



Модуль балансировки нагрузки
Скала^р МС.БН

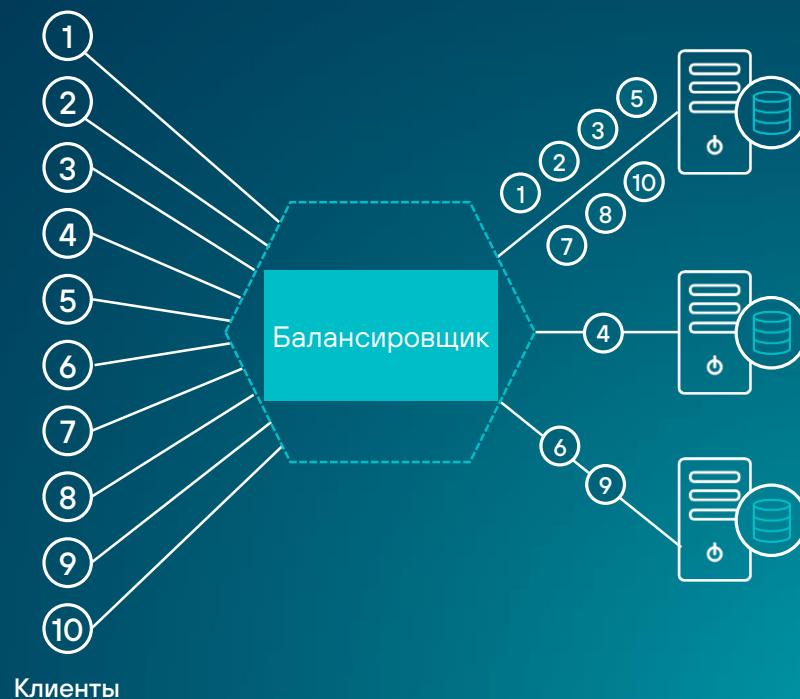
скала^р

О технологии балансировки нагрузки



Балансировка нагрузки или выравнивание нагрузки (англ. load balancing) — метод распределения заданий между несколькими сетевыми устройствами (например, серверами) с целью оптимизации использования ресурсов, сокращения времени обслуживания запросов, горизонтального масштабирования кластера (динамическое добавление/удаление устройств), а также обеспечения отказоустойчивости (резервирования). Процедура балансировки осуществляется при помощи целого комплекса алгоритмов и методов.

- **DNS балансировка** (Сетевой уровень) — Обеспечивает балансировку нагрузки и отказоустойчивость в нескольких центрах обработки данных.
- **L4 балансировка** (Транспортный уровень) — Распределение трафика идет на основе информации из протоколов TCP или UDP. Данный метод популярен в сетевых сервисах, где важно учитывать порты и протоколы, но не требуется анализировать содержимое трафика.
- **L7 балансировка** (Прикладной уровень) — Распределение идет на базе содержимого приложения (HTTP/HTTPS). Идеален для сложных веб-приложений, микросервисных архитектур, API-шлюзов, где нужно учитывать контекст запроса и принимать решения на основе содержимого HTTP-запросов.



Какие бывают балансировщики нагрузки?



- **Аппаратные балансировщики**

Это специализированные устройства, которые используются в центрах обработки данных для управления сетевым трафиком. Они предоставляют высокую производительность, безопасность и масштабируемость.

Примеры: F5 Networks, Citrix ADC (ранее NetScaler), A10 Networks.

- **Программные балансировщики**

Это программные решения, которые работают на стандартных серверах заказчика.

Они гибкие и часто более доступные по сравнению с аппаратными балансировщиками.

Примеры Open Source решений: HA Proxy, Nginx, Traefik, Apache HTTP Server

Балансировщики — ключевой компонент современных распределённых систем, который помогает им работать эффективно даже во время пиковых нагрузок.

