

[КАК СТАТЬ АВТОРОМ](#)[Питчи недели аналитиков](#)[Студента не уволишь: чем...](#)**MiraclePtr**

вчера в 21:33

Особенности проксирования через CDN/Websocket/gRPC для обхода блокировок

Средний

15 мин

7.4K

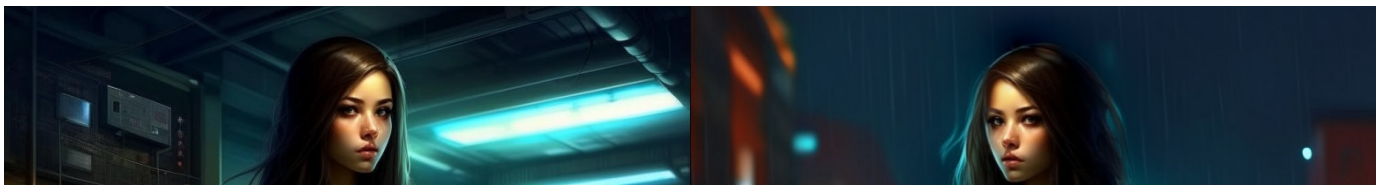
Информационная безопасность*, Системное администрирование*, Nginx*, Сетевые технологии*, IPv6*

[Тutorial](#)

Эта статья — заключительная (наконец-то!) из моего огромного цикла про недетектируемые инструменты для обхода блокировок. В предыдущих публикациях я упоминал, что клиенты и серверы XRay (форк V2Ray) и Sing-box при использовании протоколов VLESS/VMess/Trojan могут работать через веб-сокеты и gRPC, что позволяет подключаться к даже заблокированным Роскомнадзором прокси-серверам через CDN (content delivery или content distribution network) и дает дополнительные преимущества. Сегодня мы поговорим об этом поподробнее.

Для чего?

1. Если ваш прокси-сервер попал под козловую блокировку или стал недоступен по еще какой-то причине, вы по-прежнему можете достучаться до него через CDN. Вероятность блокировки целого Cloudflare или Gcore CDN гораздо ниже, потому что на них сидит чуть ли не половина всех сайтов Интернета. Даже в Туркменистане, известном своими драконовскими мерами, до недавнего времени люди умудрялись находить незабаненные адреса балансировщиков CF.
2. Cloudflare умеет проксировать IPv4-запросы на IPv6-адреса. Таким образом, для того, чтобы запустить свой прокси, достаточно купить копеечный IPv6-only-сервер (с выходом в IPv4-сеть через NAT), который можно найти меньше чем за доллар в месяц.
3. Проксирование через веб-сокеты может оказаться эффективным для пробивания через строгие корпоративные системы, которые анализируют весь передаваемый трафик с man-in-the-middle расшифровкой с подменой сертификата.



+58

175



25



Нейрокартинка для отвлечения внимания

Протоколы

Первый, самый простой и самый старый протокол — **Websocket**. Идея простая — клиент отправляет серверу HTTP-запрос с просьбой переключить подключение в режим веб-сокетов, и если сервер не возражает, то HTTP-подключение становится обычной «трубой», по которой можно слать любые данные туда-сюда. Формат и порядок передаваемых данных стандартом не регламентирован и может быть любой, поэтому CDN в них не вмешивается, а просто пересылает данные с сервера к клиенту и обратно — самое то, для того чтобы использовать подключение как прокси.

Недостатком использования веб-сокетов является более долгое время установления подключения (кроме TLS-хендшейка еще должен пройти вебсокет-хендшейк), и также тот факт, что при большом желании вебсокет-хендшейк можно детектировать по передаваемым объемам данных в начале каждого соединения.

Разработчики проксей придумали решение этой проблемы под названием «early data». Суть его в том, что вместо того, чтобы сначала попросить сервер перейти в режим websockets, а потом слать запросы на прокси, websocket-хендшейк уже будет содержать какую-то часть данных, которую мы отправляем на прокси-сервер — это и сокращает время установления соединения, и рандомизирует размеры пакетов, усложняя детектирование. Если вы используете клиент на базе XRay, то для использования early data нужно добавить «?ed=XXX» в конец пути вебсокет-адреса, где XXX — максимально допустимый размер данных, которые могут содержаться в первом запросе (в байтах). У меня без проблем работало с " ?ed=2048" , при каких-либо ошибках можно попробовать уменьшить значение. XRay всегда использует для этих данных заголовок «Sec-Websocket-Protocol», Sing-box же позволяет использовать любой заголовок, поэтому если вы подключаетесь клиентом на базе Sing-Box (например Nekobox) к серверу на базе XRay, то нужно явно указывать имя хедера «Sec-Websocket-Protocol» в настройках подключения — чуть позже увидите сами.

