

[КАК СТАТЬ АВТОРОМ](#)[Тестировщики, вам сюда](#)[Топим снег скидками: промо...](#)**UranusExplorer**

12 часов назад

Надежный обход блокировок в 2024: протоколы, клиенты и настройка сервера от простого к сложному

Простой

36 мин

26K

Настройка Linux*, Информационная безопасность*, Системное администрирование*, Сетевые технологии*

[Тutorial](#)

Поскольку блокировки интернета в РФ в последние недели и месяцы многократно активизировались, а госмарasmus все усиливается и усиливается, стоит еще раз поднять тему обхода этих самых блокировок (и делаем ставки, через сколько дней на эту статью найдоброжелатели напишут донос в РКН чтобы ограничить к ней доступ на территории страны).

Вы, наверняка, помните отличный цикл статей на Хабре в прошлом году от пользователя MiraclePtr, который рассказывал о разных методах блокировок, о разных методах обхода блокировок, о разных клиентах и серверах для обходов блокировок, и о разных способах их настройки (раз, два, три, четыре, пять, шесть, семь, восемь, девять, десять, и вроде были еще другие), и можете спросить, а зачем еще одна? Есть две основные причины для этого. Во-первых, те статьи представляют собой огромный объем материала, где каждая деталь тщательно разобрана, и предложено множество вариантов. Однако для пользователей без опыта может быть сложно разобраться в этой куче информации и понять, какой протокол и какую схему настройки использовать. Во-вторых, со времени написания этих статей прошло некоторое время, и появились новые, более простые в настройке, клиенты, новые схемы и идеи. Поэтому эту статью можно рассматривать как краткий, но информативный пересказ той горы статей с добавлениями, поправками и более простыми вариантами настройки. Кроме того, я упорядочу варианты по сложности, чтобы те, кто не уверены в своих навыках администрирования и сетевых технологий, могли воспользоваться более простыми методами, а более сложные варианты будут представлены в конце для эстетов и затейников.

Итак, вкратце, какие на сегодняшний день есть проблемы с популярными VPN- и прокси-протоколами?

Детектировать и блокировать соединения можно следующими способами, от лайтовых к хардкорным:

+192

440



68

установлено так называемое оборудование ТСПУ);

- Active probing: при маскировке протокола под HTTPS (обычный веб-сайт), цензоры пытаются подключиться к такому "сайту", чтобы проверить, действительно ли там есть настоящий веб-сервер, а так же используя пути (URL), зарезервированные в VPN-протоколах для установления соединения, в результате чего сервер себя демаскирует; есть свидетельства, что РКН тоже начал заниматься active probing'ом;
- Анализ TLS fingerprint ("отпечатка" - набора поддерживаемых шифров, эллиптических кривых, поддерживаемых расширений протокола, и т.д.) клиента. Отпечатки современных браузеров (Chrome, Firefox, и т.д.) известны, а отпечатки многих VPN- и прокси-клиентов могут от них отличаться, что влечет за собой выявление и блок таких клиентов; РКН об этом в курсе, какое-то время назад он уже пытался блокировать Tor-Snowflake по отпечатку библиотеки Pion;
- Детектирование TLS-inside-TLS статистическим анализом размеров пакетов (см. trojan-killer как пример);
- Белые списки протоколов - это когда разрешены подключения только по известным протоколам, которые узнаваемы для оборудования DPI (оно же ТСПУ), а все остальные протоколы блокируются. Если используемый протокол не похож ни на что известное механизму фильтрации, например выглядит как случайная последовательность байтов, то такое соединение блокируется. Имеются свидетельства, что РКН уже применял подобные методы во время волнений в Дагестане в прошлом году, пытаясь заблокировать Telegram (трафик tg-проху, не имеет характерных признаков и выглядит как случайный байтовый поток);
- Белые списки доменов. Это не обязательно может быть полная блокировка тех доменов, что не в списке, а, например, ограничение скорости ко всем "недоверенным" доменам, либо блокировка доступа к ним после достижения определенного предела объема переданного трафика. Такое настоящее время применяется в Иране;
- Черные списки диапазонов IP-адресов. В Туркменистане, например, этими черными списками забанено больше половины всех адресов интернета. При обнаружении запрещенного контента, VPN- или прокси-сервера на одном IP-адресе, блокируется целая подсеть.

Последний пункт особенно важен и заслуживает особого внимания. Существует известное выражение: «пока гром не грянет, мужик не перекрестится». К сожалению, оно полностью относится и к нашей сегодняшней теме. Многие люди думают «А, ну пока все работает, а как работать перестанет, тогда я установлю-перенастрою и решу проблему». Это очень опасный подход. Допустим, у вас установлен сервер Wireguard или OpenVPN, и пока всё функционирует нормально. Но уже точно известно, что Роскомнадзор на подконтрольном ему оборудовании может детектировать такие подключения. В результате они могут легко начать составлять списки IP-адресов таких серверов для полной блокировки доступа к ним когда запахнет жареным, например,

во время обострения политической жизни в стране, и я более чем уверен, что они уже составляют такие списки. Если вы на таком «отравленном» IP-адресе потом поднимите какой-нибудь другой, более надежный и недетектируемый сервис, вы все равно не сможете им пользоваться, ничего не будет работать — останется только арендовать новый виртуальный сервер, и надеяться, что на IP-адресе, который вам достанется, никто не хулиганил до этого. Если вы не верите, что дело до такого дойдет, то, увы — до такого уже доходило. Пару лет назад РКН искал Tor-мосты (bridges), запрашивая их с веб-сайта Tor, прикидываясь обычным пользователем, в результате чего IP-адреса таких мостов на какое-то время **блокировались** полностью, не помогала ни смена порта, и даже не удавалось подключиться к серверу по SSH.

Итак, начнем с того, **чем в наше время пользоваться уже не стоит**:

Обычный SOCKS - изначально не предусматривает шифрования, поэтому цензоры легко увидят, куда вы подключаетесь и заблокируют соединение.

OpenVPN - его научились блокировать уже давно в России, Китае, Иране и много где еще. Не помогает ни смена TCP/UDP, ни смена порта на нестандартный, ни более навороченные опции в конфиге. Точнее не так, в некоторых случаях они помогают, но, как говорил один известный персонаж отечественной истории, *"это не ваша заслуга, а наша недоработка"*, и в любой момент это все может перестать работать, благо технические возможности для этого давно уже есть.

IPSec - аналогично, детектируется блокируется на раз-два.

Wireguard - он очень популярен, но, к сожалению, тоже совершенно не устойчив к блокировкам, в чем мы уже неоднократно убедились.

Tor - адреса входных нод доступны в публичном реестре, и поэтому могут быть легко заблокированы. Бриджи (**bridges**) пока работают, но РКН в прошлом неоднократно их выборочно банил (видимо, запрашивая списки, прикидываясь обычным пользователям и внося полученные адреса в блок), к тому же их протокол выглядит как рандомный поток байт, и блокировку по белым спискам протоколов не переживет. **Snowflake** - после некоторых исправлений более-менее работает, можно пользоваться, однако поиск бриджа для входа там выполняется через один известный домен на CDN, который как-то раз уже тоже **блокировали**. **WebTunnel** - пока работает, но таких бриджей мало.

Что еще пока работает, но с оговорками:

SSTP, Softtherher, AnyConnect и OpenConnect - это уже гораздо лучше, поскольку выглядят как HTTPS-трафик. Проблемы есть две: их можно детектировать методом active probing (и как я говорил выше, Роскомнадзор такое уже умеет), потому что при заходе на них браузером по определенным адресам (описанным в стандартах этих протоколов) они

демаскируют себя, и вторая проблема - характерный TLS fingerprint (отпечаток) их клиентов. Первая проблема частично решена в новых версиях OpenConnect с включением специального метода "camouflage". Однако вторая проблема, к сожалению, пока не имеет решения ни у одного из них. Тем не менее, OpenConnect-клиент, теоретически, можно пересобрать с использованием OpenSSL вместо GnuTLS, что снижает вероятность обнаружения.

