?????????
??????????????????

В этой главе описана настройка внешней балансировки (на практике)

?????????????? ????? ???????

Общая схема прохождения трафика выглядит следующим образом:

Proxy-Route-MAP.png

Здесь все пользователи, для доступа к целевому ресурсу, проходят через единую точку входа - WAN-proxy.

????????? ?????????????? ????????

В общем случае, мы будем использовать HAProxy как решение для балансировки нагрузки. Установка для большинства Debian-based дистрибутивов Linux будет следующей:

```
# Синхронизация даты и времени (для примера - Мск_)
timedatectl set-timezone Europe/Moscow
# Обновление индекса пакетного менеджера и установленных пакетов
apt update && apt upgrade -y
# Установка
apt install -y haproxy
```

А дальнейшая настройка зависит от того, где именно будет выполняться балансировка: в WAN или LAN-сегменте.

????????? HAProxy ? WAN-????????

Для WAN-прокси основной задачей является соотношение доменного имени (SNI), к которому обращается клиент, с целевым ресурсом, который находится в LAN-сегменте за нашим шлюзом (маршрутизатором). В общем случае конфигурация будет состоять из 2 этапов:

????????? ????????? ???????????????? ?????

- Основной конфигурационный файл HAproxy располагается по следующему пути: /etc/haproxy/haproxy.cfg
- Пример основной конфигурации:

```
global
    log /var/log/haproxy local0 info
    log /var/log/haproxy local1 notice
    chroot /var/lib/haproxy
    stats socket /run/haproxy/admin.sock mode 660 level admin expose-fd listeners
    stats timeout 30s
    user haproxy
    group haproxy
    daemon

    # Оптимизация производительности
    maxconn 100000
    tune.ssl.default-dh-param 2048
    ssl-default-bind-options no-sslv3 no-tlsv10 no-tlsv11
    ssl-default-bind-ciphers ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384
    ssl-default-server-options no-sslv3 no-tlsv10 no-tlsv11
    ssl-default-server-ciphers ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384

    # Увеличенные буферы для больших загрузок
    tune.bufsize 32768
    tune.maxrewrite 1024

defaults
    log     global
```

```
    mode    tcp
    option  tcplog
    option  dontlognull
    option  tcpka
    option  clitcpka
    option  srvtcpka
    timeout connect 10s
    timeout client 1h
    timeout server 1h
    timeout tunnel 1h
    retries 3

# Мониторинг HAProxy (опционально)
frontend stats
    bind *:8404
    stats enable
    stats uri /stats
    stats refresh 10s
    stats admin if LOCALHOST

# Основной фронтенд для HTTPS трафика
frontend https-in
    bind *:443

    # ACL для определения сервисов по SNI
    acl host_nextcloud req.ssl_sni -i nextcloud.yourdomain.com
    acl host_service1 req.ssl_sni -i service1.yourdomain.com
    acl host_service2 req.ssl_sni -i service2.yourdomain.com

    # Используем таблицу для динамического маршрутизации (альтернативный вариант)
    use_backend %[req.ssl_sni,map_str(/etc/haproxy/sni-map.map,unknown_backend)]

    # ИЛИ использовать статические правила (раскомментировать если нужно):
    # use_backend nextcloud_backend if host_nextcloud
    # use_backend service1_backend if host_service1
    # use_backend service2_backend if host_service2

    # TCP-инспекция для извлечения SNI
    tcp-request inspect-delay 5s
    tcp-request content accept if { req_ssl_hello_type 1 }
```

```
# Бэкенд по умолчанию для неизвестных доменов
backend unknown_backend
    mode tcp
    option tcplog
    timeout server 5s
    server unknown 127.0.0.1:81

# Шаблон для обычных сервисов
backend service_template
    mode tcp
    option tcplog

    # Параметры балансировки (если несколько серверов)
    balance roundrobin

    # Настройки проксирования
    option tcp-check
    tcp-check connect

    # Поддержка Keep-Alive
    option clitcpka
    option srvtcpka

    # Увеличенные таймауты
    timeout connect 10s
    timeout server 1h
    timeout tunnel 1h

    # Сервер шаблона (заменить в конфигах сервисов)
    server template-server GATEWAY_IP:TEMPLATE_PORT check inter 2s fall 3 rise 2
```

Вот основные моменты этой конфигурации:

1. Настраиваем логирование (/var/log/haproxy)
2. Настраиваем бэкенд по-умолчанию (127.0.0.1:81, что в обычной конфигурации приведет к ошибке ConnectionTimeout)
3. Используем карту SNI, которая располагается здесь: /etc/haproxy/sni-map.map;
4. Разрешаем публичную статистику (она доступна на порту этого сервера, *:8404);
5. Мы используем TCP-прокси, т.е. не инкапсулируем входящие пакеты.

Логика работы со стороны Haproxy такая:

1. Возьми SNI из пакета, который тебе пришел от клиента;
2. Соотнеси полученный SNI с твоей базой (картой), которая лежит в /etc/haproxy/sni-map.map;
3. Если ты нашел доменное имя, на которое обращается клиент - проксируй весь трафик туда (как есть);
4. Если клиент общается на SNI, которого нет в твоей карте - отправляй его на самого себя же, на порт *:81 (там может стоять Nginx с сайтом-заглушкой).

????????? ????? ???????? (SNI)

Файл маппинга, согласно конфигурации выше, лежит по пути /etc/haproxy/sni-map.map. Изменим его, примерно следующим образом:

```
# Формат: доменное_имя бэкенд
nextcloud.yourdomain.com nextcloud_backend
service1.yourdomain.com service1_backend
service2.yourdomain.com service2_backend
# Добавляйте новые сервисы здесь
```

????????? ???????????????? ????? ??? ?????? ?? ????? (????? ???????)

??? **LAN**

(???????????) ????????? ???????????? ? ??????? ???????????????? ? LAN-????????