Другой протокол — это **gRPC**. Это протокол для межпроцессного и межсервисного взаимодействия от Google, который в качестве транспорта тоже использует HTTP. Он

поддерживается не всеми CDN (об этом чуть позже), зато его использование сложнее детектировать, а еще с его помощью можно организовать мультиплексирование (использовать одно и то же TCP-подключение для множества сессий с прокси). Где-то тут в комментариях писали, что gRPC более производительен, чем веб-сокеты, автор Sing-box наоборот жалуется, что у gRPC-клиента «poor performance», я же особой разницы не заметил.

CDN

По идее, можно использовать любые CDN, которые поддерживают проксирование websocket и/или gRPC. Среди популярных CDN, которые умеют такое на *бесплатных* тарифах, известны **Cloudflare** и **Gcore**. Расскажу о них по-подробнее.

Cloudflare — наверное, самая крупная и известная сеть доставки контента в мире (хотя, может быть Akamai и больше, не проверял). На бесплатном тарифе поддерживает как websocket'ы, так и gRPC. Про веб-сокеты в хелпе сказано, что если вы будете гонять через них очень-очень-много данных на бесплатном тарифе, то CF может связаться с вами и попросить начать платить. Я пробовал гонять много данных, пока не попросили. Про gRPC такого требования нет. Еще одна приятная особенность Cloudflare, как я уже говорил выше — это то, что можно иметь сервер, обладающий только IPv6-адресом, и CF будет принимать IPv4-подключения и перенаправлять их на IPv6-адрес.

Одно *но* — чтобы гонять через нее трафик, нужно, чтобы у вас был домен, и чтобы этот домен был делегирован на ее нейм-сервера (NS). Регистрируемся в Cloudflare, добавляем там свой домен — CF скажет вам, на какие именно NS его надо делегировать, и это нужно сделать (если не знаете как, запросите инструкции у вашего регистратора доменов или хостинг-провайдера). Либо же можно сразу купить домен у Cloudflare, цены у них нормальные, но платить, понятное дело, можно только с иностранных банковских карт.




Дальше всё просто:

Идем в **DNS** → **Records**, нажимаем «**Add record**», и добавляем запись типа A (если у вас IPv4-адрес) или AAAA (если у вас IPv6), указав в Name поддомен или @ для корневого домена, IP-адрес, и не забыв отметить галочку «**Proxied**»:

[name] points to [IPv4 address] and has its traffic proxied through Cloudflare.

Type	Name (required)	IPv4 address (required)	Proxy status	TTL
A	<input type="text"/>	<input type="text"/>	<input checked="" type="checkbox"/> Proxied	Auto
	<small>Use @ for root</small>			

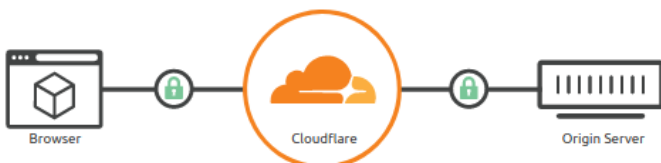
Также нужно не забыть зайти в настройки "**Network**" для вашего домена в панели CF, и убедиться, что все три самые важные галочки включены:

IPv6 Compatibility Enable IPv6 support and gateway. This setting was last changed a year ago	
API Help	
gRPC Allow gRPC connections to your origin server. Learn more.	
Help	
WebSockets Allow WebSockets connections to your origin server. Concurrent connection guidelines for your plan: low. Learn more.	
API Help	

Конечному клиенту (браузеру или прокси) Cloudflare демонстрирует свой TLS-сертификат. Подключение между серверами Cloudflare и вашим сервером по-хорошему тоже должно быть защищено TLS — для этого не обязательно запрашивать домен через Let's Encrypt, можно использовать самоподписанный сертификат (инструкция будет дальше), либо же сгенерировать «внутренний» сертификат от Cloudflare — он подписан другой цепочкой, поэтому в браузере работать нормально не будет, но для связи между балансировщиками CF и вашим хостом подойдет идеально. Настраивается это в разделе **SSL/TLS** → **Overview**: режим **Full** будет работать с самоподписанным сертификатов, а режим **Full (Strict)** будет требовать наличие «внутреннего» сертификата Cloudflare на вашем сервере:

✓ Your SSL/TLS encryption mode is Full

This setting was last changed a month ago

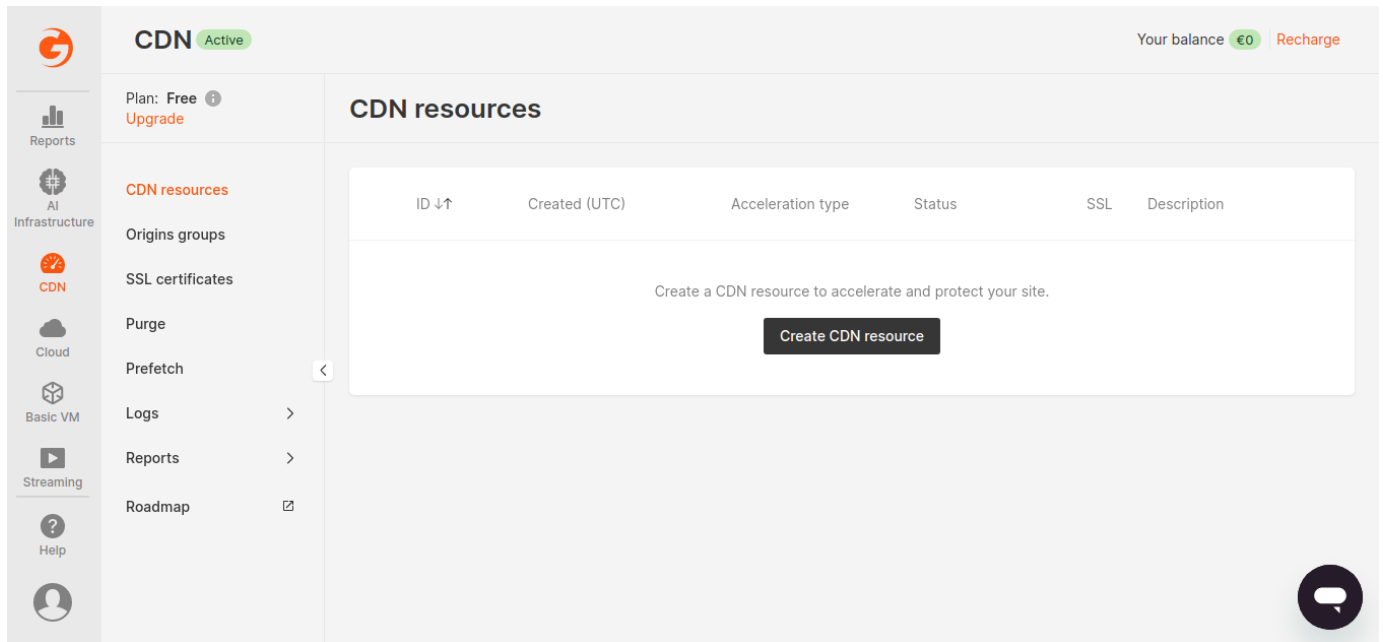


- ☐ Off (not secure) ⓘ
No encryption applied
- ☐ Flexible
Encrypts traffic between the browser and Cloudflare
- ☒ **Full**
Encrypts end-to-end, using a self signed certificate on the server
- ☐ Full (strict)
Encrypts end-to-end, but requires a trusted CA or Cloudflare Origin CA certificate on the server

► [как сгенерировать "внутренний" сертификат](#)

Дальше у нас по списку идет **Gcore**. Масштабы у них не такие огромные, как у Cloudflare, поддерживаются только веб-сокеты, gRPC нет, IPv4-to-IPv6 тоже нет. Но у gCore есть огромное преимущество — для работы через него не обязательно делегировать на него домен, достаточно просто CNAME-записи — в теории, можно использовать даже DynDNS-сервисы с бесплатными доменами, которые позволяют указывать CNAME.

Настраивается все следующим образом. Регистрируетесь в Gcore (Gcore Edge Solutions / Gcore Platform), выбираете бесплатный тариф. В панели выбираете слева «CDN», и нажимаете «Create CDN Resource»:



А дальше надо быть внимательным. Предлагаются два варианта: *accelerate entire website*, и *accelerate only static content*. Первый вариант, как и в случае с Cloudflare, требует делегирования вашего домена на неймсервера CDN. Поэтому мы выбираем второй вариант, а именно, **accelerate only static content**, хоть это и немного контринтуитивно (веб-сокеты на static content все-таки мало тянут, не правда ли?):

Acceleration and protection type

Confirm

Choose acceleration and protection type for your project

☐ Accelerate and protect entire site New
Codeless acceleration and network layer protection of the entire site with CDN and DNS services.

☒ Accelerate and protect only static assets
Acceleration and network layer protection of the static assets of your website with CDN service.
Changes in the code are required.

Нажимаем Confirm, идем на следующую страницу, и начинаем заполнять:

☒ Specify content origin ☐ Select origins group

Type of origin authentication ?

None

Origin Source ☒ Use default port

http(s):// | example.com

Specify the origin or origins group that CDN uses to pull the content from. For example, IP address or site URL without http:// or https://.

+ Add origin

Origin source — тут нужно ввести IP-адрес вашего сервера с префиксом `https://`, например «`https://12.34.56.78`». Gcore не умеет проксировать IPv6 на IPv4 (у меня отказалась, по крайней мере), поэтому тут нужен уже IPv4-адрес.

Custom domain (to create a CNAME record)

Create a domain name for content delivery through a CDN. Add it in your DNS settings.