Shadowsocks, Shadowsocks-2022, а также **Outline**. SS - известная разработка от китайских товарищей, SS-2022 - более новая версия протокола, в которой исправлен ряд уязвимостей, а Outline - удобный и простой в настройке клиент/сервер, основанный на Shadowsocks. На сегодняшний день вполне работает, можно пользоваться. Одно но: так как трафик SS представляет из себя совершенно рандомный набор байт без каких-либо узнаваемых сигнатур, блокировку по белым спискам протоколов (см. выше) он не переживет, поэтому я бы рассматривал его как запасной, но не как основной вариант. Существует так же протокол **VMess**, который по своему принципу очень похож на Shadowsocks с теми же недостатками.

Еще одна интересная особенность ванильного Shadowsocks в том, что он передает TCP-трафик как TCP, а UDP-трафик как UDP. Из-за этого при наблюдении извне он выглядит весьма характерно (хотя мне неизвестно случаев блокировок на основе такой эвристики), однако существует его расширение UoT (UDP over TCP), которое поддерживается многими серверами и клиентами, которое заворачивает UDP в TCP, в результате чего трафик выглядит гораздо более однообразно и вызывает меньше подозрений.

AWG, оно же **AmneziaWG** - улучшение протокола Wireguard, разработанное командой проекта Amnezia VPN, чтобы сделать Wireguard устойчивым к детектированию оборудованием DPI, при этом сохранив его преимущества. Он действительно работает, и работает неплохо. Минус тот же, как и SS - поскольку оборудование TCPУ этот протокол не распознаёт, то блокировку по белым спискам протоколов в случае чего он не переживет.

SSH — скажу кратко, пока работает. Правда, есть довольно много свидетельств, что в Китае, видя «нецелевое» использование SSH (может быть, анализируя объемы и тайминги передаваемых туда-сюда данные и паттерны поведения нейросетями?), ограничивают скорость передачи настолько, что пользоваться им для серфинга становится невозможным, тормозит даже консоль.

Что работает и вероятно будет продолжать работать:

VLESS — самый распространенный и перспективный на сегодняшний день протокол, от создателей известного клиента-сервера XRay. Работает поверх TLS, искусно маскируясь под обращение к какому-нибудь безобидному веб-сайту. Поддерживается в очень многих десктопных и мобильных клиентах, и именно на его основе строятся всевозможные схемы

обхода блокировок. Вместе с ним вы можете встретить также большое количество других слов, обычно обозначающие его расширения или отдельные фишки. Рассмотрим основные из них:

- **uTLS** - изменение TLS-fingerprint клиента; например, можно заставить выглядеть исходящие подключения как подключения от браузеров Chrome, Firefox, Safari, или вообще генерировать каждый раз новый рандомный фингерпринт;
- **XTLS-Vision** - специальный режим работы, затрудняющий детектирование TLS-inside-TLS. Когда это режим включен, если прокси-сервер определяет подключение к удаленному серверу по TLS 1.3, то шифруется только хендшейк, а после этого уже зашифрованные (TLS между вашим браузером и удаленным сервером) данные передаются без повторного слоя шифрования (отключается TLS между прокси-клиентом и прокси-сервером). Такая схема совершенно безопасна (данные в любом случае шифруются как минимум один раз и не видны стороннему наблюдателю), но ломает TLS-inside-TLS эвристики, а заодно экономятся ресурсы процессора;
- **XUDP** - специальное расширение для передачи UDP-пакетов через прокси, реализует полный Full Cone NAT с сохранением номера порта. Без этого расширения в некоторых случаях могут не работать аудио-видео-звонки в мессенджерах, оно было создано именно для решения этой проблемы;
- **XTLS-Reality** - вот это самое крутое. То что VLESS работает поверх TLS и маскируется под какой-то веб-сайт, означает что у вас должен быть домен и TLS-сертификат для него. А еще в случае блокировок по белым спискам, ваш никому неизвестный сайт (домен) вряд ли окажется в списке разрешенных ресурсов.
XTLS-Reality решает обе эти проблемы: вам *не нужен свой домен и сертификат*, с XTLS-Reality сервер очень достоверно маскируется под какой-нибудь известный и популярный сайт (например, google.com, vk.com, и т.д.), и представляется его аутентичным TLS-сертификатом.
Работает это так: в момент установления подключения, клиент "подмешивает" в поля хендшейка определенные данные, определить наличие которых можно только зная секретный ключ, при этом со стороны хендшейк выглядит обычно. Прокси-сервер, зная секретный ключ, может определить, был ли запрос на установку соединения послан от прокси-клиента или же от кого-то другого (например, цензора, желающего проверить, что там на сервере). В первом случае установится подключение для передачи данных через прокси, во втором случае прокси начнет прозрачно пересылать запросы на сервер сайта, под который мы маскируемся, в результате чего цензор увидит самый настоящий сертификат этого сайта, и вообще поведение нашего сервера будет полностью соответствовать поведению сервера этого сайта.

Иногда в интернете можно встретить утверждения, что "VLESS не имеет своего механизма шифрования, а значит он не безопасен". **Это не так.** Подключения VLESS **всегда** как минимум один раз шифруются TLS. VLESS действительно не имеет

дополнительных механизмов шифрования, потому что стандартного TLS для цели защиты данных более чем достаточно и нет нужды городить сверху что-то еще, от этого будет только хуже. Более того, даже когда используется XTLS-Vision, который определяет наличие TLS-внутри-TLS и отключает один из слоев шифрования, передаваемые данные между вами и конечным сервером все равно **всегда** будут зашированы хотя бы один раз.

- **WebSocket, gRPC, HTTP** - разные варианты транспортов для VLESS и других протоколов, когда VLESS используется не сам по себе, а "заворачивается" в какой-нибудь другой протокол. Веб-сокеты часто используются для доступа к прокси-серверу через популярные CDN (сети доставки контента), такие как Cloudflare или Amazon Cloudfront, что является хорошим запасным вариантом, если вдруг ваш сервер попадет под блокировку. А еще с помощью CDN можно прикрываться чужими доменами и IP-адресами, см. эту статью от всеми нами любимого автора.

Ну и упомянем еще несколько протоколов:

Trojan - более старый аналог VLESS, работает примерно по таким же принципам. Каких-либо особых преимуществ по сравнению с VLESS не имеет. Может использоваться с uTLS, XTLS-Vision и XTLS-Reality, и более того, *должен* использоваться с ними, потому что без XTLS-Vision он детектируется по принципу поиска TLS-in-TLS вообще элементарно.

Cloak - протокол, очень сильно похожий по принципам работы на XTLS-Reality (маскировка именем и настоящим сертификатом популярного веб-сайта). Используется не сам по себе, а для заворачивания в него Shadowsocks, OpenVPN и вообще всего что захочется. Shadowsocks-over-Cloak и OpenVPN-over-Cloak поддерживают, например, клиент и сервер Amnezia VPN.

Важно! Появились сообщения о том, что в некоторых странах цензоры научились блокировать Cloak, но на самом деле поводов для паники нет. Причина была в том, что Cloak долгое время использовал TLS fingerprint очень старой версии браузера Firefox, и блокировали его именно по этому критерию. В новых версиях автор обновил fingerprint на отпечаток от современной версии Firefox и пообещал в дальнейшем обновлять его регулярно. Поэтому если вы используете Cloak - не забывайте почаще обновлять клиенты.

Кроме того, если вы используете Cloak, устанавливая его с помощью каких-либо скриптов автонастройки (или той же Амнезии), обязательно меняйте домен, под который вы маскируетесь на какой-нибудь другой, не надо использовать домен, предлагаемый по умолчанию, иначе его рано или поздно тоже забанят.

Hysteria, и её новый вариант **Hysteria2** - протокол на базе QUIC (HTTP/3), целью которого является быстрое установление соединения и работа через каналы с большими пингами и потерями пакетов. Ванильная Hysteria в России работать не будет, поскольку QUIC в РФ

уже давно почти полностью забанен, однако Hysteria имеет также режим обфускации (называется «salamander»), когда заголовки пакетов шифруются и становятся ни на что не похожи — оно работает и весьма неплохо, но, к сожалению, блокировку по белым спискам протоколов не переживет. Правда, Hysteria работает по UDP, и есть шанс, что белые списки применят только к протоколам на базе TCP, а про UDP забудут (да, такое уже было).

mKCP - более старый протокол аналогичный Hysteria. Поддерживается только в XRay. Не смотря на его "более старость", в отличие от Hysteria может свои UDP-пакеты маскировать под SRTP (FaceTime), WebRTC, uTP (Bittorrent).

