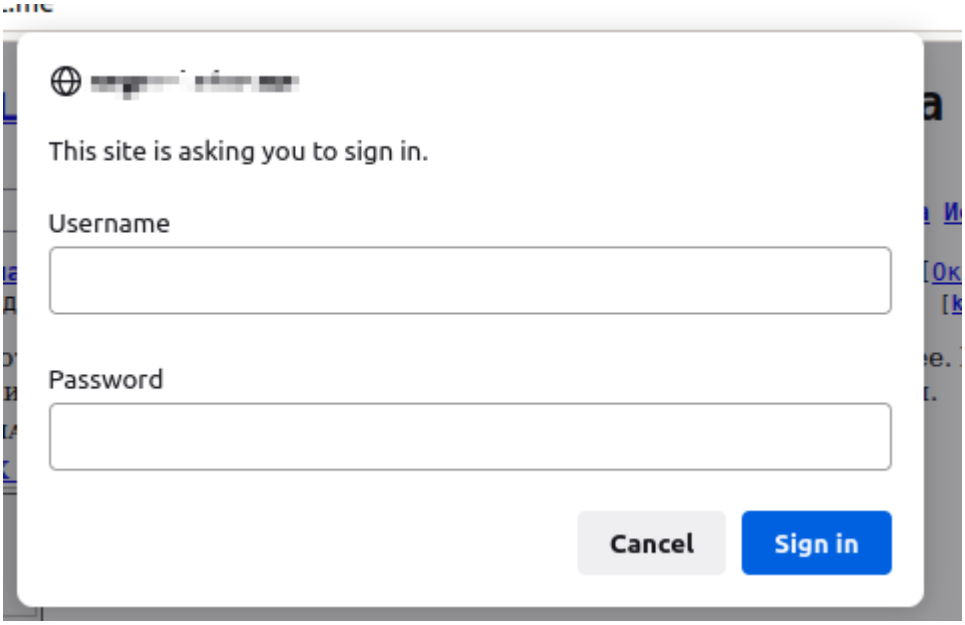
Обход блокировок: настройка сервера XRay для Shadowsocks-2022 и VLESS с XTLS-Vision, Websockets и фейковым веб-сайтом

habr.com/ru/articles/728836

Deleted user April 13, 2023



Deleted-user 13 апр 2023 в 22:24
Средний

14 мин

289K
Туториал

Предыдущие статьи серии:

«Современные технологии обхода блокировок: V2Ray, XRay, XTLS, Hysteria и все-все-все»

«Программы-клиенты для протоколов недетектируемого обхода блокировок сайтов: V2Ray/XRay, Clash, Sing-Box, и другие»

С протоколами разобрались, с клиентами разобрались, теперь наконец-то настало время рассказать о том, как же настроить свой личный прокси-сервер с современными протоколами для обхода блокировок. Мы будем настраивать сервер на базе XRay (который является форком известного V2Ray, и еще я немного упомяну Sing-Box) с протоколами Shadowsocks-2022 и VLESS с транспортом

XTLS-Vision и фейковым веб-сайтом для защиты от выявления. И в качестве запасного варианта на том же сервере мы настроим fallback на VLESS+Websockets, чтобы была возможность работать через CDN типа Cloudflare, если вдруг IP-адрес вашего сервера попадет под блокировку. В конце я приведу настройки десктопных и мобильных клиентов для подключения ко всему этому.

Поехали.

Настройку буду описывать под Debian или Ubuntu Linux. Если у вас на VPS стоит другой дистрибутив, то там будет примерно все то же самое, хотя некоторые команды и названия пакетов могут отличаться.

Итак, допустим у нас уже есть VPS с Debian или Ubuntu в какой-нибудь заморской юрисдикции, у него есть IP-адрес, на нем настроен SSH и вообще все пока что неплохо. И еще у вас должен быть какой-нибудь домен, не обязательно платный (хотя сейчас по акциям можно зарегистрировать домен в какой-нибудь не очень популярной зоне всего за доллар-два в год), подойдет даже DynDNS. Если чего-то из вышеописанного у вас нет, советую ознакомиться этой и этой статьей, там в начале описывается базовая установка и настройка VPS с Linux и регистрация бесплатного домена через DynDNS. Ну а мы идем дальше.

Первый вариант настройки я приведу для "пустого сервера" - это если на вашем сервере нет никаких других сервисов (но потом можно будет добавить еще и веб-сайт, да). Во второй половине статьи я расскажу, как настроить XRay когда у вас на машине уже крутится веб-сервер и вы не хотите лишний раз трогать его конфигурацию. Третий вариант будет с Docker ~~для самых маленьких~~.

Вариант первый, полный, подробный

Разработчики XRay подготовили скрипт, который автоматически скачивает XRay под используемую систему и создает systemd-юнит (спасибо @alegz81 что напомнил): https://github.com/XTLS/Xray-install

Устанавливается одной длинной командой

```
$ bash -c "$(curl -L https://github.com/XTLS/Xray-install/raw/main/install-release.sh)" @ install
```

Единственное различие от описанного в статье при конфигурации будет в том, что конфиги XRay будут лежать не в /opt/xray, а в /usr/local/etc/xray/.

Либо же можем установить все ручками.

Идем вот сюда: https://github.com/XTLS/Xray-core/releases и скачиваем самый свежий билд XRay-core:

```
$ wget https://github.com/XTLS/Xray-core/releases/download/v1.8.0/Xray-linux-64.zip
```

Создаем директорию, распаковываем и делаем файл исполняемым (он поставляется в .zip-архиве, поэтому разрешения при упаковке-распаковке теряются):

```
$ mkdir /opt/xray
$ unzip ./Xray-linux-64.zip -d /opt/xray
$ chmod +x /opt/xray/xray
```

Далее создадим systemd-юнит и вставим туда следущий текст (я использую nano, вы, понятное дело, можете использовать vi и вообще все что угодно):

```
$ nano /usr/lib/systemd/system/xray.service

[Unit]
Description=XRay

[Service]
Type=simple
Restart=on-failure
RestartSec=30
WorkingDirectory=/opt/xray
ExecStart=/opt/xray/xray run -c /opt/xray/config.json

[Install]
WantedBy=multi-user.target

$ systemctl daemon-reload
$ systemctl enable xray
```

Обратите внимание - в данном случае xray запустится от пользователя root. Это не очень хорошо в плане безопасности, я сделал это так в примере для упрощения мануала, но по-хорошему нужно создать для xray отдельного пользователя, запускать его от него, не забыть выставить ему права для чтения на директории и файлы от certbot/letsencrypt (об этом чуть дальше), и чтобы была возможность повесить сервер на порт 443 или другие <1000, выставить специальную опцию на бинарник/процесс.

На этом установка XRay закончена, дальнейшие действия будут одинаковы и при ручной настройке, и при использовании скрипта.

Нам будут нужны TLS-сертификаты.