[Read more on CNAME setup ↗](#).

http(s):// | cdn.example.com



SSL



Enable HTTPS

Use SSL certificate.

☒ Get free Let's Encrypt certificate ☐ Add or select your own SSL certificate

We will start issuing a Let's Encrypt certificate when the CDN resource is created. You will need to point your custom domain to the CDN to get the certificate.

Custom domain — тут укажите ваш домен, который вы будете использовать. Не забудьте включить галочку «**Enable HTTPS**» и выбрать «Get free Let's Encrypt certificate». Нажимаем Confirm чтобы идти дальше. После этого Gcore скажет вам, какое именно значение надо прописать в поле CNAME для вашего домена — это будет какой-то адрес, скорее всего заканчивающийся на *.gcdn.co:

Set up your DNS

Confirm

1. Log in to your DNS hosting provider.

2. Add a CNAME record pointing to a.gcdn.co

Go to your DNS hosting settings and create the following CNAME record CNAME a.gcdn.co. [Read more on CNAME setup ↗](#).

An example in BIND zone file format below:

```
$ORIGIN .  
 CNAME a.gcdn.co.
```

На следующей странице Gcore попытается нам помочь настроить движок веб-сайта, выбираем I don't have CMS и идем дальше:

Integrate with your project

PreviousConfirm

Integrate with your project

I have CMSI don't have CMS

Replace an origin domain with the CNAME in a path to the static files.

Replace:

```

```

With:

```
;
```

It will also work for all the other types of static files (CSS, JS and SWF).

В самом конце не забываем включить поддержку Websocket'ов:

Quick options setup

PreviousConfirm

Optimize performance ⓘ

GZip compression

☐ Enable GZip compression

Content will be compressed by CDN servers using the Gzip compression algorithm before delivery. Then only uncompressed content will be requested from the origin server.

We support Gzip for the following data types: application/javascript; application/json; application/x-javascript; application/xml; application/xml+rss; application/wasm; text/css; text/html; text/javascript; text/plain; text/xml; image/svg+xml.

WebSockets

☒ Enable WebSockets

Allow WebSockets connections to an origin server.

Ииии... готово!

После создания ресурса, нужно еще поменять режим на HTTPS-only:

Настройка серверов и клиентов

Если вы читаете эту статью, то вероятно настраивали сервер XRay по одному из моих мануалов: [Обход блокировок: настройка сервера XRay для Shadowsocks-2022 и VLESS с XTLS-Vision, Websockets и фейковым веб-сайтом](#), [Bleeding-edge обход блокировок с полной маскировкой: настраиваем сервер и клиент XRay с XTLS-Reality быстро и просто](#), [3X-UI: Shadowsocks-2022 & XRay \(XTLS\) сервер с простой настройкой и приятным интерфейсом](#).

В первом случае, у вас уже настроен XRay-сервер, который принимает подключения на 443 порт по TLS, у него есть сертификаты, и при этом не используется XTLS-Reality, домен только свой. Во втором случае используется XTLS-Reality, то есть сервер маскируется под какой-то популярный сайт, принимая подключения с SNI его домена, и отдавая его подлинный TLS-сертификат. В третьем случае возможно и то и то, смотря как настраивали:)

А если вы планируете использовать Cloudflare, и у вашего сервера есть IPv6-адрес, то возможно вообще настроить websocket/gRPC, не трогая основную инсталляцию и не меняя ничего не ней.

И сейчас мы разберем, что же нужно сделать, чтобы иметь возможность работать через CDN и websockets/gRPC для всех этих вариантов.

Во всех случаях нам понадобится установленный на сервер **Nginx**. Поскольку суть проксирования через CDN — в маскировке под легитимный HTTPS-трафик, то нам нужен будет веб-сервер, чтобы хостить какой-нибудь безобидный сайтик. Плюс к этому, почему-то в XRay нельзя задать gRPC-транспорт как fallback, и здесь нам тоже поможет Nginx. Для работы на одном сервере и WS/gRPC, и XTLS-Reality нужен будет SNI-прокси, и Nginx тоже умеет работать в этом режиме. Поэтому устанавливаем Nginx.

Обратите внимание, если у вас на сервере стоит Debian или Ubuntu, то нужно устанавливать не просто пакет `nginx`, а `nginx-full`, потому что нам потребуются

специфические модули Nginx, которые не входят в стандартную поставку (как оно в других дистрибутивах — не знаю, напишите в комментарии).

Также нам понадобится TLS-сертификат, если у вас его еще нет. Как я уже говорил, при работе через CDN хватит самоподписанного сертификата (все равно его не будет видно снаружи, он используется только для связи между фронтендом CDN и вашим сервером), либо «внутреннего» сертификата от Cloudflare. Инструкцию по получению сертификата от Cloudflare я уже описывал чуть выше, а самоподписанный сертификат можно сгенерировать одной командой вот так:

```
openssl req -x509 -newkey rsa:4096 -nodes -sha256 -keyout /etc/ssl/private/ss
```

Запомните пути к файлам, они вам потом пригодятся.