Ну из более экзотических протколов на грани с извращениями:

PingTunnel - реализация туннеля через ICMP-пинги. Можно использовать оригинальный PingTunnel, либо GOST (но между собой они не совместимы), который умеет еще кучу всего.

DNSTT - туннелирование через DNS-запросы. На всякий случай уточню, это не просто заворачивание трафика в UDP на 53-ий порт. В случае с DNSTT прокси-клиент делает запросы на настоящий DNS-сервер, например, на DNS-сервер вашего провайдера или DNS Яндекса, пытаясь получить DNS-записи для вашего домена (а точнее, для случайно сгенерированного поддомена, в имени которого зашифрованы передаваемые на прокси). Тот, в свою очередь, будучи вынужденным ответить на запрос, обращается к вашему специально подготовленному DNS-серверу, который ответит на запрос, заодно напихав в ответ данные, передаваемые от прокси-сервера к вам, и в итоге вы их получите от DNS-сервера вашего провайдера/яндекса/итд. Про настройку DNSTT есть отличная статья на [Хабре](#), а в качестве клиента можно использовать [DarkTunnel](#) (DNSTT+SSH).

Последние два варианта работают очень медленно (скорость в лучшем случае два-три мегабита в идеальных условиях), но они работают даже там, где не работает больше вообще ничего. Например, с помощью DNSTT некоторым затейникам удавалось бесплатно сидеть в интернете на борту самолетов, когда за такую услугу авиакомпании требовали немалых денег (если что, осуждаем).

Теперь по реализациям:

XRay (оно же XRay-core) - на сегодняшний день самая актуальная и популярная реализация клиента и сервера. Поддерживает Shadowsocks, Shadowsocks-2022, VMess, VLESS, Trojan, всевозможные транспорты (websockets, gRPC) и всевозможные расширения (XTLS-Vision, XTLS-Reality, XUDP, uTLS). Более того, авторы XRay — как раз-таки создатели этих самых расширений XTLS, поэтому там всегда самая свежая и правильная их реализация.

V2Ray - старый проект, из которого когда-то форкнулся XRay. В настоящее время развивается очень вяло и не поддерживает многое из того, что здесь описано. Если вы где-нибудь в интернете встретите статью, описывающую настройку V2Ray в качестве сервера, то скорее всего она очень старая и лучше поискать что-нибудь более свежее. В 2024 вместо V2Ray стоит использовать XRay, он развивается гораздо активнее и умеет гораздо больше.

Sing-box - аналог Xray, поддерживает все то же самое, что и XRay, и даже больше - умеет SSH как клиент, Hysteria, и всякое другое. Форматы конфигов XRay и Sing-box не совместимы. Сами реализации протоколов в XRay и Sing-box между собой, в принципе, совместимы, но есть и небольшие неприятности. Например, если у вас и клиент и сервер XRay, то XUDP (решение проблем со звонками в мессенджерах) будет работать автоматически, по умолчанию, если у вас на клиенте Sing-box, а на сервере XRay, то чтобы он работал, нужно обязательно явно активировать XUDP в настройках (обычно называется "packet encoding" или как-то так), а вот если у вас на клиенте XRay, а на сервере Sing-box, тогда оно работать не будет :(

X-UI, 3X-UI, Marzban, Hiddify - так называемые "панели", веб-интерфейсы для прокси-серверов. В их основе чаще всего используется тот же самый XRay. Изначально задуманы как более простой в установке и настройке вариант сервера для неподкованных пользователей...

...но я со своей стороны пользоваться ими не советую. Почему? Причин несколько.

Во-первых, в консоль лезть все равно придется, просто хотя бы для того чтобы установить эти панели из Docker-образа.

Во-вторых, красивый интерфейс все равно не избавит вас от скрупулезного заполнения кучи полей в конфиге, причем заполнять их надо очень внимательно, чтобы не ошибиться, иначе ничего не будет работать.

В-третьих, "настройки по умолчанию" эти панели нередко предлагают бредовые или даже небезопасные. В одной панели при создании VLESS-конфига (не помню в какой, кажется это был Marzban) панель автоматически сгенерировала и подставила порт типа 22245. Хотелось схватиться за голову и закричать "Вы что делаете-то, ироды?!" (поясню: TLS должен быть *только* на 443 порту, иначе это очень подозрительно, особенно для XTLS-Reality).

В-четвертых, этих панели хоть и умеют генерировать ссылки и QR-коды для легкой настройки клиентов, в этих ссылка/кодах не хватает важных параметров. Например, в некоторых панелях в сгенерированных ссылках не подставляется параметр "fp=chrome" или "fp=firefox" (fingerprint), в результате чего если в вашем клиенте не задан uTLS-fingerprint по умолчанию, есть риск нарваться на блокировку. Там часто не хватает

параметра "packetEncoding=xudp", в результате чего если у вас будет клиент, основанный на sing-box, могут быть проблемы со звонками в мессенджерах.

И в-пятых, самое важное - часто неопытные пользователи, устанавливая эти панели, оставляют их на стандартных портах и со стандартным путем (URL), в результате чего, недоброжелатель, жаждущий узнать "а что это у нас там на сервере, не прокси ли?" может просто просканировать стандартные порты, на одном из которых панель радостно признается ему "привет, да, это я", что демаскирует сервер целиком и полностью, и ладно еще, если пользователь не забыл изменить дефолтные логин/пароль. Плюс панель может содержать уязвимости безопасности, и тут бояться уже надо не цензоров, а малолетних кулхацкеров, которые просто сканируя Интернет могут на нее наткнуться, использовать эксплоит и устроить на вашем сервере что-нибудь нехорошее (например, ботнет).

Клиентов под все это дело существует тоже немало - **Nekobox/Nekoray**, **Hiddify-Next**, **v2rayN**, **v2rayNG**, **FoxRay**, **Streisand**, **Shadowrocket**, **InvisibleMan**, и т.д. Часть из них основана на XRay (даже если в названии есть "v2ray"), часть на Sing-box. Про них мы поговорим попозже в этой же статье.

Итак, я надеюсь, что дойдя до этой части, у читателя примерно сложилось представление, что стоит, а что не стоит использовать для обхода блокировок роскомпозора в 2024. Варианты пусть и не настолько надежные, как сбор чемоданов и заведение трактора (самый лучший, но, к сожалению, не всем доступный вариант), но какое-то время помогут продержаться. В этой статье мы будем настраивать VLESS с XTLS-Reality, потому что он на сегодняшний день самый надежный, и при этом довольно простой в настройке. В качестве "запасного варианта" на случай ЧП мы еще поднимем Shadowsocks-2022, mKCP и SSH на том же сервере, а в конце я поделюсь более сложными и навороченными схемами для желающих экспериментов.

Что если мне нужен именно VPN?

Иногда бывает так, что человеку нужен не прокси, а именно VPN. Например, для объединения сетей, доступа в корпоративную сеть или коммуникацию между разными клиентами. И желательно чтобы криворукие обезьяны с гранатами из надзорных органов его случайно не заблокировали ковравыми блокировками протоколов.

Тут, к сожалению, все довольно грустно.

Если у вас было объединение сетей и вы использовали Wireguard, то самым простым вариантом будет перейти на AmneziaWG.

Если у вас было объединение сетей и вы использовали OpenVPN, то заверните его в Cloak, но готовьте к просадке производительности.

Если вам надо, чтобы к вашей сети могли подключаться пользователи с обычных десктопов и телефонов, то можно попробовать OpenConnect сервер с режимом camouflage (см. статью от MiracitPtr об этом).

А мы, тем временем, продолжим говорить о прокси.

Если все кажется очень сложным

Я помню, что люди разные, и кто-то, прочтя эту статью может сказать "Это все не для меня, слишком сложно, не знаю что делать". Сразу предупрежу, несмотря на огромный объем, на самом деле настройка простых вариантов (я начну с них) вполне возможна за один вечер даже для человека с совсем минимальным опытом системного администрирования.

Но если все равно кажется слишком сложно или не получается, есть другие варианты.

Вариант раз - **панель X-UI или 3X-UI** (по ссылке статья с инструкцией по настройке). Хотя я выше и рассказывал, почему я не люблю эти панели, но глупо отрицать, что лучше с панелью и хоть каким-то доступом в неподцензурный интернет, чем без панели остаться в чебурнете.