Устанавливаем certbot и запрашиваем сертификат для нашего домена (например, example.com):

```
$ apt install certbot
$ certbot certonly --standalone --preferred-challenges http -d example.com
```

Если вам нужно иметь два домена или домен и поддомен (например, один будет доступен напрямую, другой через CDN), то можно указать ещё один аргумент -d в этой команде и у вас будет сертификат сразу для двух доменов. А ещё оно поддерживает wildcards.

Certbot спросит ваш емайл на всякий случай, спросит согласны ли вы с правилами, запросит сертификат от LetsEncrypt, положит его в папочку /etc/letsencrypt и создаст правило, чтобы он обновлялся каждые 3 месяца. При каждом обновлении сертификата нужно перезапускать XRay-сервер, давайте попросим certbot делать это автоматически:

```
$ nano /etc/letsencrypt/renewal/example.com.conf
```

и там в конец добавим строку

```
renew_hook = systemctl reload xray
```

Теперь переходим к самому интересному. Создаем и редактируем конфиг:

```
$ nano /opt/xray/config.json # или в /usr/local/etc/xray/ в случае использования скрипта
```

```
{
  "log": {
    "loglevel": "info"
  },
  "routing": {
    "rules": [],
    "domainStrategy": "AsIs"
  },
  "inbounds": [
    {
      "port": 23,
      "tag": "ss",
      "protocol": "shadowsocks",
      "settings": {
        "method": "2022-blake3-aes-128-gcm",
        "password": "aaaaaaaaaaaaaaaaabbbbbbbbbbbbbbbb",
        "network": "tcp,udp"
      }
    },
    {
      "port": 443,
      "protocol": "vless",
      "tag": "vless_tls",
      "settings": {
        "clients": [
          {
            "id": "7957c33c-d9ca-11ed-afa1-0242ac120002",
            "email": "user1@myserver",
            "flow": "xtls-rprx-vision"
          }
        ],
        "decryption": "none",
        "fallbacks": [
          {
            "path": "/myverysecretpath",
            "dest": "@vless-ws"
          },
          {
            "dest": "8080"
          }
        ]
      },
      "streamSettings": {
        "network": "tcp",
        "security": "tls",
        "tlsSettings": {
          "alpn": [
            "http/1.1",
            "h2"
          ],
          "certificates": [
            {
              "certificateFile":
"/etc/letsencrypt/live/example.com/fullchain.pem",
              "keyFile": "/etc/letsencrypt/live/example.com/privkey.pem"
            }
```

```
          ]
        }
      },
      "sniffing": {
        "enabled": true,
        "destOverride": [
          "http",
          "tls"
        ]
      }
    },
    {
      "listen": "@vless-ws",
      "protocol": "vless",
      "tag": "vless_ws",
      "settings": {
        "clients": [
          {
            "id": "7957c33c-d9ca-11ed-afa1-0242ac120002",
            "email": "user2@myserver"
          }
        ],
        "decryption": "none"
      },
      "streamSettings": {
        "network": "ws",
        "security": "none",
        "wsSettings": {
          "path": "/myverysecretpath"
        }
      }
    }
  ],
  "outbounds": [
    {
      "protocol": "freedom",
      "tag": "direct"
    },
    {
      "protocol": "blackhole",
      "tag": "block"
    }
  ]
}
```

На что обратить внимание. В inbounds мы задаем правила обработки входящих подключений - первым идет Shadowsocks-2022 на 23 порту (можете использовать любой другой порт, само собой). О том, что эта версия протокола именно 2022 говорит method "2022-blake3-aes-128-gcm". Ключ - любой в шестнадцатеричной форме, его длина зависит от типа шифра, в примере 128-битный шифр, если используете 256-битный, то ключ, соответственно, должен быть в два раза длиннее.

Дальше идет VLESS через TLS, стандартный порт 443. В секции "clients" задаются пользователи-клиенты, в примере он только один. ID клиента можно сгенерировать любым онлайновым UUID-генератором. Также для юзера задается опция "flow" со значением "xtls-rprx-vision", означающая что этот пользователь будет подключаться с использованием XTLS-Vision. В настройках "streamSettings" вы можете увидеть пути к сертификатам, которые мы запросили у LetsEncrypt, в пути должен быть файл соответствующий вашему домену. В "fallbacks" задаются правила о том, что делать, если юзер был не опознан, либо подключение производится не через чистый VLESS-протокол: если мы видим HTTP-запрос с URI /myverysecretpath, то передаем подключение на обработчик vless-ws, для всего остального - на порт 8080, где у нас будет висеть веб-сервер с фейковым (или даже настоящим) веб-сайтом.

И наконец, третим идет вариант VLESS через Websocket, на том же 443 порту. Таким образом, например, можно подключаться к серверу не напрямую, а через CDN, что поможет если ваш сервер вдруг заблокировали цензоры или если вы подключаетесь через строгий корпоративный прокси. Настройка его почти аналогична предыдущему пункту, UUID пользователя там указан тот же самый, единственные различие - нет опции "xtls-rprx-vision", потому что через CDN она работать не будет, и есть секция "wsSettings", где указан тот же секретный путь на сервере /myverysecretpath что и в fallbacks.
См. также: Особенности проксирования через CDN/Websocket/gRPC для обхода блокировок

В комментариях к предыдущей статье упоминали, что websocket-транспорт не всегда работает надежно и эффективно, а еще при очень больших объемах передаваемого трафика Cloudflare может обидиться и начать просить перейти на платный тариф. Поэтому вместо websocket некоторые советуют использовать gRPC-транспорт. Я пробовал, и у меня не получилось нормально настроить fallback на gRPC. В комментарии к статье хабраюзер @s7eepz приложил пример настройки fallback'а на gRPC через Nginx - но важный момент, Nginx должен быть собран с gRPC-модулем. В Debian/Ubuntu и в официальных репозиториях от разработчиков он собран без него.

И как вы могли заметить, в конфиге упомянут порт 8080 для fallback. На нем у нас должен слушать веб-сервер с сайтом для маскировки. Самый просто вариант это сделать - поставить позади него nginx:

```
$ apt install nginx
$ nano /etc/nginx/sites-enabled/default
$ systemctl restart nginx
```

Где /etc/nginx/sites-enabled/default в самом простом случае будет представлять собой что-то типа такого:

```
server {
        listen 127.0.0.1:8080 default_server;
        listen [::1]:8080 default_server;

        root /var/www/html;
        index index.html index.htm index.nginx-debian.html;

        server_name _;

        location / {
                try_files $uri $uri/ =404;
        }
}
```