Далее, для всех трех вариантов нужно будет добавить websocket- и grpc-inbound'ы в конфиг XRay, если их там еще нет. Для всех трех вариантов они будут выглядеть одинаково:

```
{
  "listen": "127.0.0.1",
  "port": 8888,
  "protocol": "vless",
  "tag": "grpc",
  "settings": {
    "clients": [
      {
        "id": "_UUID_вашего_пользователя"
      }
    ],
    "decryption": "none"
  },
  "streamSettings": {
    "network": "grpc",
    "grpcSettings": {
      "serviceName": "TestChatGRPC"
    }
  }
},
{
  "listen": "127.0.0.1",
  "port": 8889,
```

```
"protocol": "vless",
"tag": "ws",
"settings": {
  "clients": [
    {
      "id": "_UUID_вашего_пользователя"
    }
  ],
  "decryption": "none"
},
"streamSettings": {
  "network": "ws",
  "wsSettings": {
    "path": "TestChatWS"
  }
}
},
```

«TestChatGRPC» и «TestChatWS» должны совпадать с секретными URL'ами, которые мы потом внесем в конфиг Nginx. Лучше не использовать урлы из примера, а придумать свои. Номера портов 8888 и 8889 — тоже должны совпадать с портами, на которые проксирует Nginx (см. дальше).

Если вы используете панель **X-UI** или **3X-UI**, то там все аналогично — нужно создать новые inbounds, выбрать тип транспорта grpc или websocket, указать IP-адрес для входящих подключений 127.0.0.1, соответствующие порты и URL'ы. Websocket/gRPC на отдельном IPv6-адресе.

Обратите внимание: при подключении через Nginx или CDN, нельзя использовать flow «xtls-rprx-vision». Работать не будет.

Приводим конфиг Nginx (например, в /etc/nginx/sites-enabled/default) к такому виду:

```
server {
    listen [_тыт_ваш_IPv6_адрес]:443 ssl ipv6only=on http2 so_keepalive=on;

    server_name your_domain.com;

    # сюда можно положить какие-нибудь странички фейкового сайта, или использовать
    index index.html;
    root /var/www/html;
```

```
# путь к вашим сертификатам, самоподписанным либо от Cloudflare
ssl_certificate /etc/ssl/certs/ssl-cert-snakeoil.pem;
ssl_certificate_key /etc/ssl/private/ssl-cert-snakeoil.key;

client_header_timeout 52w;
keepalive_timeout 52w;
# замените TestChatGRPC на какую-нибудь секретную строку
location /TestChatGRPC {
    if ($content_type !~ "application/grpc") {
        return 404;
    }
    client_max_body_size 0;
    client_body_buffer_size 512k;
    grpc_set_header X-Real-IP $remote_addr;
    client_body_timeout 52w;
    grpc_read_timeout 52w;
    grpc_pass grpc://127.0.0.1:8888;
}

# аналогично замените TestChatWS на какую-нибудь другую секретную строку
location /TestChatWS {
    if ($http_upgrade != "websocket") {
        return 404;
    }
    proxy_pass http://127.0.0.1:8889;
    proxy_redirect off;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "upgrade";
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_read_timeout 52w;
}
}
```

Смотрите комментарии к приведенному конфигу — нужно будет указать IPv6-адрес вашего сервера, домен, путь к сертификатам.

Дальше, нужно поменять настройки XRay так, чтобы он прекратил слушать на всех IPv4 и IPv6-адресах, и начал слушать только на одном IPv4-адресе. В конфиге XRay для вашего VLESS-inbound на 443 порту, добавьте поле `"listen": "127.0.0.1"`, либо измените его, если оно уже есть. Обратите внимание, что нельзя указывать там «0.0.0.0», пытаюсь заставить XRay слушать на всех только IPv4-адресах — он толкует это значение своеобразно и слушает на IPv6 тоже.

Проверяем конфиг Nginx командой «`nginx -t`», если есть ошибки — исправляем, если ошибок нет, то перезапускаем сначала XRay (`systemctl restart xray` если вы ставили по моим инструкциям), и потом Nginx (`systemctl restart nginx`). Настройка сервера закончена, можно переходить к настройке клиента (будет описана дальше).

Websocket/gRPC на одном адресе с VLESS XRay на 443 порту без XTLS-Reality

Конфигурация XRay или X-UI/3X-UI для этого варианта полностью аналогична предыдущему пункту, только с небольшим дополнением. Для основного VLESS-inbound необходимо задать fallback, если он не задан, например, на 127.0.0.1:8080.

В JSON-конфиге нужно добавить в секцию «settings» этого inbound'a такое:

```
"fallbacks": [
  {
    "dest": 8080
  }
]
```

(внимательно с форматированием, если вы добавили новый параметр в конец секции, предыдущий, уже существовавший, должен иметь запятую в конце, т.к. он более не последний элемент структуры).