Вариант два - **Amnezia VPN**. Позволяет развернуть надежный VPN-сервер на любом VPS буквально парой нажатий кнопок из приложения. Поддерживает протоколы Shadowsocks, Amnezia-WG и OpenVPN-over-Cloak, о которых я рассказывал выше. Есть клиенты под все популярные десктопные и мобильные платформы. Проект разрабатывается группой энтузиастов на донаты и гранты, организует хакатоны, проходил сторонний аудит безопасности.

Единственный нюанс — при использовании Amnezia с протоколом Cloak, не забудьте зайти в настройки сервера в приложении, и поменять домен маскировочного сайта по-умолчанию на какой-нибудь другой, так будет надежнее.

Начинаем настройку. Выбор хостера, подготовка сервера

Первый частый вопрос: какой VPS/VDS выбрать? Ответ: любой за пределами РФ. Но, понятное дело, многих находящихся в РФ интересуют хостеры, позволяющие оплачивать услуги с российских банковских карт. Можно воспользоваться услугами российских хостеров, сдающих в аренду сервера за рубежом (например, довольно популярной локацией являются Нидерланды), но тут надо быть внимательным. Недавно в России приняли закон о том, что хостеры должны вноситься в определенный реестр, что влечет для них обязательства типа слива ваших данных госорганам. Те хостеры, что поумнее,

разделили свой бизнес на разные части: российскую (домен в зоне ".ru" и российское юрлицо) и выставление счетов в рублях, и не-российскую (домен в другой зоне, например ".com", и юрлицо где-нибудь в Британии, Эмиратах или на Кипре) с выставлением счетов в евро/долларах, но при этом даже во втором случае часто можно по-прежнему платить российскими рублевыми банковскими картами. Поэтому при возможности стоит предпочесть второй вариант. Например, **VDSina** (юрлицо в ОАЭ) с недавних пор предлагает сервера в Амстердаме за 2 доллара в месяц. У **Aeza** (британское юрлицо) иногда появляются сервера с промо-тарифом в Швеции за 1 евро в месяц, да и обычные тарифы у них относительно недорогие. Еще есть **Inferno Solutions** с юрлицом в UK, но там подороже.

Также не самым плохим вариантом может быть не-российские хостеры, но при это дружелюбные к российским клиентам, например **PQ.hosting** (Молдова), **Friendhosting** (Болгария), **LLHOST** (Эстония), **is*hosting** (Эстония).

Ну и наконец, если у вас есть возможность оплачивать услуги иностранными картами, то можно выбирать все что угодно. Многие хостеры также поддерживают оплату криптовалютой. Если ищите подешевле, существует веб-сайт **LowEndBox** (и его спутник, форум **LowEndTalks**), где публикуются разные акционные предложения от хостеров со всего мира, и иногда там можно найти очень неплохие виртуалки очень задешево, например за 10-15 долларов в год. Главное проверяйте внимательно, совсем дешевые VPS (за 7 долларов в год) могут не иметь публичного IPv4-адреса (только IPv6 и NAT-IPv6). Поставить и использовать прокси на них тоже можно (через Cloudflare CDN), но неопытным пользователям я бы в это ввязываться не советовал.

Особых системных требований нет. Я тестировал XRay с несколькими пользователями на очень дешевом VPS с одним ядром и 512 мегабайт памяти, проблем не было никаких, более того, он вполне себе работает даже на виртуалках с 256 мегабайтами. Тип виртуализации - особо не важен, нормально будет и с KVM, и с OpenVZ.

Какую ставить операционную систему? Я буду описывать настройку на Debian-based ОС, то есть если вы чайник в этом деле, то выбирайте Debian или Ubuntu. На других дистрибутивах работать тоже будет, но могут отличаться пути к файлам, команды пакетного менеджера, и т.д. **Важно:** если вы используете Fedora/CentOS и после установки ничего не работает (не удастся подключиться к прокси), проверьте, что на сервере на фаерволе не заблокированы порты, которые вы используете (например, 443) или вообще все порты кроме разрешенных.

Итак, вы купили VPS у хостера и вам пришли логин и пароль от него. Что я советую сделать сразу после установки:

1. Сменить пароль пользователя root с изначального на ваш уникальный. Не использовать учетную запись root, а создать непривилегированного пользователя

командой `adduser` и назначить ей сложный пароль командой `passwd <username>` .
Когда вы убедитесь, что она работает, стоит запретить вход для root-юзера по SSH, выставив (раскомментировав) в "no" или "prohibit-password" опцию `PermitRootLogin` в файле `/etc/ssh/sshd_config`

2. Перевесить SSH со стандартного порта 22 на нестандартный порт (например, на 54321, 41467, выберите любой случайный выше 1000). Это особенно важно для маскировки XTLS-Reality. Порт задается опцией " `Port` " в том же `/etc/ssh/sshd_config`
3. Обновите версии пакетов в системе на свежие, в Debian и Ubuntu это делается командами `apt update` и `apt upgrade` .

Процесс установки XRay неоднократно описан в упомянутых выше статьях, я повторю его здесь.

Разработчики XRay разработали скрипт, который автоматически загружает XRay для вашей операционной системы и создает systemd-юнит для его запуска: [ссылка на скрипт](#).

Установить его можно одной длинной командой:

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

Единственное отличие при использовании скрипта заключается в том, что файлы конфигурации XRay будут храниться не в `/opt/xray` , а в `/usr/local/etc/xray/` .

Или вы можете установить XRay вручную. Для этого переходим по ссылке на последний релиз `XRay-core` и скачиваем самую свежую версию:

```
wget https://github.com/XTLS/Xray-core/releases/download/v1.8.9/Xray-linux-64.zip
```

Создаем каталог, распаковываем и делаем исполняемым файл (поскольку он поставляется в архиве `.zip` , права доступа могут быть потеряны при распаковке):

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

Затем создаем systemd-юнит (`nano /usr/lib/systemd/system/xray.service`) и вставляем в него следующий текст

```
[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
```

Активируем:

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

Просто: Настройка VLESS с XTLS-Reality

Сначала нужно определиться, по какой сайт вы будете маскироваться.

Вариант раз: это должно быть какой-то очень популярный иностранный сайт, желательно не связанный с политикой и СМИ, не относящийся к "нежелательным организациям", и к которому наврядли возникнут вопросы со стороны органов РФ (например, youtube.com - довольно плохой выбор в этом плане, есть шанс блокировки в будущем). Просто попробуйте вспомнить какой-нибудь ресурс, которым пользуются много людей и компаний, который цензоры до последнего не будут блокировать без большой нужды, чтобы не разломать пол-интернета (и нет, это не гугл). Предлагать варианты тут не буду, пусть каждый выберет какой-то свой. И да, не обязательно использовать главный домен сайта, можно брать какие-то из поддоменов.

Выбрав домен, проверяем, что он подходит под наши критерии командой curl. Например, для www.microsoft.com команда будет выглядеть так:

```
curl -v -L --tlsv1.3 --http2 https://www.microsoft.com
```

Если в ответ вы видите много логов и HTML-код, то все хорошо, должно работать. Если в ответ вы получаете какую-то ошибку, то стоит попробовать поискать другой домен для маскировки.

Вариант два: поищите какие-нибудь не слишком популярные сайты (к ним меньше внимания) в сети того же хостинга, что и ваш сервер. Для этого существует утилита <https://github.com/XTLS/RealTLSScanner>, скачать готовый экзешник можно там со страницы Releases.

Запускаете ее командой `./RealTLSScanner -addr IP_вашего_VPS -showFail` и внимательно изучаете результаты. Если в какой-то из строк написано "Found TLS", то сначала попробуйте зайти на этот адрес (по доменному имени) браузером, должен открыться какой-то сайт без ошибок, далее проверьте с помощью какого-нибудь сервиса или утилитой ping, что IP-адрес за этим доменом совпадает с тем, что вы нашли (иначе это может быть какой-нибудь ваш единомышленник, поднявший XTLS-Reality у себя), и в

заключение проверьте командой `curl` как выше, что сервер соответствует всем требованиям. Если сработало - поздравляю, используйте этот домен для маскировки.