Балансировщики применяются для:

- Повышения надежности системы
- Уменьшения времени ответа на запрос
- Оптимизации использования ресурсов
- Предотвращения перегрузки какого-либо одного ресурса
- Максимизации пропускной способности



Модуль балансировки нагрузки Скала^р

Описание продукта

Модуль балансировки нагрузки Скала^р MC.БН

представляет собой аппаратный отказоустойчивый кластер балансировки нагрузки с использованием OEM компонентов ПО Angie ADC

Возможности:

- Балансировка нагрузки на уровнях L3/L4/L7 модели OSI
- Глобальная балансировка на основе DNS (GSLB)
- Система управления и мониторинга
- Управление через API, CLI, GUI



Производительность кластера в режиме Active–Active —
2x 20 Гбит/с, сетевые интерфейсы 25 Гбит/с

Модуль балансировки нагрузки Скала[^]р



Функциональность

- **DNS-балансировка** — распределяются запросы, возвращая разные IP-адреса одного доменного имени для разных клиентов. Решения принимаются на основе географического расположения пользователей, загруженности серверов, политик маршрутизации
- **Балансировка нагрузки на уровне L4** — принимаются решения на основе алгоритма балансировки и информации о TCP/UDP-сессиях, включая IP-адреса, порты источника и назначения
- **Балансировка нагрузки на уровне L7** — анализ содержимое HTTP/HTTPS запросов (например, URL, заголовки, куки, методы запроса) и принятие решения о маршрутизации на основе этих данных
- **Удержание клиентских сессий** — возможность определения всех запросов от одного клиента, и направление их на один бэкенд
- **Обработка и ускорение TLS** — Балансировщик может терминировать на себе зашифрованные соединения, снимая таким образом нагрузку с серверов
- **Проверка доступности серверов** — используются гибкие механизмы проверки доступности серверов
- **Ролевая модель доступа** — набор полномочий, привязанных к должностям или рабочим задачам
- **Управление и мониторинг** — через API/GUI/CLI

Модуль балансировки нагрузки Скала[^]р



Поддерживаемые режимы балансировки

- **Round Robin** — запросы распределяются по серверам последовательно (используется по умолчанию)
- **Балансировка на основе веса** (weight) — запросы распределяются по серверам с учетом веса каждого сервера
- **Балансировка ip-hash** — сервер выбирается с помощью хэш-функции на основе IP-адреса клиента, что обеспечивает «sticky session». Гарантирует, что запросы от одного клиента будут попадать на один сервер, если этот сервер доступен
- **least_conn** — новый запрос направляется на сервер с минимальным количеством активных соединений (наименее загруженный)
- **least_bandwidth** — балансировка на основе наименьшего потребления полосы пропускания. Периодически вычисляется среднее использование полосы пропускания для каждого апстрим-сервера, используя формулу скользящего среднего для сглаживания колебаний со временем
- **least_packets** — балансировка на основе наименьшего числа пакетов в единицу времени
- **least_time** — балансировка на основе наименьшего времени ответа. Вероятность передачи соединения активному серверу обратно пропорциональна среднему времени его ответа
- **feedback** — механизм балансировки нагрузки по обратной связи (по произвольному параметру из ответа на основной или проверочный запрос)

Модуль балансировки нагрузки Скала^р

Поддерживаемые проверки доступности серверов



Балансировщик Скала^р МС.БН использует пассивные и активные проверки состояния бэкендов. Если бэкенд начинает отвечать с ошибками, то он временно исключается из пула доступных бэкендов.

- Пассивные проверки доступности определяют состояние бэкенд узлов на основе пользовательских ответов. Позволяет мгновенно принимать решение.
- Активные периодические проверки устанавливают доступность бэкенд узлов, в том числе со сложной логикой. Узел проходит проверку, если запрос к нему успешно выполняется с учетом всех заданных параметров.
- Механизмы определения доступности сервера: tcp-check, http-check, icmp-check.
- Slow start позволяет постепенно вводить в строй восстановленный бэкенд узел. Тем самым «непрогретый» бэкенд узел не будет немедленно испытывать критическую нагрузку.