В интерфейсе X-UI/3X-UI тоже есть опции для задания фоллбэков. Фоллбэк должен быть только один, и URL задавать не надо — мы по умолчанию переадресуем абсолютно все не прошедшие VLESS-аутентификацию подключения на Nginx.

Далее, приводим тот же конфиг Nginx (например, `/etc/nginx/sites-enabled/default`) к такому виду — очень похоже на предыдущий пункт, но есть несколько различий:

```
server {
    listen 127.0.0.1:8080 so_keepalive=on;
```

```
# тут ваш домен
server_name your_server_name.com

# сюда накидать каких-нибудь страничек
index index.html;
root /var/www/html;

client_header_timeout 52w;
keepalive_timeout 52w;
location /TestChatGRPC {
    if ($content_type !~ "application/grpc") {
        return 404;
    }
    client_max_body_size 0;
    client_body_buffer_size 512k;
    grpc_set_header X-Real-IP $remote_addr;
    client_body_timeout 52w;
    grpc_read_timeout 52w;
    grpc_pass grpc://127.0.0.1:8888;
}

location /TestChatWS {
    if ($http_upgrade != "websocket") {
        return 404;
    }
    proxy_pass http://127.0.0.1:8889;
    proxy_redirect off;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "upgrade";
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_read_timeout 52w;
}
}
```

Основное отличие — мы слушаем на локалхосте на 8080 порту, и не используем SSL — TLS-подключение terminates силами XRay, и в nginx мы получаем уже дешифрованные данные.

Как это работает: входящие подключения от клиентов попадают на XRay, который устанавливает TLS-соединение. Если клиент аутентифицировал себя как клиент для протокола VLESS «без ничего», то он работает с прокси напрямую. Если нет (например, это браузер, или мы работаем через WS/gRPC) — подключение передается на Nginx, где в зависимости от URL клиенту отдается либо фейковый сайт, либо подключение снова отправляется на XRay, но уже на websocket- или grpc-inbound.

Проверяем конфиг Nginx командой «nginx -t», если есть ошибки — исправляем, если ошибок нет, то перезапускаем сначала XRay (systemctl restart xray если вы ставили по моим инструкциям), и потом Nginx (systemctl restart nginx). Настройка сервера закончена, можно переходить к настройке клиента (будет описана дальше).

Websocket/gRPC вместе с XTLS-Reality

Как вы уже знаете из предыдущих статей, суть XTLS-Reality в том, что прокси-сервер выдает себя за веб-сервер какого-нибудь популярного сайта, переадресовывая все подключения, не прошедшие проверку «свой-чужой» на него. Поэтому просто так использовать Websocket или gRPC не получится, это ломает всю идею XTLS-Reality. Но можно использовать хак с SNI-прокси. Обратите внимание: если у вас на сервере настроен XTLS-Reality, то через Websocket/gRPC подключаться к нему нужно *только* через CDN! Иначе для сторонних наблюдателей будет немного подозрительно, что на одном и том же IP-адресе висит сразу и какой-нибудь microsoft.com, и ваш маленький фейковый никому неизвестный веб-сайт.

В первую очередь, редактируем ваш конфиг XRay, сделаем так, чтобы его VLESS/XTLS-Vision inbound слушал на локалхосте на порту 8443:

```
"listen": "127.0.0.1",  
"port": 8443
```

Если вы используете X-UI или 3X-UI, там это точно также меняется в настройках inbound'ов.

Далее, настроим Nginx. Конфиг сайта по умолчанию будет похожий на предыдущие варианты, но с небольшими отличиями:

```
server {  
    listen 127.0.0.1:8444 http2 so_keepalive=on;  
  
    # ваш домен  
    server_name your_server_domain;
```



```
# сюда накидайте страничек
index index.html;
root /var/www/html;


# путь к сертификатам
ssl_certificate /etc/ssl/certs/ssl-cert-snakeoil.pem;
ssl_certificate_key /etc/ssl/private/ssl-cert-snakeoil.key;


client_header_timeout 52w;
keepalive_timeout 52w;
location /TestChatGRPC {
    if ($content_type !~ "application/grpc") {
        return 404;
    }
    client_max_body_size 0;
    client_body_buffer_size 512k;
    grpc_set_header X-Real-IP $remote_addr;
    client_body_timeout 52w;
    grpc_read_timeout 52w;
    grpc_pass grpc://127.0.0.1:8888;
}

location /TestChatWS {
    if ($http_upgrade != "websocket") {
        return 404;
    }
    proxy_pass http://127.0.0.1:8889;
    proxy_redirect off;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "upgrade";
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_read_timeout 52w;
}
}
```

И потом нам надо будет настроить непосредственно SNI проху, для этого можно использовать тоже Nginx. Одно но — конфиги сайтов из /etc/nginx/sites-enabled обычно