Важно отметить тот факт, что XTLS-Reality имеет недостаток - если сайт, под который вы маскируетесь, испытывает технические проблемы (например, временно отключил TLSv1.3), или огородился от вашего сервера (улав от проксируемых запросов на установление соединения), или попал под блокировку, то точно так же перестанет работать и ваш прокси. В этом случае совершенно нормально будет поменять используемый маскировочный домен на какой-то другой. Мы потом еще настроим на сервере Shadowsocks и SSH как запасной вариант на случай проблем, а пока продолжаем с настройкой VLESS.

Вам нужно будет сгенерировать несколько параметров командами XRay:

`xray uuid` - сгенерирует UUID, это что-то типа логина пользователя

`xray x25519` - сгенерирует приватный и публичный ключ сервера.

Тут внимательно. **Приватный** ключ используется только в конфиге на сервере, **публичный** ключ же наоборот, в конфиге сервера не упоминается, зато его должны будут знать все клиенты вашего прокси. Сохраните оба ключа в надежное место.

После этого нужно подготовить конфиг-файл. Пример конфига может показаться страшным для неопытного пользователя. **Но!** На самом деле ничего страшного тут нет, все что требуется, это скопипастить содержимое файла из статьи и изменить немножко строк.

Итак, приводим (редактором nano или как вам удобнее) файл `/opt/xray/config.json` (или `/usr/local/etc/xray/config.json` если ставили скриптом) к следующему виду:

```
{
  "log": {
    "loglevel": "info"
  },
  "inbounds": [
    {
      "listen": "IP_адрес_вашего_сервера",
      "port": 443,
      "protocol": "vless",
      "tag": "reality-in",
      "settings": {
        "clients": [
          {
            "id": "ваш_UUID",
            "email": "user1",
            "flow": "xtls-rprx-vision"
          }
        ]
      }
    }
  ]
}
```



```
    ],
    "decryption": "none"
  },
  "streamSettings": {
    "network": "tcp",
    "security": "reality",
    "realitySettings": {
      "show": false,
      "dest": "ваш_маскировочный_домен:443",
      "xver": 0,
      "serverNames": [
        "ваш_маскировочный_домен"
      ],
      "privateKey": "ваш_ПРИВАТНЫЙ_ключ",
      "minClientVer": "",
      "maxClientVer": "",
      "maxTimeDiff": 0,
      "shortIds": ["" ]
    }
  },
  "sniffing": {
    "enabled": true,
    "destOverride": [
      "http",
      "tls",
      "quic"
    ]
  }
},
"outbounds": [
  {
    "protocol": "dns",
    "tag": "dns"
  },
  {
    "protocol": "freedom",
    "tag": "direct"
  },
  {
    "protocol": "blackhole",
    "tag": "block"
  }
],
"routing": {
  "rules": [
```

```
{
  "type": "field",
  "protocol": "bittorrent",
  "outboundTag": "block"
},
"domainStrategy": "IPIfNonMatch"
}
```

Это может показаться сложным и страшным, но на самом деле ничего особо сложного там нет, и все уже почти готово.

Файл состоит из нескольких секций. Первая - `log`, определяет настройки логгирования (мы ставим уровень `info`). Вторая, `"inbounds"`, определяет, какие протоколы и с какими параметрами будут использоваться для входящих подключений на прокси. Третья, `"outbounds"`, описывает, куда может пойти трафик с вашего прокси, там указаны `"freedom"` (выход в чистый интернет), `"blackhole"` (если надо не пускать пользователя куда-то) и `"dns"` (встроенный dns-сервер). Последний `"routing"` определяет дополнительные правила, что и куда мы будем перенаправлять, например, подключения по протоколу Bittorrent блокируем, отправляя в `blackhole`, чтобы ваши пользователи не создали вам проблем, раздавая гигабайты через ваш сервер (если не надо можете удалить эти строки из конфига). Все остальное отправляется в свободный интернет (по умолчанию).

Что вам важно в нем поменять:

- в `"inbounds"` в параметре `"listen"` укажите IPv4-адрес вашего сервера
- `inbounds -> settings -> clients -> id` укажите UUID, который вы сгенерировали ранее
- `inbounds -> streamSettings -> realitySettings -> serverNames` укажите домен, под который вы маскируетесь, и там же в `"dest"` укажите еще раз его же, но добавив в конец `":443"`
- `inbounds -> streamSettings -> realitySettings -> privateKey` сюда вставьте ваш приватный ключ, который вы сгенерировали ранее

Сохраните файл. Готово! Можно пробовать.

Перезапустите XRays командой `systemctl restart xray`. Сразу после этого можно проверить что все нормально командой `systemctl status xray` (должно быть написано `active (running)`), а если он не запускается или что-то идет не так, то можно посмотреть сообщения об ошибках и логи командой `journalctl -u xray`. Обычно дело

в лишней или пропущенной запятой в конфиге (можно использовать онлайн-JSON-валидаторы для проверки).

Теперь нам надо собрать ссылку для клиента, чтобы можно было легко и быстро добавлять подключение к вашему прокси на всевозможных устройствах и раздавать доступ родным и друзьям. Ссылка будет выглядеть так:

```
vless://ваш_UUID@IP_адрес_вашего_сервера:443/?  
encryption=none&type=tcp&sni=домен_сайта&fp=chrome&security=reality&alpn=h2&flow=xtls-rprx-vision&pbk=ваш_публичный_ключ&packetEncoding=xudp
```

Соответственно, вместо `ваш_UUID` подставляете UUID, дальше после собаки - IPv4 адрес вашего сервера, `домен_сайта` - фейковый домен, под который вы маскируете, `ваш_публичный_ключ` - публичный ключ, который вы сгенерировали выше, а в конце после символа `#` можно написать название вашего сервера, которое увидит пользователь (латинскими буквами, без пробелов).

И пусть вас не смущает `encryption=none`, ваши данные будут надежно зашифрованы, этот параметр просто говорит об отсутствии дополнительных слоев шифрования в VLESS, он к нему неприменим (смотрите начало статьи).

Пример:

```
vless://a69578b6-bc45-48bd-94c4-a22c10a86c9d@77.88.21.37:443/?  
encryption=none&type=tcp&sni=pornhub.com&alpn=h2&fp=chrome&flow=xtls-rprx-  
vision&security=reality&pbk=Ho5AGwe2seTxluUNgQN14FA_2PorZSPDgTEEXwiDG8&pa  
cketEncoding=xudp#MyVlessServer
```

Обратите внимание, в конфиге используется **приватный** ключ сервера, а в ссылке - **публичный**, не перепутайте.

Эту ссылку можно вставлять в разные клиенты (чаще всего допускается вставка из буфера обмена), и из нее автоматически создастся подключение в клиенте. Можете делиться этой ссылкой со своими родными и друзьями. На мобильных клиентах часто есть функция сканирования QR-кода камерой, в таком случае из этой ссылки можно сгенерировать QR-код любым онлайн-генератором кодов и использовать его.

Если что-то не работает, ссылку всегда можно вставить в клиент типа Nekray/Nekobox (и под Android тоже), и проверить, все ли там правильно, каждый параметр по-отдельности.

► [Вот так выглядит наша ссылка из примера после вставки в Nekobox](#)