Модуль балансировки нагрузки Скала[^]р

Распределение нагрузки



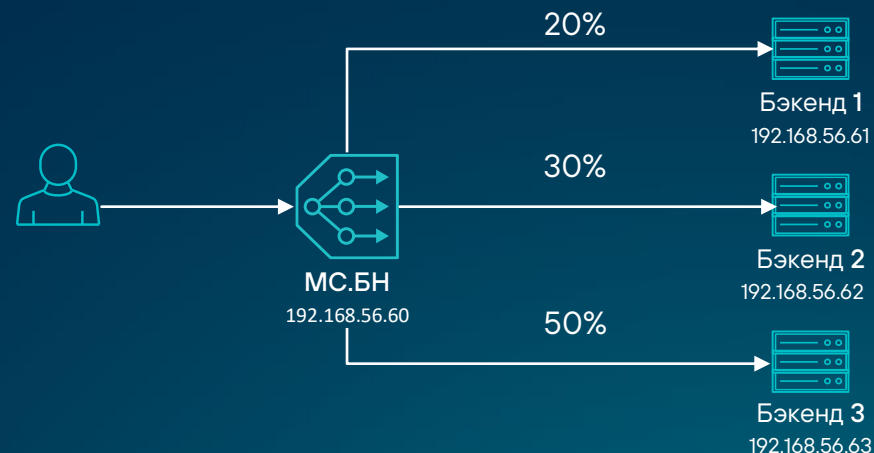
Основные методы балансировки можно использовать как по отдельности, так и в различных сочетаниях:

- **Round Robin** — запросы распределяются последовательно по каждому бэкенду
- **Weighted Round Robin** — аналогично, с учётом весов бэкендов
- **Least Connections** — новый запрос отправляется на бэкенд с наименьшим количеством активных соединений
- **Weighted Least Connections** — аналогично, но учитывается вес бэкендов

Весовые коэффициенты позволяют задать долю (вес) запросов, обрабатываемых каждым сервером приложений.

Вес может задаваться/определяться как статически, так и динамически по заданным критериям, с учётом, например:

- Производительности
- Текущей утилизации
- Времени отклика
- По настраиваемой пользователем логике



```
upstream myapp1 {
    server 192.168.56.61 weight=20;
    server 192.168.56.62 weight=30;
    server 192.168.56.63 weight=50;
}

server {
    listen 80;
    server_name localhost;

    location / {
        proxy_pass http://myapp1;
    }
}
```

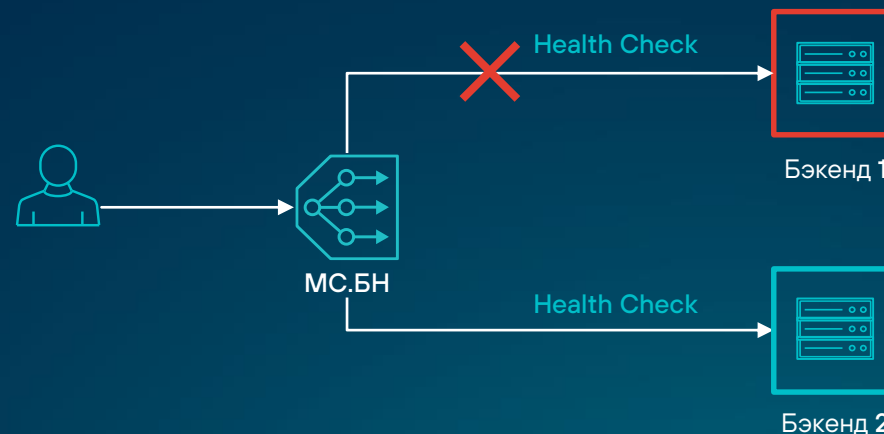
Модуль балансировки нагрузки Скала^р

Отказоустойчивость и проверки

Доступность бэкендов может контролироваться в Балансировщике динамически разными методами, отдельно или в разных сочетаниях, например:

- На уровне сети: **ICMP Ping** — проверка доступности через стандартный ICMP Echo Request
- На уровне транспорта: **TCP Health Check** (TCP Ping) — проверка доступности порта приложения
- На уровне сессии и/или приложения: **HTTP(S) Health Check** — проверка через запрос специального URL, и ожидание корректного кода ответа
- **Кастомизированная** проверка по пользовательской логике — с помощью встроенных или внешних скриптов
- **Пассивный мониторинг** — без активного опроса, Балансировщик определяет доступность бэкенда по наблюдениям на успешными/неуспешными реальными соединениями

Конечное решение может быть принято как по отдельным критериям оценки работоспособности, так и по совокупности



```
upstream myapp1 {  
    server 192.168.56.61 max_fails=2  
    fail_timeout=10s;  
    server 192.168.56.62 max_fails=2  
    fail_timeout=10s;  
    server 192.168.56.63 max_fails=2  
    fail_timeout=10s;  
}  
  
server {  
    listen      80;  
    server_name localhost;  
  
    location / {  
        proxy_pass http://myapp1  
    }  
}
```

