасная обработка данных
  - Хранение чувствительной информации в открытом виде
  - Передача чувствительной информации в открытом виде (отсутствие защиты при передачи данных)
  - Использование слабых алгоритмов шифрования
  - Неконтролируемое создание копии чувствительной информации

АРХИТЕКТУРА И БЕЗОПАСНОСТЬ

Ошибки в архитектуре, влияющих на безопасность системы:

- Уязвимости в обработке пользовательского ввода
  - Отсутствие валидации входных данных
  - Возможность выполнения различных инъекций
  - Возможность реализации XSS атак
- Недостаточное логирование и мониторинг системы
  - Отсутствие регистрации критических операций
  - Неполнота аудита безопасности
  - Низкая наблюдаемость системы
- Использование заимствованных компонентов
  - Наличие уязвимостей
  - Наличие скрытых каналов передачи данных

АРХИТЕКТУРА И БЕЗОПАСНОСТЬ

Ошибки в архитектуре, влияющих на безопасность системы:

- Проблемы с масштабируемостью
  - Неконтролируемое распределение нагрузки
  - Утечка памяти
  - Риск отказа в обслуживании (DoS)
- Избыточная сложность
  - Уменьшение надежности системы
  - Потенциальные ошибки при внесении изменений
  - Сложность поддержки и восстановления
- Отсутствует контроль целостности системы
- Отсутствует возможность обновления системы
- Отсутствует возможность резервирования системы
- Оценка влияние AI на разработку системы

АРХИТЕКТУРНЫЕ ПОДХОДЫ НА ПРИМЕРЕ РУБИКОН NOVA

РУБИКОН NOVA

Назначение:

- Сетевое устройства
  - Коммутатор уровня L2 (свитч)
  - Коммутатор уровня L3 (маршрутизатор)
- Выполнение функций безопасности:
  - Межсетевой экран (L2-L4)
  - Фильтрация приложений (L7, AppID)
  - Фильтрация сервисов (ServiceID)
  - Фильтрация по пользователям (UserID)
  - SSL инспекция, DPI, AV, IDPS, proxy
- Построение оверлейных сетей
- Интеграция с внешними СЗИ