▸ Не обязательно, но желательно

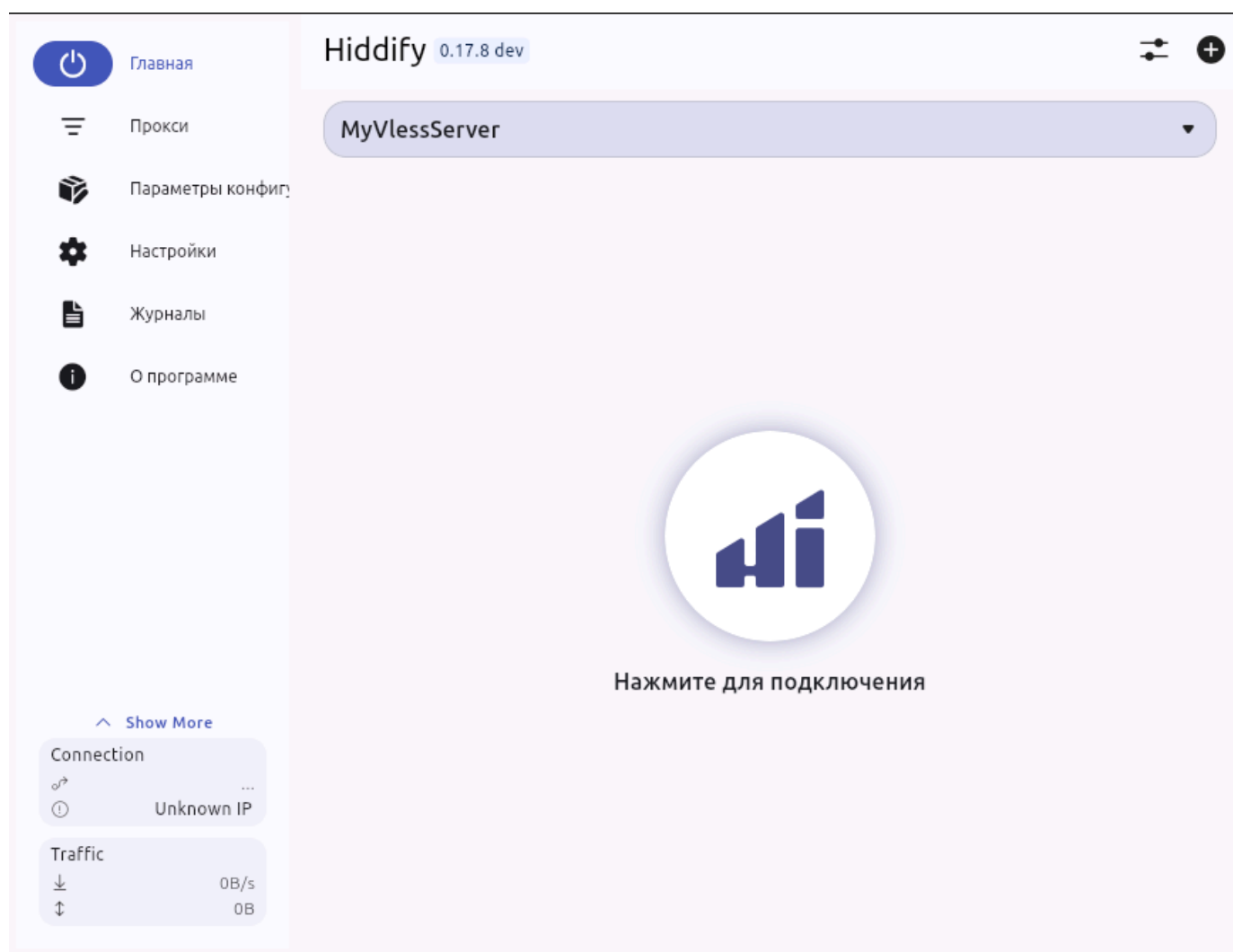
Клиенты

Самый красивый и простой в настройке: **Hiddify-Next**.

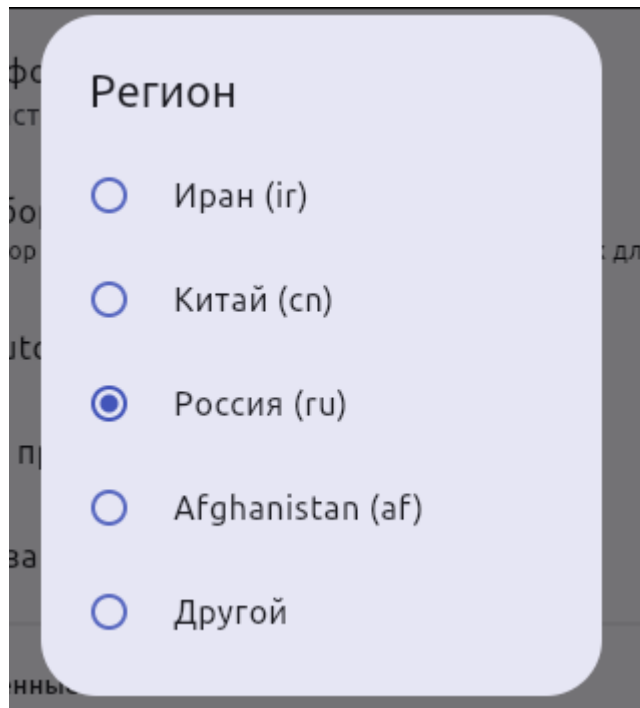
Работает под Windows, Linux, macOS, под Android, в разработке версия под macOS.

Скачать бинарник под вашу десктопную систему или .apk-файл для Android можно на странице проекта: <https://github.com/hiddify/hiddify-next/releases>

Чем он хорош? Первое - основан на sing-box и поддерживает много всего, второе - простой интерфейс (буквально одна большая кнопка "сделать зашифр" "Подключиться"):



Третье - при первом запуске он предлагает выбрать страну, и если вы выберете Россию, то автоматически применятся правила, направляющие трафик до иностранных ресурсов через ваш прокси, а до российских сайтов — напрямую, ничего дополнительно настраивать не надо (если вам не нужна эта функция, выберите опцию «Другое» в списке стран, можно поменять в настройках):



Новые прокси (ссылкой из буфера обмена) добавляются по клику на "+" справа сверху. Главное **не используйте** опцию "Ввести вручную", ссылка не вставится (это баг).

На Android интерфейс аналогичный, только более упрощенный (и можно сканировать QR с камеры).

Из других клиентов можно упомянуть

для десктопов: **Nekobox/Nekoray** (Windows, Linux, macOS), **v2rayN** (Windows), **InvisibleMan** (Windows)

для Android: **v2rayNG**, **NekoboxForAndroid**

Что же делать пользователям iPhone? Под айфоны и айпады есть клиент **Streisand**.

После импорта нужно перейти в Streisand в настройки роутинга, выбрать эту конфигурацию и активировать роутинг. Конфигарция содержит 3 правила, 1-ое для всех российских IP-адресов по GeoIP, второе для всех .ru-доменов, а третье также напрямую пропускает трафик до google, apple, telegram и whatsapp - возможно во время блокировок мессенджеров или ютуба не самая лучшая идея, в случае чего это третье правило можно удалить.

Другие достойные клиенты под iOS: **FoxRay** (бесплатный) и **Shadowrocket** (платный).

Как настраивать отдельную маршрутизацию в разных клиентах и всякие полезные лайфхаки можно найти в этой статье.

Просто: добавляем Shadowsocks

Как я уже говорил выше, XTLS-Reality при всех его отличных плюсах не идеален и в редких случаях может перестать работать в какой-то момент из-за проблем с доменом, который вы используете для маскировки.

Добавим в наш конфиг возможность подключаться через Shadowsocks, так сказать, на черный день. В секцию inbounds в нашем конфиге после нашего первого inbound'a (не забудьте добавить запятую после скобки), добавим еще один inbound:

```
...
{
  "port": 54321,
  "tag": "ss-in",
  "protocol": "shadowsocks",
  "settings": {
    "method": "2022-blake3-aes-128-gcm",
    "password": "WmhpdmUgQmVsYXJ1cyEhIQ==",
    "network": "tcp,udp"
  },
  "sniffing": {
    "enabled": true,
    "destOverride": [
      "http",
      "tls",
      "quic"
    ]
  }
}
...
```

В поле "port" впишите какой-нибудь номер порта (не тот, что в примере!), в поле password вставьте произвольный ключ длиной 16 байт в формате Base64, можно сгенерировать его командой `openssl rand -base64 16`

▸ Пример итогового конфига

Ссылка делается еще проще:

```
ss://method:password@IP_адрес_сервера:порт
```

Пример для конфига выше:

```
ss://2022-blake3-aes-128-  
gcm:WmhpdmUgQmVsYXJ1cyEhIQ==@77.88.21.37:54321#MyShadowsocksServer
```

Эту ссылку тоже можно копипастить в клиенты, генерировать из нее QR-код и делиться с друзьями.

Просто: настройка SSH-прокси

И добавим еще про запас SSH. SSH-сервер вы уже перевесили на нестандартный порт, как я советовал выше (не 22, ведь перевесили же, правда?). Больше никаких особых действий не требуется, но я бы посоветовал создать на сервере отдельного системного пользователя для SSH-прокси, ограничив его в правах, чтобы подключаясь к серверу логином-паролем клиент не мог выполнять никаких команд в системе. Сделать это можно так:

```
adduser --no-create-home --shell /bin/true proxyclient # создали юзера
```

Готово!

Ссылка может быть такой:

```
ssh://proxyclient:пароль@IP_адрес_сервера:порт#MySshServer
```

такую ссылку понимает только Streisand и больше никто.