инклюдятся в секцию «http» конфига Nginx, а нам нужно настроить секцию «stream», поэтому можно добавить ее прямо в /etc/nginx/nginx.conf:

```
stream {
    map $ssl_preread_server_name $backend {
        www.microsoft.com      reality;
        your_domain_name      local;
        default                 reality;
    }

    upstream reality {
        server 127.0.0.1:8443;
    }

    upstream local {
        server 127.0.0.1:8444;
    }

    server {
        listen          443 reuseport so_keepalive=on;
        ssl_preread      on;
        proxy_pass        $backend;
    }
}
```

Тут все просто — сначала идет домен, под который вы маскируетесь с XTLS-Reality (например, `www.microsoft.com`), потом идет ваш домен. Подключения с SNI вашего домена Nginx перенаправит на 8444 порт (где обработает его как веб-сайт, или переадресуется на `websocket/grpc-inbound`'ы), а все остальное, включая фейковый домен XTLS-Reality — на 8443 порт, где слушает XRay.

Проверяем конфиг Nginx командой «`nginx -t`», если есть ошибки — исправляем, если ошибок нет, то перезапускаем сначала XRay (`systemctl restart xray` если вы ставили по моим инструкциям), и потом Nginx (`systemctl restart nginx`). Настройка сервера закончена, можно переходить к настройке клиента.

Настройка клиентов

Одинакова для всех трех пунктов. Я просто приведу скриншоты Nekobox, для других клиентов настраивать по аналогии.

Для Websocket:

The screenshot shows a configuration window titled "Edit" with a close button (X) in the top right corner. The window is divided into several sections for configuring a Websocket proxy.

- Common**: Fields for Name, Address, and Port (set to 443).
- VLESS**: Fields for UUID and Flow (dropdown menu).
- Custom Json Settings**: A button labeled "Not set".
- Settings**: Three dropdown menus for Network* (ws), Security* (tls), and Packet Encoding* (xudp).
- Network Settings (ws)**: Fields for Path* (/TestChatWS), Host*, EarlyData Length (2048), and EarlyData Name (Sec-WebSocket-Protocol).
- TLS Security Settings**: A checkbox for "Allow insecure*" (unchecked), a "Certificate" section with a "Not set" button, and fields for SNI* and ALPN*.
- TLS Camouflage Settings**: A dropdown for uTLS (chrome), and fields for Reality Pbk* and Reality Sid.

At the bottom right, there are "Cancel" and "OK" buttons.

Для gRPC:

Edit

Common

Name: [redacted]
Address: [redacted]
Port: 443

VLESS

UUID: [redacted]
Flow: [dropdown]

Custom Json Settings

[Not set]

Settings

Network*: grpc
Security*: tls
Packet Encoding*: xudp

Network Settings (grpc)

Path*: TestChatGRPC

TLS Security Settings

☐ Allow insecure* | Certificate: Not set
SNI*: [redacted]
ALPN*: [redacted]

TLS Camouflage Settings

uTLS: chrome
Reality Pbk*: [redacted]
Reality Sid: [redacted]

Cancel OK

И, понятное дело, значения «Path» должны совпадать с секретными URL'ами, которые вы задали в конфигах Nginx и XRay.

Fair use

Понятное дело, что то, что CDN разрешают на своих бесплатных тарифах проксировать Websockets и gRPC — это такой жест доброй воли от них. Давайте не наглотать и использовать эти возможности только в совсем безвыходных случаях, которые, надеюсь, все-таки не наступят.

И в заключение

Судя по ныне обсуждаемому законопроекту, запрещающему распространение информации о средствах обхода блокировок, подобному контенту на Хабре жить осталось недолго. Чтобы не потерять — сохраняйте локально копии этой и других статей из цикла и делитесь ими с друзьями и коллегами.

А если есть желание сказать автору спасибо за все его труды — то BTC
bc1q6mtfkxj0grqse8cefc4zgw4n9wckpvh5kp3h69

Теги: xray, v2ray, sing-box, vless, xtls, cdn, cloudflare, gcore, websocket, grpc


Хабы: Информационная безопасность, Системное администрирование, Nginx, Сетевые технологии, IPv6

Редакторский дайджест

Присылаем лучшие статьи раз в месяц

✕

Электронпочта



278

70

Карма

Рейтинг

Surrounded by idiots @MiraclePtr
В борделе на пианино играю


💬

Комментарии 25

Публикации

ЛУЧШИЕ ЗА СУТКИ

ПОХОЖИЕ

- 

andreybrylb

4 часа назад

Существование треугольника Шарыгина — это настоящее математическое чудо


👍 Простой

🕒 2 мин

👁 7.5K

💎 +47

🔖 31

💬 8
- 

igor_suhorukov

14 часов назад

Я бы не жил в Сочи в этих местах...