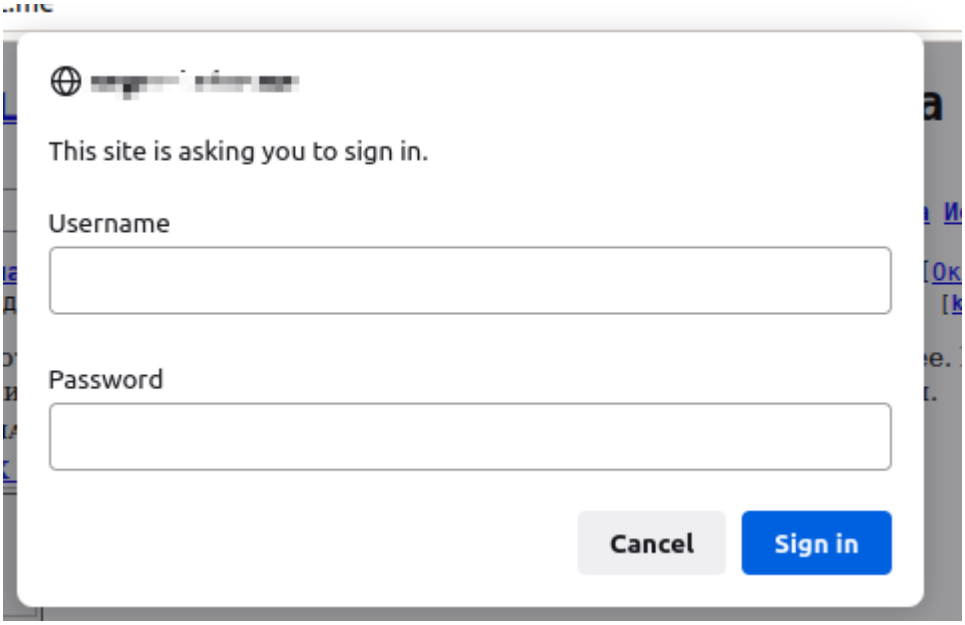
(главное изменение по сравнению с дефолтной конфигурацией - сервер слушает не на всех адресах, а только на localhost, и не на 80, а на 8080 порту).

После этого при попытке подключиться к вашему IP-адресу обычным браузером (то, что могут автоматически делать цензоры, пытаясь выявить прокси-сервера), отвечать будет Nginx, отдавая страницы лежащие в /var/www/html. По умолчанию там лежит заглушка, можно закинуть туда какие-нибудь странички и видео с котятками.

Если лень заморачиваться с поднятием фейкового сайта, есть второй вариант - пусть веб-сервер спрашивает у подключающихся логин и пароль, и отклоняет все введенные варианты:

```
location / {
                auth_basic "Administrator’s Area";
                auth_basic_user_file /etc/htpasswd;
        }
```

Сам файл /etc/htpasswd может вообще даже не существовать - нам не нужно проверять правильность пароля, мы будем отклонять все подряд, делая вид что пароль не подошел. Nginx все равно запустится даже без этого файла.



В браузере это будет выглядеть вот так

Третий вариант - переадресовывать подключения на какой-нибудь другой сайт. XRay не умеет перекидывать подключения на внешние сервера, только на локальные, поэтому тут нам опять поможет Nginx:

```
server {
    listen 127.0.0.1:8080 default_server;
    listen [::1]:8080 default_server;

    server_name _;

    location / {
         proxy_pass http://lib.ru;
    }
}
```

В результате при попытке открытия адреса прокси браузером загрузится зеркало lib.ru - замените его на какой-нибудь другой сайт. Использовать для этого какие-либо популярные или навороченный сайты явно не стоит, а вот какую-нибудь богом забытую хомпагу эпохи Web 1.0 или безымянную webftp-файлосвалку - уже можно. А чтобы некоторые тупые боты или пауки поисковых систем не нагнали вам трафика, можно добавить опции ratelimit-модуля в Nginx и ограничить скорость передачи данных с "переадресованного" сайта, например, до 1 мегабита.

Перезапускаем еще раз xray:

```
$ systemctl restart xray
```

Проверяем что все нормально запустилось:

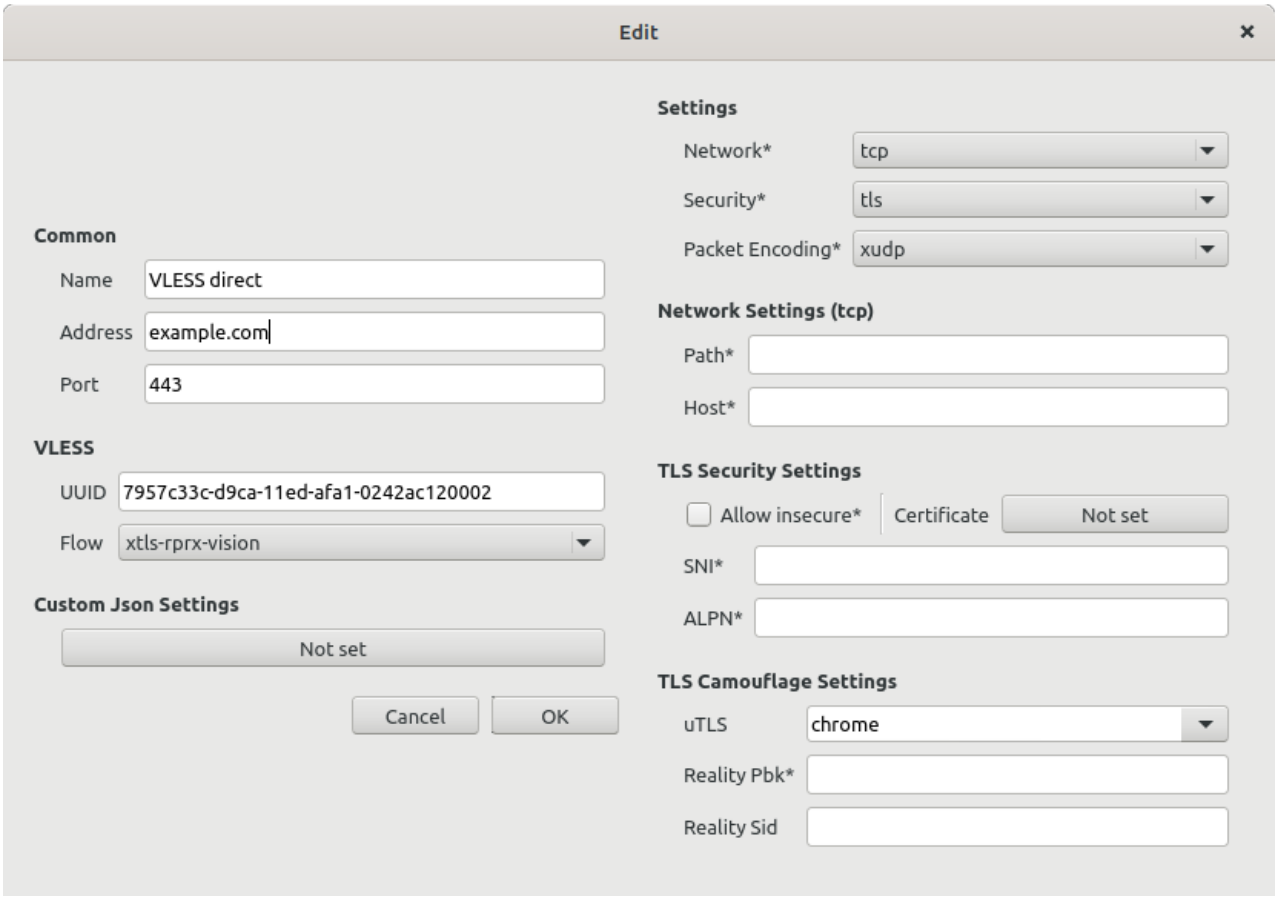
```
$ journalctl -u xray
```

Например, XRay может ругнуться что не удается распарсить JSON-файл, обычно это связано с лишними запятыми в конце {} блока, в этом случае он укажет, на какой строке ошибка. Исправляем ошибки, перезапускаем еще раз, и переходим к настройке клиентов.