Модуль балансировки нагрузки Скала[^]р

Конвертация протоколов

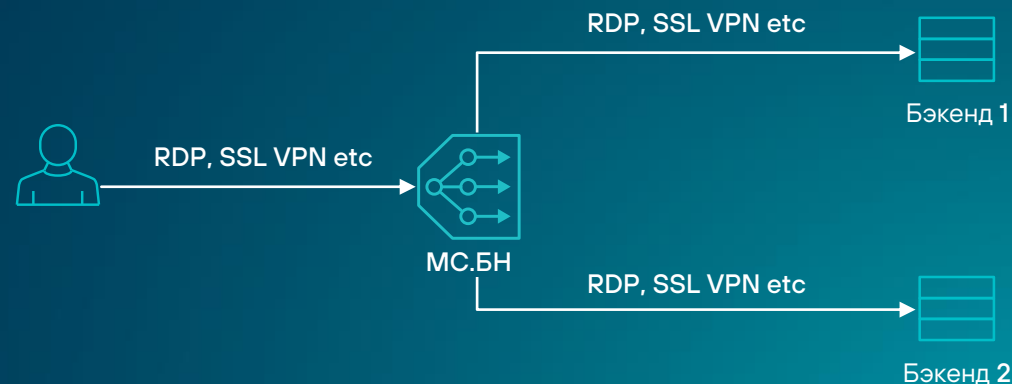
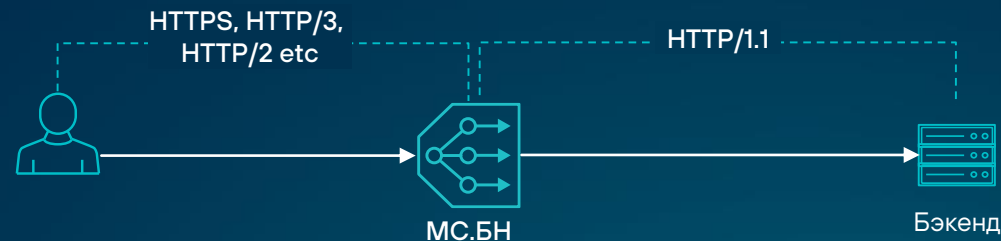


Балансировщик может в ряде сценариев преобразовывать протоколы с фронта на бэкенд.

Например, Балансировщик нагрузки может терминировать на себе **HTTP/HTTPS** (SSL) трафик в различных вариациях и использовать **HTTP без шифрования** при коммуникациях с бэкендами.

Также возможна балансировка трафика с statefull- и stateless-сессиями, по протоколам **TCP/UDP** на уровне L4 без раскрытия, с применением тех же правил и методик, как и для балансировка HTTP.

Например, можно применять Балансировщик для распределения нагрузки на **RDP/VDI** фермы, **SSL VPN** шлюзы.



Модуль балансировки нагрузки Скала[^]р

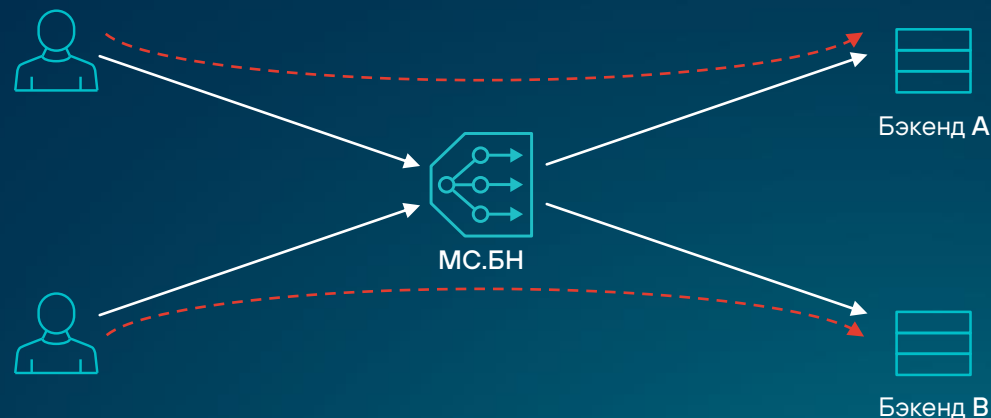
Персистентность сессий и распределение



Балансировщик может распределять нагрузку с учётом персистентности клиентских сессий — чтобы трафик от одного пользователя терминировался на одном бэкенде до тех пор, пока он доступен.