В Hiddify-next, к сожалению, есть баг, ссылка вставляется, подключение создается, но не работает. Пока баг не исправят, на Андроиде можно использовать вышеупомянутый [NekoboxAndroid](#) или [SSH Custom](#), создав там подключение с типом SSH и заполнив логин, пароль, адрес сервера и порт руками.

В десктопном Nekobox (не забудьте выбрать Core = sing-box в Basic Settings) тоже можно добавлять SSH-подключения, там нет пункта "SSH" в списке типов, но можно выбрать "Custon (sing-box outbound)" и ввести конфиг вручную:

```
{
  "type": "ssh",
  "server": "IP_адрес",
  "server_port": ваш_порт,
  "user": "proxycient",
  "password": "ваш_пароль",
  "client_version": "SSH-2.0-OpenSSH_7.4p1"
}
```

Просто: увеличиваем скорость

Можно настроить на сервере Bottleneck Bandwidth и Round-trip propagation time (BBR) congestion control algorithm. В файл `/etc/sysctl.conf` вписать

```
net.core.default_qdisc=fq
net.ipv4.tcp_congestion_control=bbr
```

и потом выполнить команду `sysctl -p`

Чуть сложнее: добавляем mKCP

mKCP, хоть и не новый, хорош тем, что он работает по UDP, оптимизирован для работы под плохие каналы, а еще он умеет маскировать свои пакеты под SRTP (FaceTime), WebRTC (звонки в мессенджерах) и uTP (Bittorrent), поэтому заслуживает внимания. Один минус - он не поддерживается в клиентах на базе Sing-box, то есть в Hiddy-next работать не будет, как и Nekobox. Зато можно настроить его в Nekoray (с ядром XRay), v2rayN, v2rayNG, и т.д.

Добавьте еще один inbound:

```
"inbounds": [
  ...
  {
    "port": ПОРТ,
    "protocol": "vless",
    "tag": "kcp-in",
    "settings": {
      "clients": [
        {
          "id": "ВАШ_UUID",
```

```
        "email": "user1"
    },
    ],
    "decryption": "none"
},
"streamSettings": {
    "network": "kcp",
    "kcpSettings": {
        "mtu": 1350,
        "tti": 20,
        "uplinkCapacity": 10,
        "downlinkCapacity": 20,
        "congestion": false,
        "readBufferSize": 1,
        "writeBufferSize": 1,
        "header": {
            "type": "utp"
        },
        "seed": "MySecretPassword"
    }
},
"sniffing": {
    "enabled": true,
    "destOverride": [
        "http",
        "tls",
        "quic"
    ]
}
},
...

```

"port" - любой нестандартный порт. "id" - ваш UUID, как в других примерах с VLESS.

"header" - можно выбрать srtp, utp, webrtc. "seed" - придумайте какой-нибудь пароль для шифрования.

Ссылка для клиентов, которые поддерживают mKCP:

```
vless://ваш_UUID@IP_адрес_вашего_сервера:порт/?
encryption=none&type=kcp&security=none&seed=ВАШ_ПАРОЛЬ&headerType=ваш_header_t
ype&packetEncoding=xudp
```

Пример:

```
vless://a69578b6-bc45-48bd-94c4-a22c10a86c9d@77.88.21.37:443?  
security=none&encryption=none&seed=test&headerType=srtp&type=kcp#MyKcpServer
```

Сложно: переадресация на Cloudflare WARP

Считается, что если у вас есть прокси за границей, то лучше не ходить на него на сайты расположенные в РФ или имеющие отношение к ее компаниям. Логика простая: если цензоры могут анализировать ваш трафик, плюс трафик подконтрольных им ресурсов, то сопоставив факт "подключение от вас на иностранный IP-адрес, и подключение с этого IP-адреса на сайт внутри страны, ровно в один момент и с одинаковыми объемами принятых-переданных данных во времени" почти однозначно скажет, что на этом IP-адресе есть прокси.

Поэтому часто советуют настраивать отдельную маршрутизацию на клиентах (доступ через прокси только на иностранные сайты, а к ресурсам внутри страны напрямую), но что делать если пользователь не осилил это сделать, или что-то не сработало? Можно, конечно, отправлять на прокси обращения к российским доменам и российским IP-адресам в блок, но это не самая user-friendly тактика.

Зато мы можем сделать следующее. Есть Cloudflare Warp - бесплатный VPN-сервис для всех желающих от известной телеком-компании. Он основан на Wireguard, Wireguard блокируется в России, но в иностранной юрисдикции, где у вас стоит сервер, проблем быть не должно. XRay может работать как wireguard-клиент, и может переадресовывать на другие сервера запросы, попадающие под определенные критерии, следовательно, если мы сложим этот пазл, то мы сможем выпускать определенные запросы в Интернет с другого IP-адреса, принадлежащего Cloudflare.

Заодно мы можем заворачивать на WARP запросы к сервисам, которые не пускают к себе пользователей с адресов хостеров, например, ChatGPT. Давайте попробуем сделать это.

Вам надо сгенерировать креды доступа к Cloudflare WARP. В интернете есть много скриптов для этого, например этот, а в мобильном клиенте NekoboxForAndroid есть даже специальная фишка в меню Tools, которая это делает. Для примера я уже сгенерил реальные данные, но сколько они будут работать после публикации на Хабре отвечать не могу :)

Итак, у вас есть реквизиты для доступа к WARP: адрес сервера, порт, локальные ipv4/ipv6 адреса, приватный ключ, публичный ключ, MTU, и строка под названием "Reserved". Создадим outbound в конфиге XRay для этого:

```
"outbounds": [  
  ...
```

```
{
  "protocol": "wireguard",
  "tag": "warp",
  "settings": {
    "secretKey": "KCvEagg0f0HomfZTt/CnnOwNkU0amQWdpdWLFyq6n1U=",
    "address": [
      "172.16.0.2/32",
      "2606:4700:110:86fc:8fa7:2f45:cf12:8db/128"
    ],
    "peers": [
      {
        "endpoint": "engage.cloudflareclient.com:2408",
        "publicKey": "bmXOC+F1FxEMF9dyiK2H5/1SUtzH0JuVo51h2wPfgyo="
      }
    ],
    "mtu": 1280,
    "reserved": "WPM9",
    "workers": 2,
    "domainStrategy": "ForceIP"
  }
},
...
]
```

А потом добавим условия в секцию "routing"->"rules":

```
"routing":
{
  "rules":
  [
    ...
    {
      "type": "field",
      "domain": ["geosite:openai", "geosite:category-gov-ru", "domain:ru"],
      "outboundTag": "warp"
    },
    {
      "type": "field",
      "ip": ["geoip:ru"],
      "outboundTag": "warp"
    },
    ...
  ]
}
```



```
]
}
```

Перезапускаемся, и все работает. Если не работает - смотрим логи, почему. Можно для теста открыть какой-нибудь 2ip.ru и сравнить его результат с 2ip.io.

На клиенте ничего дополнительно настраивать не надо.

Полный список доступных категорий geoip и geosite можно найти вот в этой репе: v2fly/domain-list-community: Community managed domain list. Generate geosite.dat for V2Ray. (github.com)

Сложно: steal-from-yourself, или если у вас есть свой сайт. А еще subscriptions

Возможно вы не хотите зависеть от чужих сайтов и доменов, а хотите использовать свой, и готовы ради этого купить домен, настроить веб-сервер и сертификаты. Или даже возможно у вас уже есть какой-нибудь свой сайт на своем VPS или любой другой веб-сервис (например, файлохранилка). И вы хотите на тот же сервер заодно установить прокси, и это отличный вариант.

Решение простое: вы буквально будете маскироваться под свой собственный сайт. Допустим, вы установили на сервер веб-сервер, зарегистрировали домен, получили на этот домен сертификат с помощью let's encrypt или еще как, настроили сервер с этим сертификатом, наполнили сайт контентом или установили там какой-то сервис. Веб-сервер у вас слушает на порту 443.

Чтобы добавить прокси, вам надо перевесить веб-сервер на какой-нибудь другой порт, например 8443, и сделать так, чтобы он слушал не на всех IP-адресах (0.0.0.0), а только на localhost (127.0.0.1).