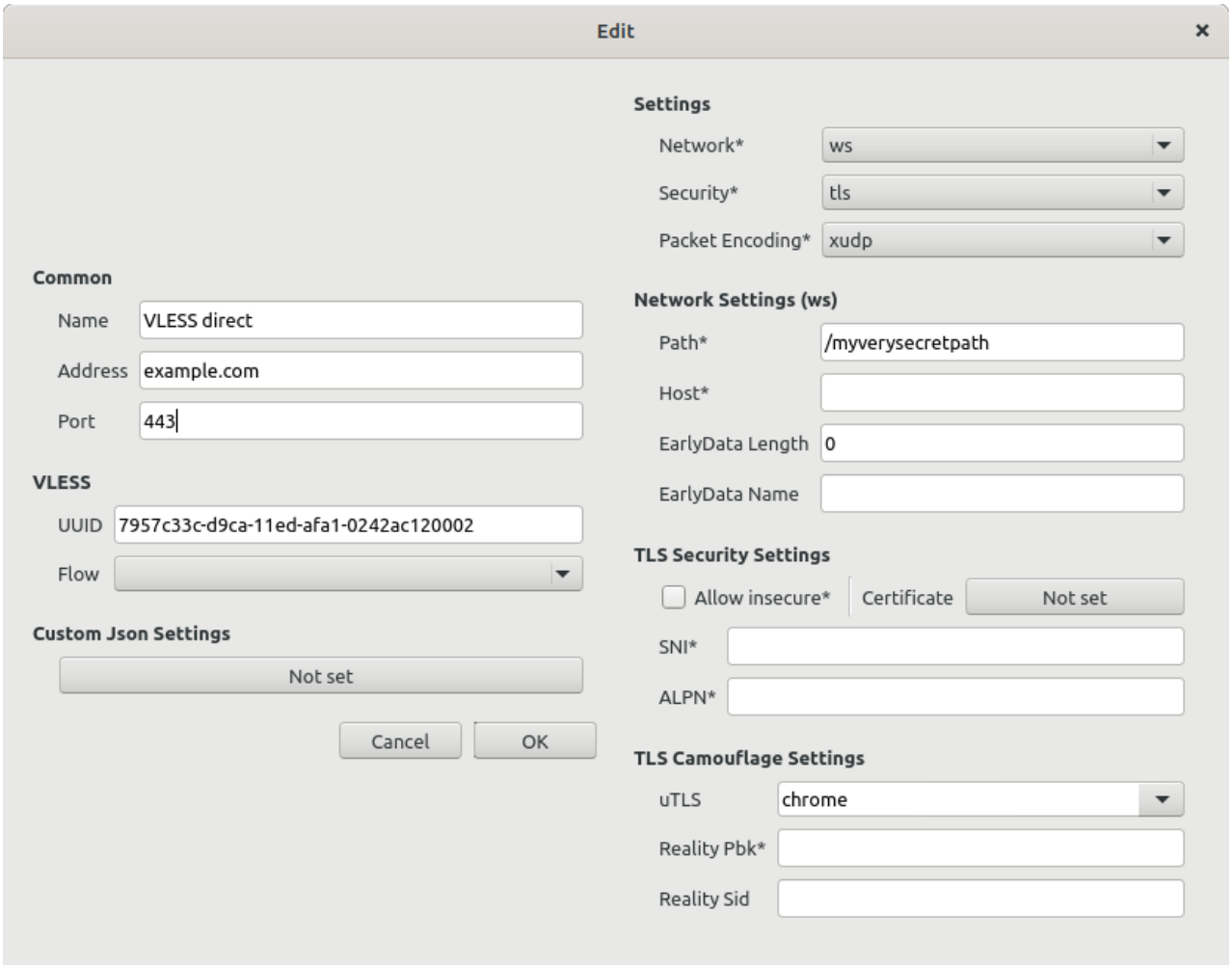
Я буду показывать на примере Nekoray/Nekobox, но абсолютно то же самое можно сделать и в другом клиенте, настройки будут одинаковые. Скачиваем Nekoray, выбираем в настройках core Sing-box (и Nekoray волшебным образом становится Nekobox).

Идем в server -> new profile, и далее заполняем поля следующим образом.

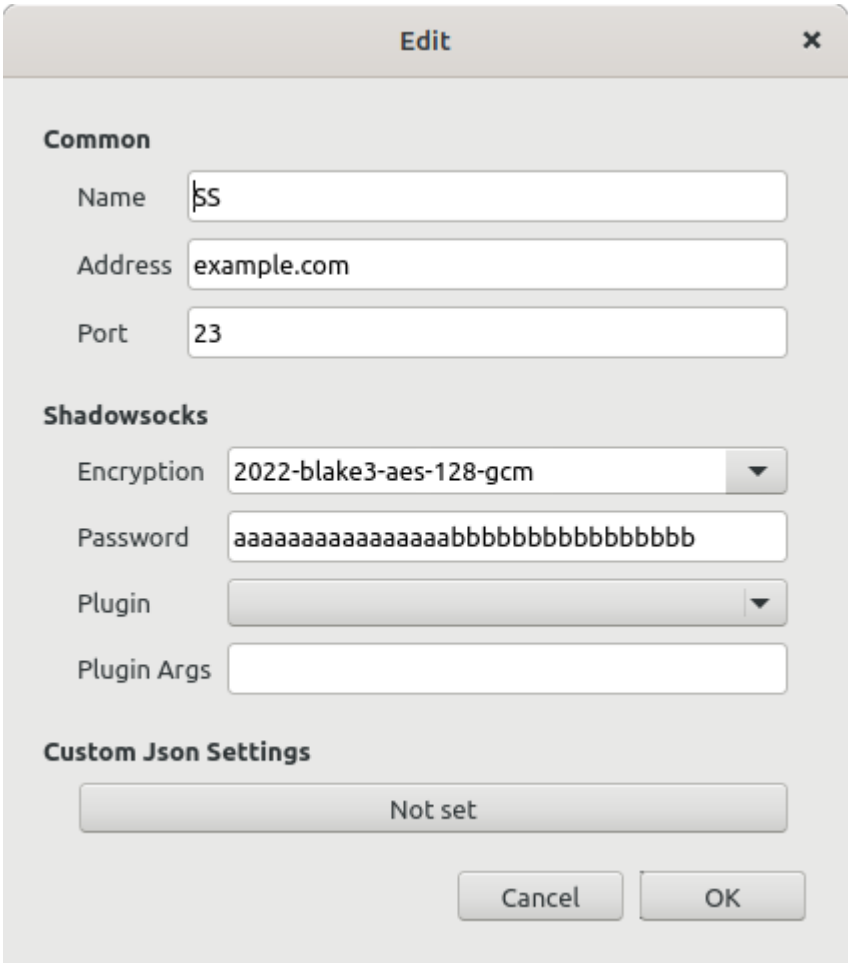
Для прямого VLESS + XTLS-Vision:



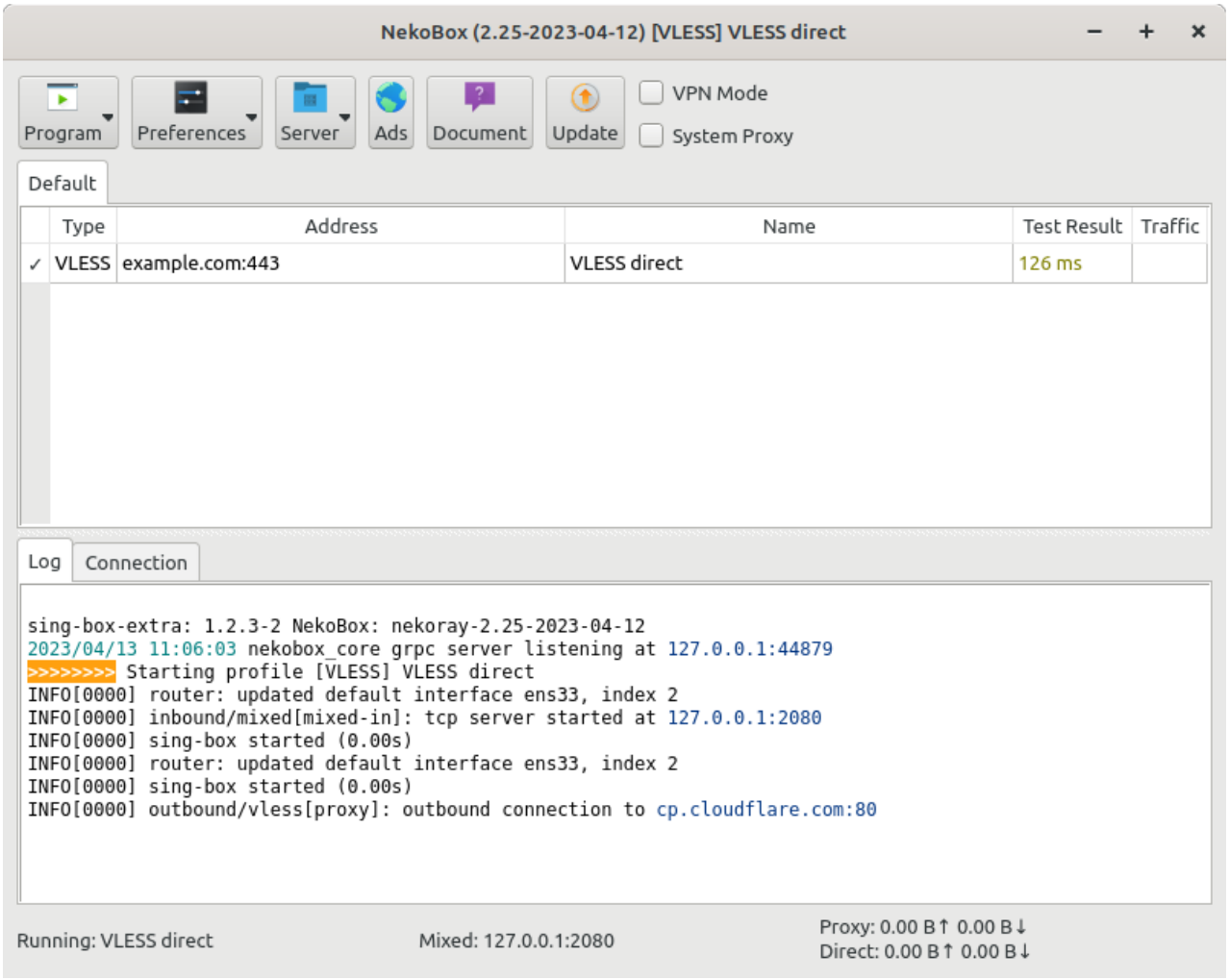
Для VLESS-over-Websockets:



Для Shadowsocks:



Выбираем нужное подключение в списке на главном окне, кликаем правой кнопкой мыши -> Current Select -> URL test, и видим в логе и в окошке, что пинг успешен:



Все. Теперь достаточно нажать сверху галочку System proxy или VPN mode, и вы попадаете в интернет через ваш новый прокси.

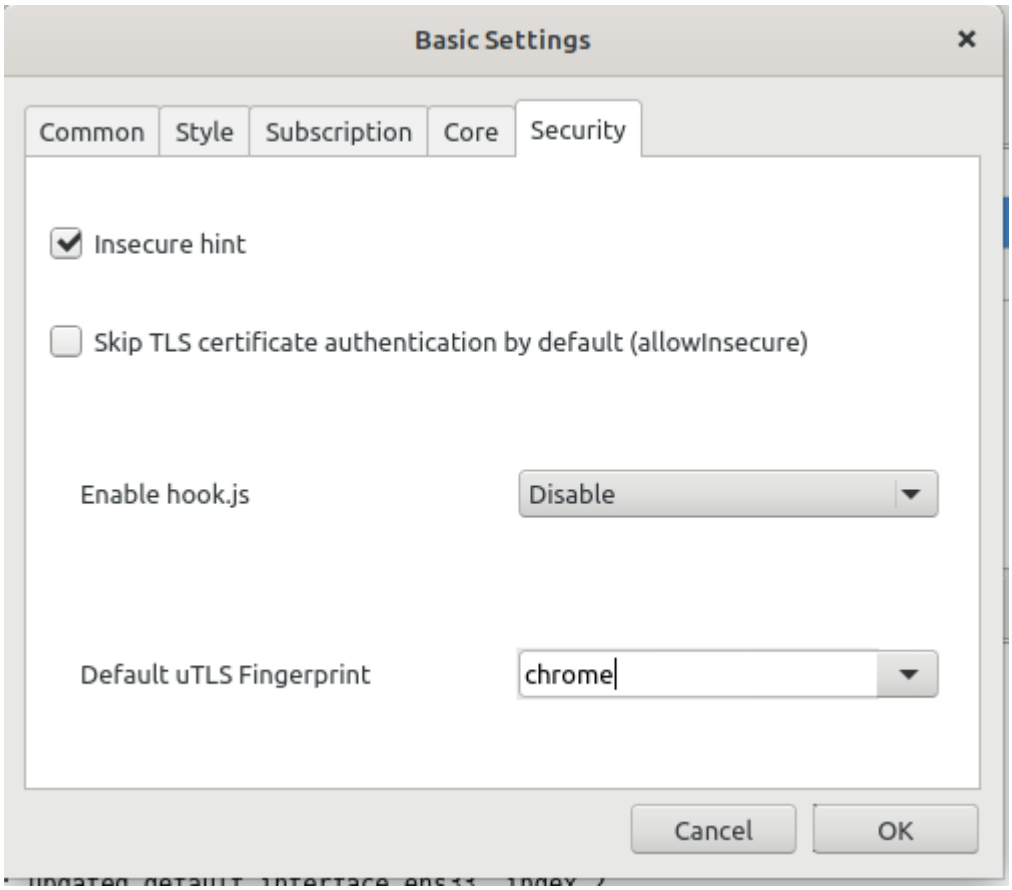
Чтобы настроить в других клиентах или на других устройствах (например, на смартфоне, или поделиться сервером с друзьями), кликаем на сервер правой кнопкой мыши, выбираем Share -> QR code and Link, и получаем ссылку, которую можно отправить кому-нибудь например через Telegram и QR-код, который можно отсканировать камерой во многих клиентах:

Соответственно, потом на мобильном устройстве в Nekobox, или в v2rayNG, или в Wings X, или в любом другом клиенте, нажимаем что-то типа "Add server" -> "Scan QR" - и все, новый сервер у вас в списке, можно подключаться.

Важно: некоторые клиенты при добавлении сервера по ссылке или QR теряют настройку uTLS, поэтому перепроверяйте все ли на месте после добавления нового сервера.

Лайфхак #1: а еще можно упороться и добавить в Nekobox еще и SSH в качестве подключения, пример конфигурации есть вот здесь (сначала надо будет подключиться родным системным ssh-клиентом, сгенерить клиентский ssh-ключи и сделать ssh-copy-id, в Windows это тоже работает).

Лайфхак #2: Чтобы не забывать ставить uTLS fingerprint для каждого подключения отдельно, его можно задать дефолтное значение в общих настройках Nekobox:



Вариант второй, полуготовый

А теперь представим, что у вас на VPS уже установлен веб-сервер с каким-нибудь сайтом, уже настроены TLS-сертификаты, и все остальное, и нужно просто аккуратно добавить прокси, желательно не ломая конфиг сервера.

Вариант раз: заиметь еще один поддомен, и разруливать TLS-подключения еще на этапе хэндшейка по SNI с помощью, например, HAProxy или ssl_preread модуля в Nginx. Тогда настройка XRay будет полностью аналогична описанному в предыдущем пункте, разве что только надо будет перевесить его с 443 на другой порт.

Вариант два: TLS-сессия будет терминироваться веб-сервером, и в случае обращения к определенному URL он будет передавать подключение на прокси. Этот вариант проще, единственное ограничение - никакого XTLS (ни Vision, ни Reality) уже не получится, и производительность будет немного ниже.

Итак, допустим, у вас настроен Nginx (или любой другой веб-сервер с каким-нибудь сайтом). Нужно средствами веб-сервера настроить переадресацию обращений к определенному урлу на прокси. Варианта два - использовать websockets (и надо не забыть передать специфичные для них хедеры), или использовать gRPC (если ваш сервер умеет его проксировать). В Nginx это будет выглядеть примерно так, для веб-сокетов:

```
location /myverysecretpath {
         proxy_pass http://127.0.0.1:8888;
         proxy_http_version 1.1;
         proxy_set_header Upgrade $http_upgrade;
         proxy_set_header Connection "Upgrade";
         proxy_set_header Host $host;
       }
```

А конфиг XRay будет таким:

```
{
  "log": {
    "loglevel": "info"
  },
  "routing": {
    "rules": [],
    "domainStrategy": "AsIs"
  },
  "inbounds": [
    {
      "port": 23,
      "tag": "ss",
      "protocol": "shadowsocks",
      "settings": {
        "method": "2022-blake3-aes-128-gcm",
        "password": "aaaaaaaaaaaaaaaaabbbbbbbbbbbbbbbb",
        "network": "tcp,udp"
      }
    },
    {
      "listen": "localhost",
      "port": 8888,
      "protocol": "vless",
      "tag": "vless_ws",
      "settings": {
        "clients": [
          {
            "id": "7957c33c-d9ca-11ed-afa1-0242ac120002",
            "email": "user1@myserver"
          }
        ],
        "decryption": "none"
      },
      "streamSettings": {
        "network": "ws",
        "security": "none",
        "wsSettings": {
          "path": "/myverysecretpath"
        }
      }
    }
  ],
  "outbounds": [
    {
      "protocol": "freedom",
      "tag": "direct"
    },
    {
      "protocol": "blackhole",
      "tag": "block"
    }
  ]
}
```

Как видно, почти то же самое, что и в предыдущем варианте, только нет inbound для "прямого" TLS-подключения, и вообще нет ничего про TLS - сервер слушает 8888 порт и сразу обрабатывает его как веб-сокет. /myverysecretpath, понятное дело, должен совпадать в конфиге веб-сервера и в конфиге прокси.

Настройки клиентов для этого варианта будут полностью аналогичны настройкам клиентов для Shadowsocks и VLESS+Websocket из прошлого пункта.

Вариант с gRPC по примеру из официальной репы с примерами у меня так и не заработал (чует мое сердце, там есть какой-то подвох с TLS и с переадресацией на него) - так что если у кого-то есть рабочие конфиги для XRay и Nginx с gRPC, делитесь в комментариях.

Вариант третий для самых ленивых (Websockets-only)

```
$ apt install docker.io docker-compose
$ mkdir /etc/xray/
$ nano /etc/xray/config.json
$ nano /etc/xray/Caddyfile
$ nano docker-compose.yml
```

- ▶ /etc/xray/config.json:
- ▶ /etc/xray/Caddyfile
- ▶ docker-compose.yml

```
$ docker-compose up -d
```

Тут в качестве веб-сервера используется Caddy, он же сам запрашивает и обновляет TLS-сертификаты (certbot не нужен). IPv6 не будет, но все остальное в принципе работает - опять же, только WS, и никакого XTLS. Lazydocker вам в помощь.