Это достигается за счёт применения **хэширования** пользовательских идентификаторов (в простейшем случае — по IP) в динамическом режиме.

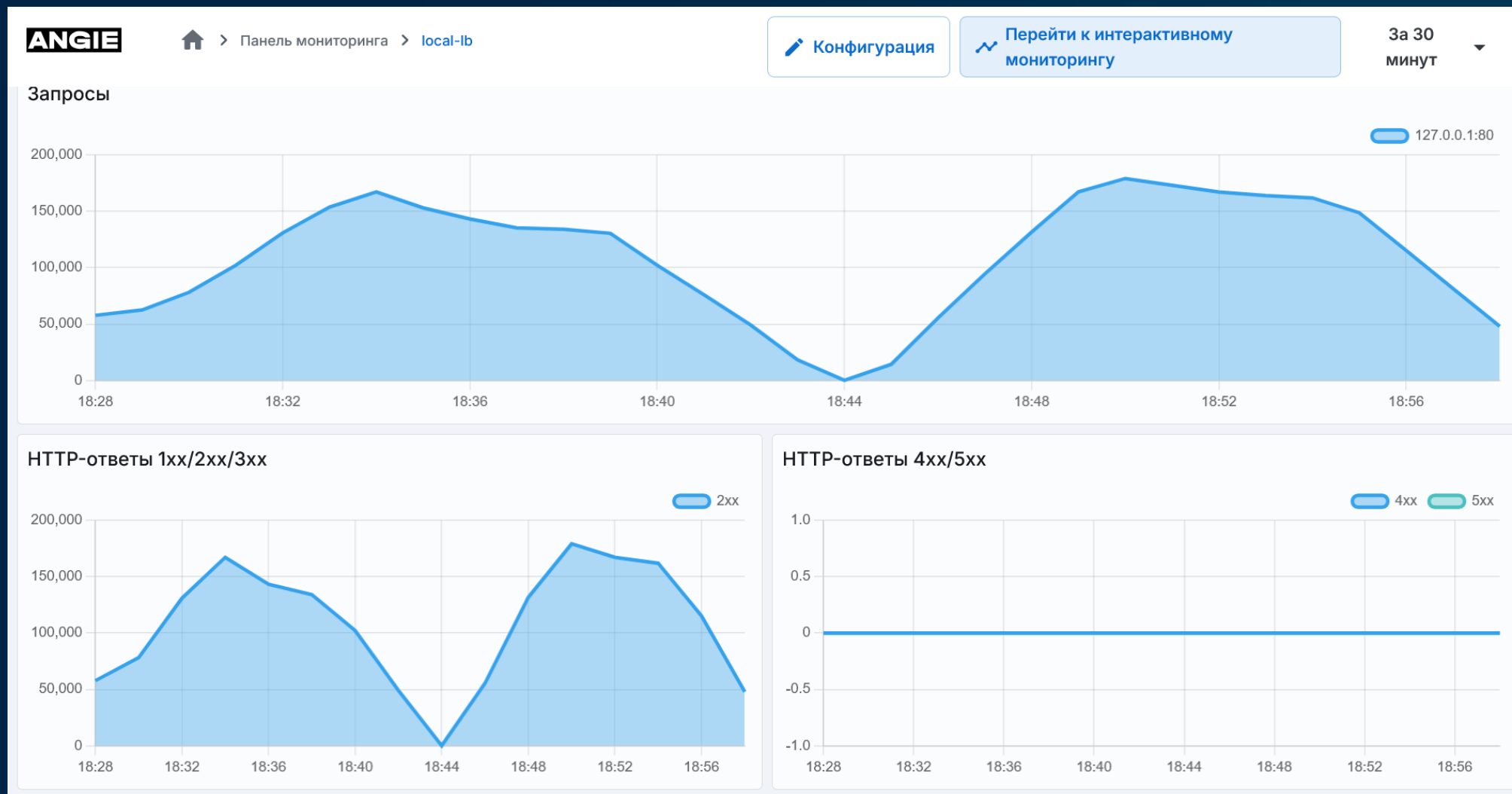
Кроме этого, типовые сценарии балансировки предполагают **перенаправление определённой доли** клиентского трафика на выделенный отдельный бэкенд (или группу), например, для постепенного релиза новой версии приложения или сервиса на ограниченную аудиторию.



```
upstream myapp1 {  
    ip_hash;  
    server 192.168.56.61;  
    server 192.168.56.62;  
}  
  
server {  
    listen 80;  
    server_name localhost;  
  
    location / {  
        proxy_pass http://myapp1;  
    }  
}
```