👍 Простой

🕒 6 мин

👁 14K

💎 +30

🔖 53

💬 59
- 

AndreyBolotov1989

13 часов назад

Как оптимизирован завод: вопросы оптимальной загрузки и «иридиевых чапельников»



Простой



9 мин



1.7K

[Обзор](#)

+29



23



3



orionII

5 часов назад

Вышла Java 21



Средний



18 мин



5K

[Обзор](#)

+28



20



14



AlexxIT

14 часов назад

Диалоги с кофеваркой, про Яндекс Алису и умный дом Home Assistant



Простой



6 мин



4.8K



+27



26



13



ravor84

13 часов назад

Перф-тесты VS аномалии. Вечная битва за производительность приложений на iOS



17 мин



1.5K



+22



9



5



Doctor_IT

7 часов назад

Платформа для анализа данных за вечер



11 мин



1.2K

[Кейс](#)

+21



35



0



angelov-iaroslav

8 часов назад

А теперь — поподробнее про флюс



Простой



12 мин



2.1K

Обзор



+21



19



3



SergeyAstanin

12 часов назад

Контроль состояния аккумулятора, если у вас стоит предпусковой подогреватель



Средний



8 мин



1.6K

Тutorial



+21



10



3



melnik909

11 часов назад

Очередной ответ на вопрос: «Зачем нужна семантика?»



Средний



7 мин



1K

Аналитика



+20



24



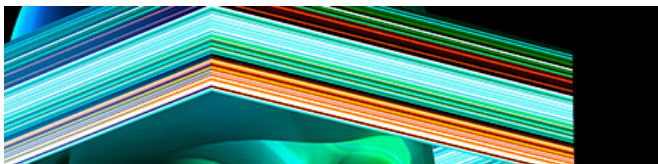
0

Конференция SmartDev 2023 об архитектуре, безопасности, AI, ML, работе с большими данными и других технологиях

Турбо

Показать еще

МИНУТОЧКУ ВНИМАНИЯ





Узнайте, что будет на SmartDev 2023



Глупым вопросам и ошибкам — быть! IT-менторство на ХК

ВАКАНСИИ

Системный аналитик (банковские проекты)

от 250 000 до 350 000 ₽ · Банк «Открытие» · Можно удаленно

Разработчик Golang

от 300 000 до 350 000 ₽ · Bell Integrator · Екатеринбург · Можно удаленно

Golang Developer

от 4 000 до 8 000 \$ · Karma8 · Можно удаленно

Middle DevOps Engineer

от 200 000 ₽ · Онлайн школа "СОТКА" · Можно удаленно

Системный администратор Linux

до 200 000 ₽ · Notamedia · Москва · Можно удаленно

Больше вакансий на Хабр Карьере

ЧИТАЮТ СЕЙЧАС

Блогер обнаружил в российском мониторе со 140 баллами локализации тайваньский чип от Realtek

👁 5.9K 💬 29

Существование треугольника Шарыгина — это настоящее математическое чудо

👁 7.6K 💬 8

Вышла Java 21

👁 5K 💬 14

«Подарил удочки и попрощался с друзьями на год»: как я стал Android-разработчиком, отказавшись даже от прогулок

 26K

 78

Химики придумали унитаз, к которому ничего не прилипает


 6.4K

 48

Конференция SmartDev 2023 об архитектуре, безопасности, AI, ML, работе с большими данными и других технологиях

Турбо

ИСТОРИИ





Статьи, советы и мнения о подготовке резюме и собеседованиях




Полезная подборка о собеседованиях

Разбор резюме дизайнеров на Хабр Карьере






Разбор резюме дизайнеров



Что почитать хорошего? Недельный топ-7 годных статей из блогов компаний



Топ-7 годных статей из блогов компаний



Учиться и ещё раз учиться
Подборка советов, лайфхаков и мнений для более эффективного обучения



Учиться хорошо



В одно обычное утро Станислав, инженер-разработчик C++, думал над проблемами своего проекта

У нас высоконагруженный проект, а он пишет как филозофские, не домысливая, ни автоматизации тестирования, ни автоматизации деплоя. Вроде там же мидлы C++, указатели да память вертеть умеет, а в консоли застрял, кхм...



Золотая рыбка, хочу, чтоб у меня все было

РАБОТА

DevOps инженер
44 вакансии

Специалист по информационной безопасности
109 вакансий

Системный администратор
119 вакансий

Все вакансии

Ваш аккаунт	Разделы	Информация	Услуги
Войти	Статьи	Устройство сайта	Корпоративный блог
Регистрация	Новости	Для авторов	Медийная реклама
	Хабы	Для компаний	Нативные проекты
	Компании	Документы	Образовательные программы
	Авторы	Соглашение	Стартапам
	Песочница	Конфиденциальность	Спецпроекты



Настройка языка

Техническая поддержка

Вернуться на старую версию