Нюансы и мудрости

На сегодняшний день связка VLESS+XTLS-Vision является самой проверенной и устойчивой к блокировкам. Однако нужно иметь в виду еще пару вещей:

1. Обязательно используйте uTLS на клиентах, выставляя правильный TLS fingerprint. Клиенты, которые не умеют в uTLS лучше не использовать;
2. Обязательно поднимите фейковый веб-сайт или настройте fallback-переадресацию на какой-нибудь левый адрес;
3. С uTLS связан интересный баг: если при использовании XTLS-Vision вы почему-то не можете подключиться, в логах сервера видна ошибка типа "failed to use xtls-rprx-vision, found outer tls version 771", попробуйте сменить версию uTLS. У меня, например, при выборе "android" клиент не подключается, а при выборе "chrome" все окей;

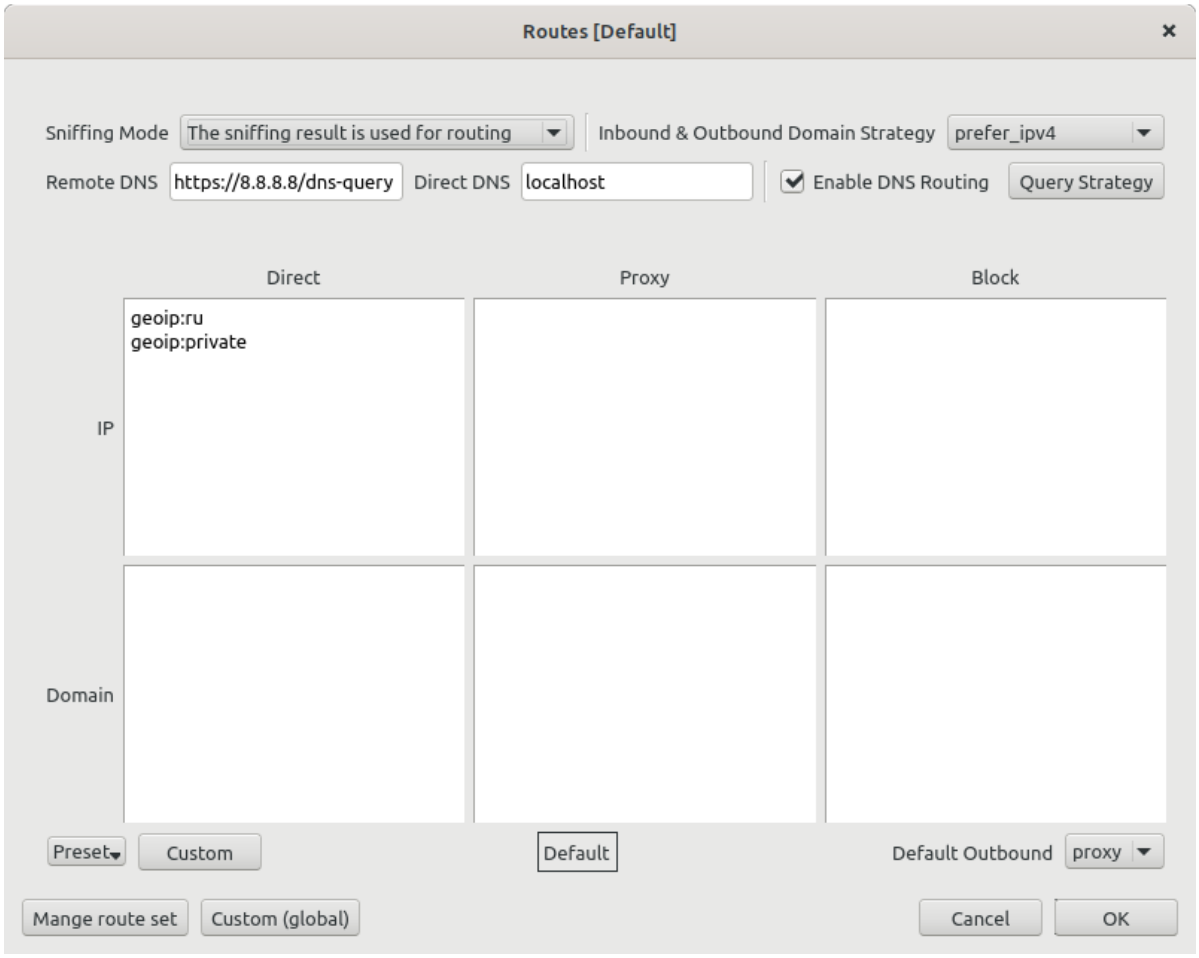
4. С XTLS лучше, чем без него;

5. Во время отладки конфигурации в случае проблем с TLS может помочь опция "allowInsecure" на клиенте;

6. Очень рекомендуется настраивать на клиентах правила маршрутизации, чтобы трафик до .ru-доменов и хостов с российскими IP шел напрямую, а не через прокси (в клиентах для такого поставляется GeoIP база данных).

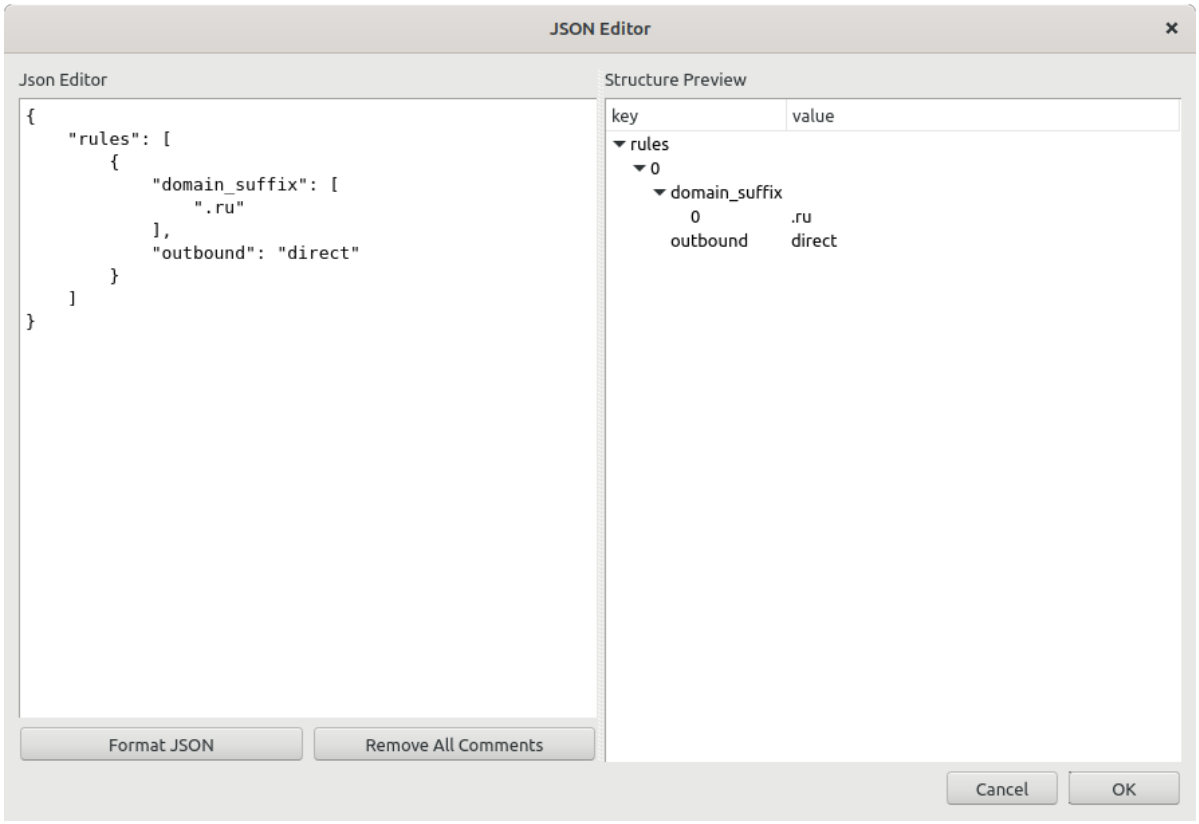
   GeoIP из Nekobox (да и других клиентов) знает российские диапазоны, а вот GeoSite - уже нет, увы.