Планируется более 200 функций

СТРУКТУРА

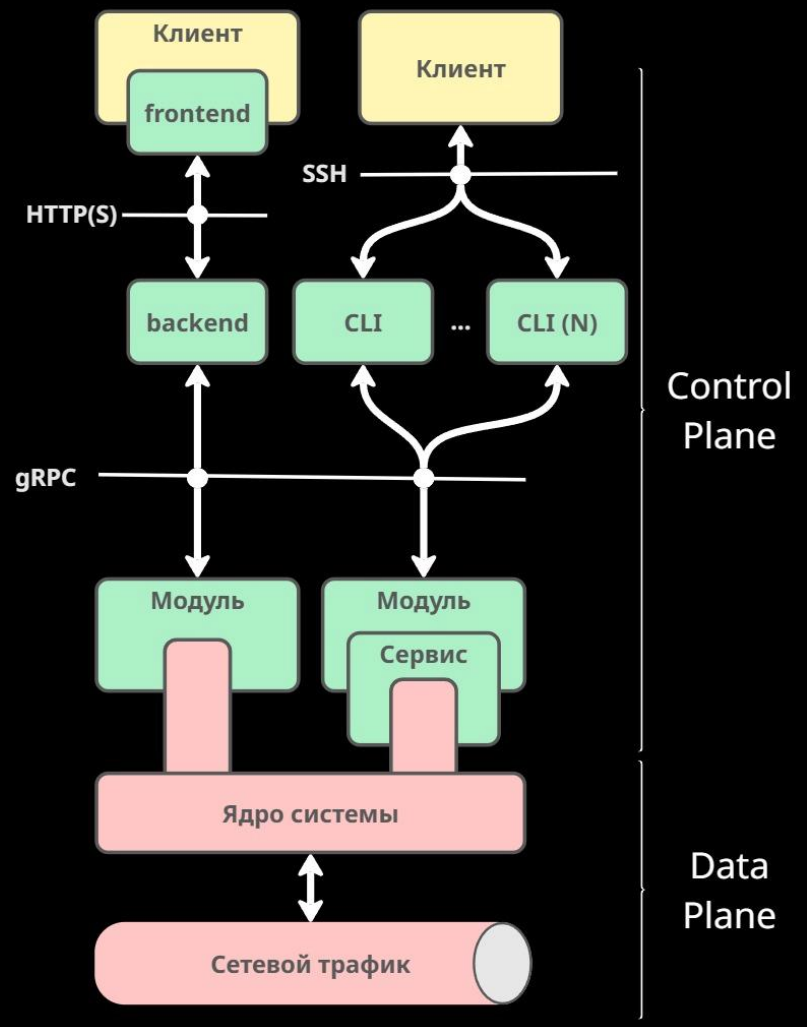
Интерфейс управления

Функциональны модули

Ядро системы
(Среда функционирования)

- Пользовательский интерфейс: WEB, CLI, SSH
- Реализация основной бизнес-логики
- Ядро системы на базе семейства Linux
- Аппаратная платформа: x86_64, ARM64

ДЕКОМПОЗИЦИЯ



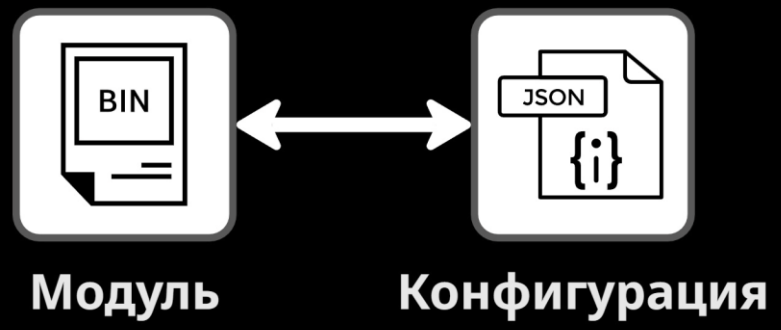
Композитная архитектура:

- Слои