Модуль балансировки нагрузки Скала^р

Мониторинг и графические панели параметров работы



Модуль балансировки нагрузки Скала^p



Сравнение с конкурентами

№	Параметр	HA Proxy	Nginx	Скала^p MC.БН	F5
1	Периодическая проверка доступности обслуживающих (бэкенд) серверов	✓		✓	✓
2	Несколько алгоритмов определения доступности сервера: tcp-check, http-check, mysql-check	✓		✓	✓
3	Балансировка HTTP / HTTPS / TCP-запросов между «живыми» серверами	✓	✓	✓	✓
4	Поддержка TLS	✓	✓	✓	✓
5	Возможность закрепления определенных клиентов за конкретными обслуживающими серверами (stick-tables)	✓		✓	✓
6	Поддержка HTTP/2.	✓	✓	✓	✓
7	Балансировка на основе наименьшего потребления полосы пропускания (least bandwidth)			✓	✓
8	Балансировка на основе наименьшего числа пакетов в единицу времени (least packets)			✓	✓
9	Балансировка нагрузки на основе кратчайшего ответа (least time)			✓	✓
10	Постепенная подача трафика на проксируемый сервер (slow start)			✓	✓
11	Обновление списка upstream серверов по SRV-записям DNS	✓	✓	✓	✓
12	Конфигурация через API проксируемых серверов без перезапуска приложения			✓	✓
13	Балансировка по произвольному параметру из ответа на основной или проверочный запрос		✓	✓	✓
14	Поддержка балансировки GSLB с учетом активных проверок upstream серверов у каждого балансера			✓	✓

№	Параметр	HA Proxy	Nginx	Скала^p MC.БН	F5
15	Поддержка протоколов динамической и статической маршрутизации			✓	✓
16	Встроенная поддержка ACME для перевыпуска сертификатов без использования сторонних утилит (таких как certbot)			✓	✓
17	Поддержка ГОСТ TLS на основе КристоПро			✓	
18	API для интеграции с Prometheus для сбора метрик в реальном времени			✓	✓
19	Визуальный интерфейс управления конфигурацией и мониторинга с ролевой моделью доступа			✓	✓
20	Наличие технической поддержки производителем на территории России			✓	
21	Возможность приоритизации дорожной карты с вендором			✓	
22	Присутствие в реестре отечественного ПО			✓	
23	GSLB на основе хеша IP адреса назначения/источника, хеш порта/источника				✓
24	GSLB на основе наименьшего числа соединений				✓
25	GSLB на основе кратчайшего ответа				✓
26	GSLB на основе RTT (round-trip time)				✓
27	Возможность аутентификации администратора по Tacacs/Radius/Ldap				✓
28	Поддержка SNMPv3 и SNMP Trap				✓
29	Детальное логирование				✓
30	Возможность формирования отчетов				✓
31	Откат настроек на предыдущее состояние				✓
32	iRules				✓

Модуль балансировки нагрузки Скала^р

Аппаратная поставка в виде модуля МС.БН



- Функциональность локальной балансировки (LTM)
- Функциональность глобальной балансировки (GTM)
- Поддержка динамических протоколов маршрутизации BGP, OSPF, IS-IS
- Сетевые интерфейсы 4x25 Гбит/с на сервер
- Кластер Active-Active
- Производительность 2x20 Гбит/с на Модуль

Дополнительные преимущества:

- Присутствие в реестре ПО (Реестровая запись №24972)
- Локальная техническая поддержка от вендора Скала^р 8x5 или 24x7
- Возможность доработки функционала по требованиям заказчика





Модуль балансировки
нагрузки
Скала^p MC.BH