   В итоге у меня работает вот такая настройка, GeoIP активируется стандартным образом в Nekobox:



   правила по суффиксам Nekobox не умеет, но их умеет sing-box в его основе, поэтому жмем "Custom" и прописываем

   ▶ вот такое



После этого весь трафик до .ru-доменов и российских IP-адресов будет идти напрямую.
Если нужно заблокировать полностью - то в первом окне вместо Direct вставить в Block, а в JSON-коде исправить direct на block.

Ещё можно иметь два сервера (low-end сервер в РФ можно арендовать рублей так за 60), и приняв подключение передавать его на следущий сервер, указав в outboundTag не freedom и не block, а тег соответствующего outbound'a (XRay может работать сразу и как сервер, и как клиент, не забываем).

7. При проксировании на Nginx или любой другой сервер, так же хорошей практикой считается проксировать HTTP/1.1 и HTTP/2 отдельно.
   В конфиге Nginx для этого нужно что-то типа такого:

```
listen 127.0.0.1:8888;
listen 127.0.0.1:8889 http2;
```

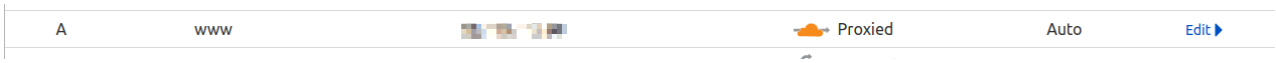
А в конфиге XRay:

```
"fallbacks": [
                {
                    "dest": "8888"
                },
                {
                    "alpn": "h2",
                    "dest": "8889"
                }
            ]
```

А что там с CDN?

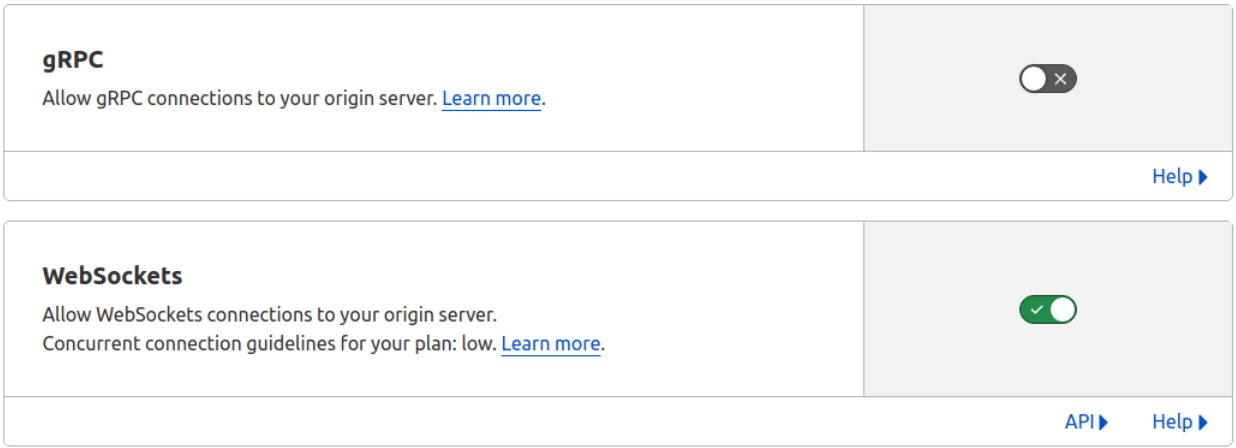
Пока что известно две CDN, которые позволяют на бесплатных аккаунтах работать с подобным: Cloudflare позволяет проксировать Websocket и gRPC, GCore позволяет проксировать Websocket (насчет gRPC не знаю, не проверял). Про Cloudflare говорят, что при проксировании очень больших объемов через вебсокеты они могут попросить перейти на платный тариф, про gRPC такого не написано.

Для работы через CDN нужен будет уже полноценный домен (не DynDNS), который можно делегировать на NS CDN-сети и управлять им там. Дальше нужно включить проксирование для конкретного домена:



Лайфхак: если у вас дешевый IPv6-only сервер, вы можете указать для него только AAAA-запись (IPv6), и Cloudflare все равно позволит вам подключаться к нему по IPv4 через свою сеть. *Смекалочка*.

Ну и не забыть отдельно включить в настройках проксирование WS и gRPC:



Подробнее: Особенности проксирования через CDN/Websocket/gRPC для обхода блокировок

А что с XTLS-Reality?

Технология многообещающая, уже даже во многих клиентах поддерживаемая, но с ней надо разбираться отдельно, и ее настройка - тоже разговор отдельный. Кто уже смог и осилил - пишите в комментарии, а лучше вообще еще одну статью. Источник

вдохновения и примеры конфигов с XTLS-Reality можно найти здесь.

А что с Sing-box?

Sing-box - активно развивающийся и тоже многообещающий клиент и сервер, и он может использоваться вместо XRay, поскольку тоже поддерживает Shadowsocks-2022, VLESS, Trojan, XTLS-Vision и XTLS-Reality, а еще умеет работать с Hysteria, Naiveproxy, и всякое другое.

Официальный сайт
Github
Документация по настройке
Разработчики реорганизуют репу, поэтому переход по ссылкам в документации может выдавать 404 ошибку — без паники, смотрим название файла, и находим правильный путь в гите по названию, дальше никаких проблем.
Как и XRay, Sing-box умеет делать fallback'и, только здесь в секции «listen» оно называется «detour», и значением этого параметра должен быть «tag» другого inbound'a.
Сайт со скриптами автоустановки и примерами конфигураций

А можно проще, и чтобы все и сразу?

Для Xray и сотоварищей существует много разных user-friendly серверных морд с простой установкой.

Есть Marzban, например - его тоже можно ставить через Docker, он включает в себя XRay, обещает красивый интерфейс для настройки и управления пользователями и даже встроенный Telegram-бот.

Ещё есть Libertea, Hiddify (про него сказано что он умеет Reality), и разные форки X-UI, где обещают все то же самое.

Но тестировать и сравнивать их у меня пока времени и терпения нет :)

На этом всё. Удачи вам в нелегком деле настройки всего этого дела, и да прибудет с вами сила.

Следущая статья серии:
«Bleeding-edge обход блокировок: настраиваем сервер и клиент XRay с XTLS-Reality быстро и просто»

Предыдущие статьи серии:

«Современные технологии обхода блокировок: V2Ray, XRay, XTLS, Hysteria и все-все-все»

«Программы-клиенты для протоколов недетектируемого обхода блокировок сайтов: V2Ray/XRay, Clash, Sing-Box, и другие»

Если вы хотите сказать спасибо автору — сделайте пожертвование в один из благотворительных фондов: "Подари жизнь", "Дом с маяком", "Антон тут рядом".