- Сервис-ориентированная
- Unix way

Преимущества:

- Разделение на доменные области
- Выделенные границы проекта
- Наблюдаемость
- Масштабируемость
- Тестируемость на ранних стадиях
- Управляемость проектом и зависимостями
- Различный стек технологий

КОНФИГУРАЦИЯ



Архитектура это процесс

(предусмотрена возможность перехода на БД в случае необходимости)

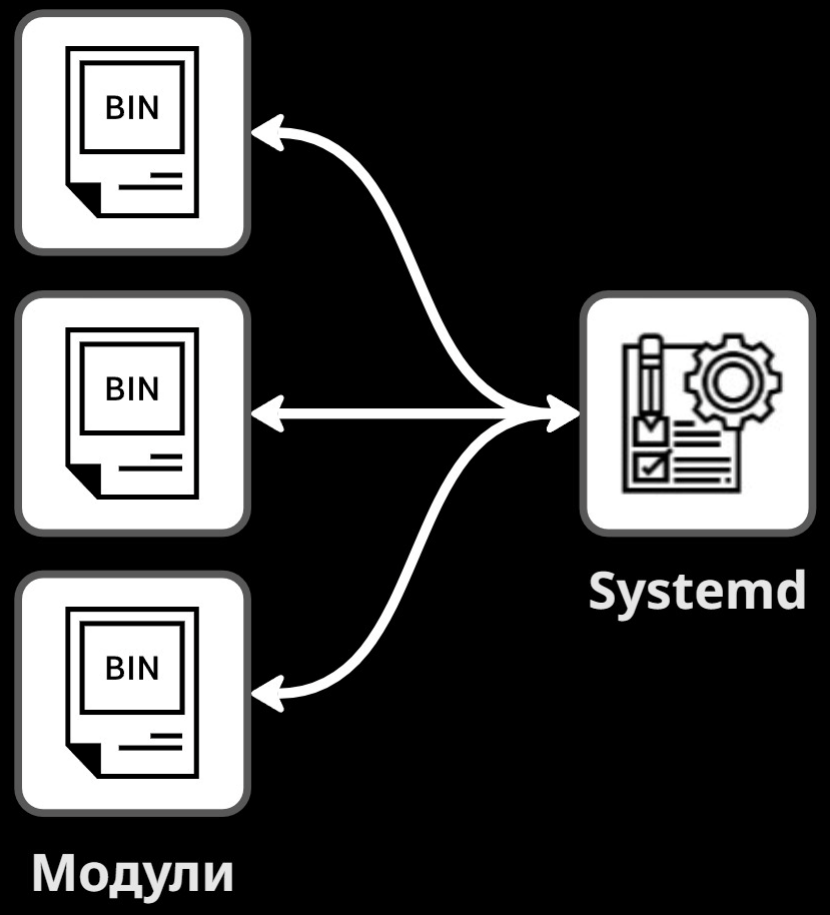
Конфигурация в виде файлов:

- Простота работы
- Изоляция и независимость
- Переносимость

Недостатки:

- Необходимость реализации функций, которые есть в БД из коробки (транзакции, контроль версии, резервирование)
- Масштабируемость

УПРАВЛЕНИЕ МОДУЛЯМИ

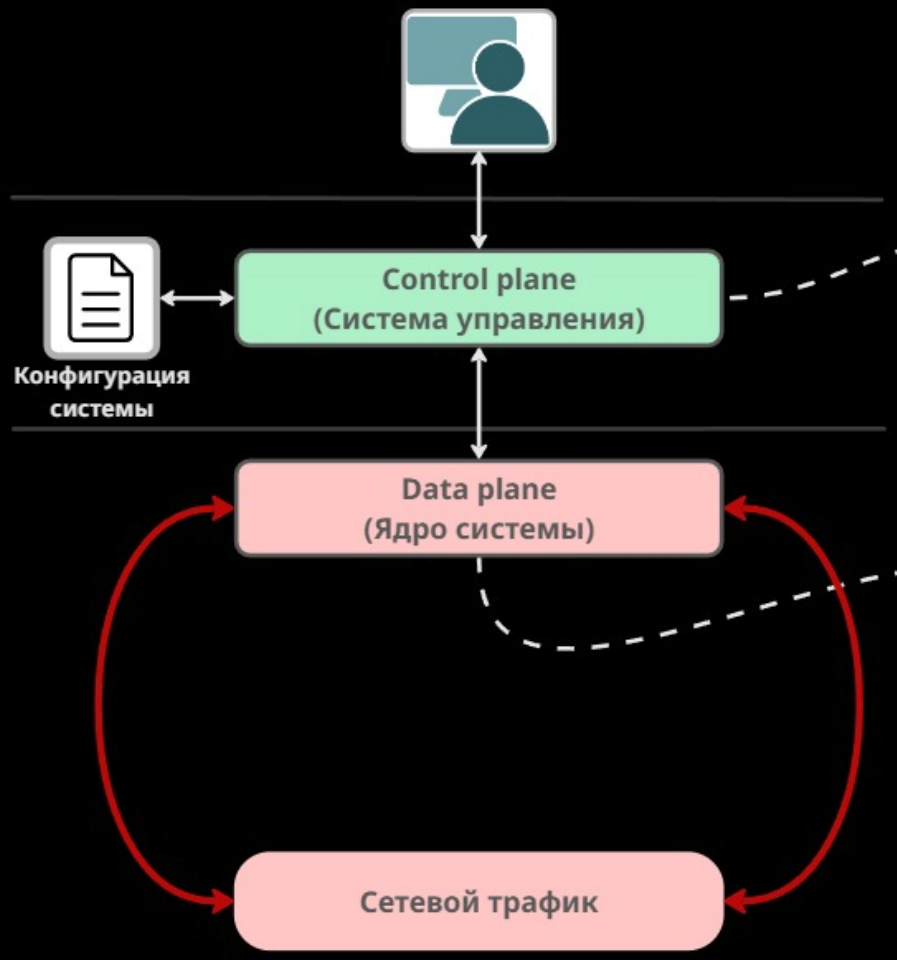


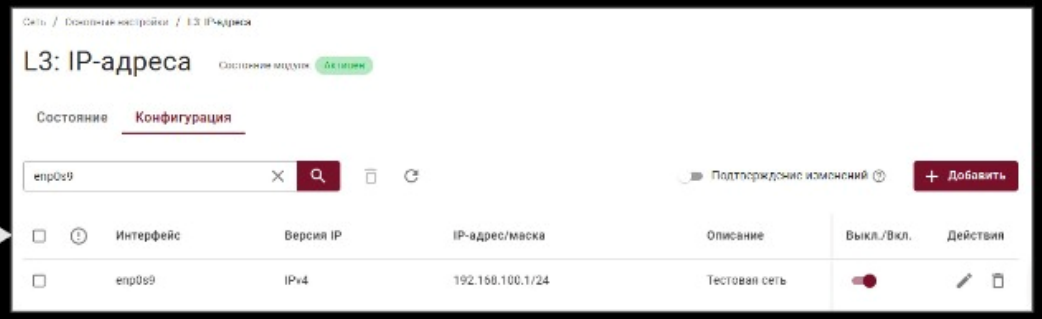
Централизованное управление процессами через systemd

Преимущества:

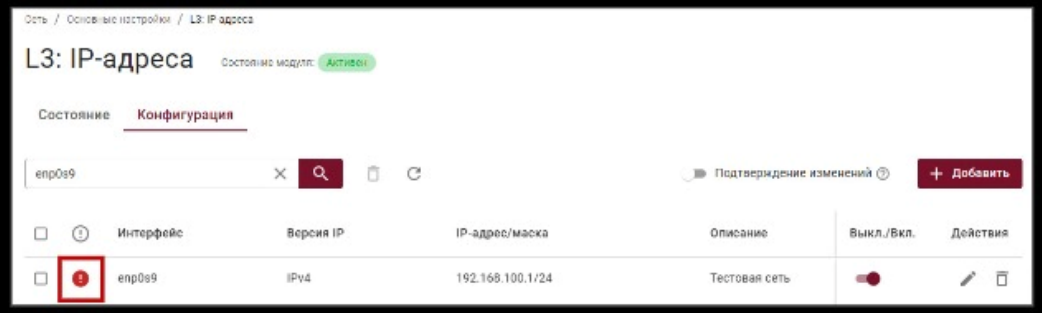
- Управление модулями
- Управление зависимостями между модулями
- Журналирование
- Обеспечение устойчивости системы

КОНТРОЛЬ СОСТОЯНИЯ





Автоматическая проверка согласованности ядра системы и конфигурации:



КОНТРОЛЬ СОСТОЯНИЯ

Режимы функционирования:

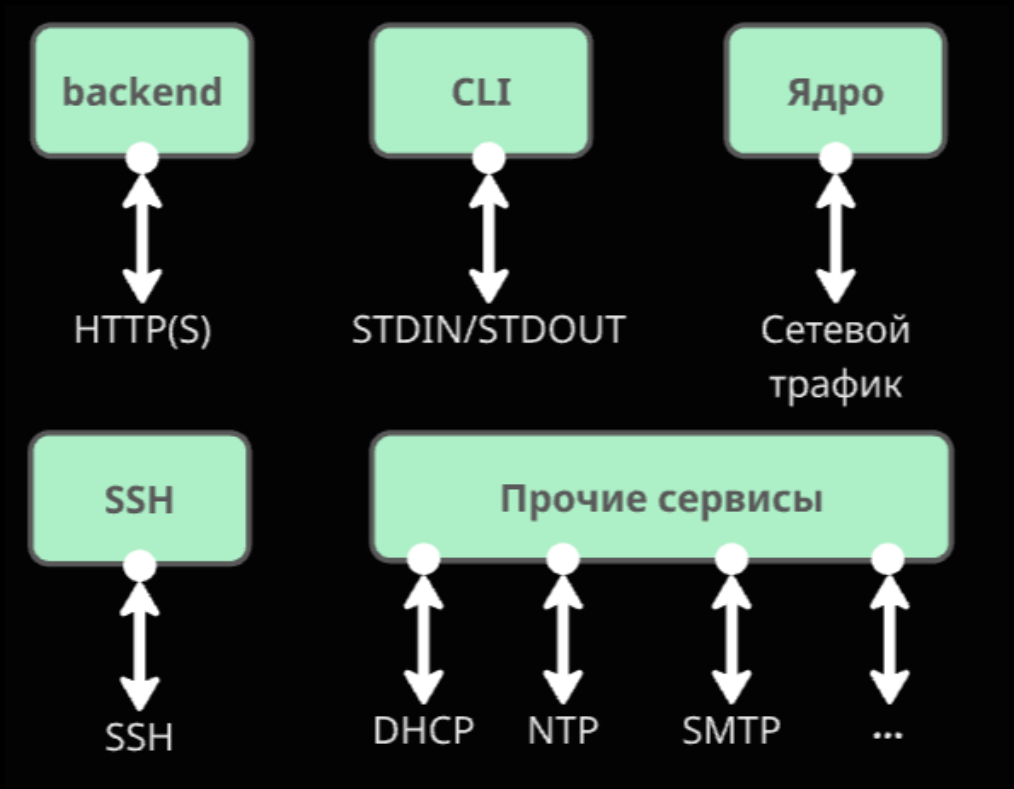
- Online
- Offline

Система контроля состояния, автоматически производит проверку состояния системы.

Нарушение функционирования:

- Логирование
- Уведомления
- Перевод в режим offline

ИНТЕРФЕЙСЫ ВЗАИМОДЕЙСТВИЯ

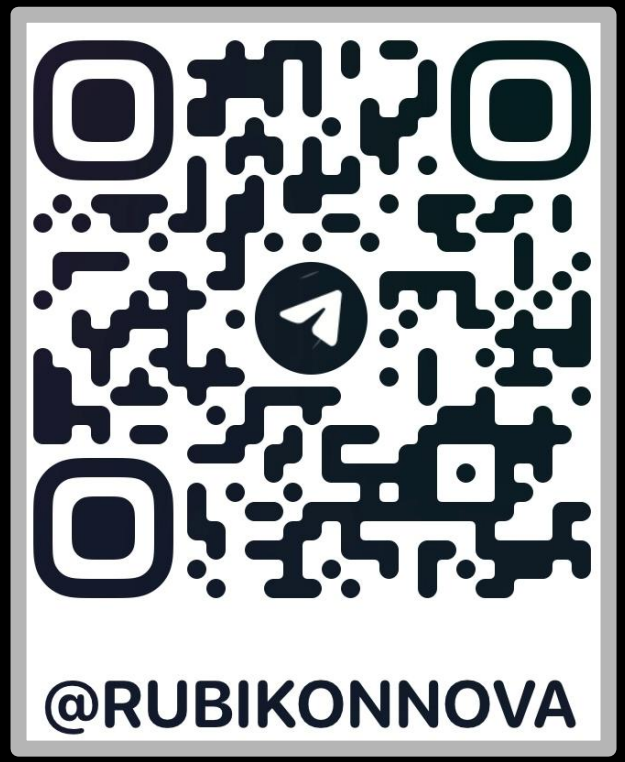


Внешние интерфейсы для взаимодействия с системой:

- Используемые протоколы (формат)
- Функциональные возможности
- Разграничение доступа
- Тип передачи данных (клиент/сервер, поток)

ВОПРОСЫ?

Спасибо за внимание



НПО Эшелон
https://npo-echelon.ru

https://t.me/RubikonNova