После этого настраиваете XRay с XTLS-Reality так, как было описано в самом начале, только в "serverNames" указываете не какой-то чужой, а ваш домен, а в "dest" пишете "127.0.0.1:8443".

После этого у вас на 443 порту по-прежнему доступен ваш сайт/сервис, но вместе с тем вы можете подключаться к нему как к прокси с XTLS-Reality и ни от кого не зависеть.

Идеальная маскировка.

И еще один плюс такой схемы. На своем веб-сервере мы можете разместить все используемые ссылки для подключения к прокси ("vless://", "ss://" и т.д.) где-нибудь в текстовом файле, ограничив доступ к нему файлом robots.txt, чтобы его не индексировали

поисковики. После этого вы можете раздавать знакомым и друзьям и вставлять в клиенты уже не эти ссылки, а ссылку на этот файл - прокси-клиент скачает файл с вашего сервера и добавит все ссылки из него к себе. Этот механизм обычно называют "Subscriptions". А если что-то поменялось, то достаточно нажать на кнопку типа "Обновить подписки", и клиент скачает обновление, а некоторые клиенты даже умеют делать это автоматически. Удобно, да?

Сложно: VLESS over HTTP/2

XRay, как и многие другие прокси, при стандартной настройке на каждое подключение от вашего компьютера (браузера) к какому-либо удаленному серверу создает новое подключение от вашего компьютера (прокси-клиента) к прокси-серверу. Условно говоря, если вы в браузере открываете сайт, при этом браузер обращается к 5 хостам в Интернете, от вашего прокси-клиента к вашему прокси-серверу будет установлено 5 соединений (и еще одной для DNS-резолвов). Некоторые считают это неэффективным. В прокси-клиентах существует режим "Mux" для мультиплексирования соединений, но он работает не всегда. Например, если у вас и на клиенте и на сервере XRay, то он будет работать без проблем, если у вас и на клиенте и на сервере Sing-box, то тоже, а вот если у вас на клиенте Sing-box, а на сервере XRay, то после включения этого режима наоборот все перестанет работать.

Но есть решение, которое работает со всеми клиентами и серверами - HTTP/2 в качестве транспорта. HTTP/2 имеет мультиплексирование на уровне протокола, и прокси будет его использовать, в итоге вместо множества подключений от прокси-клиента к прокси-серверу будут висеть всего одно или два. Единственный минус, пока что HTTP/2-транспорт не совместим с XTLS-Vision, нужно будет убрать поле "flow" и на клиенте и на сервере. Не буду вдаваться в подробности, это все описано в документации и в примерах XRay, просто приведу пример inbound'a в конфиге и ссылки для такого подключения:

```
"inbounds": [  
  {  
    "listen": "IP_адрес_вашего_сервера",  
    "port": 443,  
    "protocol": "vless",  
    "tag": "reality-in",  
    "settings": {  
      "clients": [  
        {  
          "id": "ваш_UUID",  
          "email": "user1"  
        }  
      ],  
    },  
  },  
]
```

```
"decryption": "none"
},
"streamSettings": {
  "network": "http",
  "httpSettings": {
    "host": ["somefakedummytext.com"],
    "path": "/PtnPnh",
    "read_idle_timeout": 10,
    "health_check_timeout": 15,
    "method": "GET"
  },
  "security": "reality",
  "realitySettings": {
    "show": false,
    "dest": "ваш_маскировочный_домен:443",
    "xver": 0,
    "serverNames": [
      "ваш_маскировочный_домен"
    ],
    "privateKey": "ваш_ПРИВАТНЫЙ_ключ",
    "minClientVer": "",
    "maxClientVer": "",
    "maxTimeDiff": 0,
    "shortIds": [""]
  }
},
"sniffing": {
  "enabled": true,
  "destOverride": [
    "http",
    "tls",
    "quic"
  ]
}
},
```

Ссылка: `vless://ваш_UUID@IP_адрес_вашего_сервера:443/?
encryption=none&type=http&sni=домен_сайта&host=somefakedummytext.com&path=%2FPtnPnh&fp=chrome&security=reality&alpn=h2&flow=xtls-rprx-
vision&pbk=ваш_публичный_ключ&packetEncoding=xudp`

Сложно: работа через CDN

CDN могут быть отличным запасным вариантом, когда ваш сервер попал под блокировку или возникли какие-то проблемы с Reality. Прямо скажем, вероятность бана CDN существенно ниже, чем какого-то отдельного адреса или домена.

Напрямую с VLESS/XTLS/SS работать через CDN нельзя, но мы можем использовать WebSocket-транспорт, который отлично пролезает через популярные CDN такие как Cloudflare, GCore, Amazon Cloudfront, Fastly и другие. Кстати, ещё вебсокеты хорошо пролезают через всякие корпоративные системы фильтрации интернета в офисах.

Настраивается оно довольно просто, если бы не одно но. У нас на 443 порту уже висит XTLS-Reality, маскируясь под чужой сайт, 80 порт тоже лучше не занимать чем-то иным (будет странно, если на разных портах отвечают разные сайты), поэтому встает вопрос, как совместить на одном сервере и то и то. В своей статье MiraclePtr предлагал для этого довольно сложные варианты с SNI-прокси или цепочки, когда запрос с XRay уходит на Nginx, потом обратно на XRay, и все это с сертификатами. Мы же существенно упростим схему.

Итак, доступны два варианта, и вам надо решить, какой именно будете использовать. Вариант первый, если 1) у вас есть домен (любой, хоть самые всратый .ru с распродажи за 100 рублей) и 2) у вашего сервера есть IPv6 адрес (а он сейчас есть почти всегда) - вы можете проксироваться через Cloudflare.

Вариант второй, если у вас есть карточка иностранного банка или кто-нибудь у кого можно ее попросить (платить не надо, она нужна для регистрации бесплатного аккаунта на Amazon Cloud, деньги списывать не будут, там в бесплатном тарифе включен один *терабайт* трафика) - то вы можете проксировать через Amazon Cloudfront. Домен покупать не нужно.

Настраиваются об эти варианта со стороны сервера почти одинаково.

Дополнительный Inbound будет выглядеть как-то так:

```
"inbounds": [  
  ...  
  {  
    "port": ПОРТ,  
    "listen": "IP_АДРЕС",  
    "protocol": "vless",  
    "tag": "ws-in",  
    "settings": {  
      "clients": [  
        {  
          "id": "ваш_UUID",
```

```
    "email": "user1"
  },
  ],
  "decryption": "none"
},
"streamSettings": {
"network": "ws",
  "wsSettings": {
    "path": "/PtnPnh"
  }
},
"sniffing": {
  "enabled": true,
  "destOverride": [
    "quic",
    "http",
    "tls"
  ]
}
}
```

"/PtnPnh" в "path" надо заменить на что-то свое, уникальное (любая строка из латинских букв и цифр, путь URL). gprx-xtls-vision не используется, он не работает через CDN. А вот дальше внимательно. В зависимости от того, какую схему будете использовать.

Если используете **Cloudflare**, то: в поле "listen" пишете **IPv6**-адрес вашего сервера, а порт ставите 80. Обратите внимание, IPv6-адрес в конфиге XRay должен быть в фигурных скобках, например "[aaaa:bbbb:cccc:1]".

Если используете **Amazon**, то: в поле "listen" пишете IPv4-адрес вашего сервера (как с Reality), а порт ставите какой-нибудь нестандартный, например 33380.

Почему так? Cloudflare умеет проксировать IPv4 в IPv6, но номера портов поддерживает только стандартные. Amazon не умеет работать с IPv6-origins, зато не возражает против нестандартного порта (о которых цензор даже и не узнает).

Теперь надо настроить саму CDN. Начнем с Cloudfare. Регистрируетесь там, регистрируете домен (где угодно), добавляете домен в Cloudflare и делегируете его на те сервера, которые вам скажет CF. У CF отличный хелп, а если что-то не ясно, в интернете есть тонна инструкций обо всем этом.

Дальше. В **"DNS"** в интерфейсе Cloudfare Dashboard создайте запись типа AAAA для вашего домена, можно для главного (@), можно для какого-то поддомена, обязательно

поставьте активной галочку Proxy status и сохраните:

Type

AAAA

Name (required)

@

Use @ for root

IPv6 address (required)

aaaa:bbbb:cccc::1

Proxy status

☒ Proxied

TTL

Auto

Не сомневайтесь, что у домена есть только AAAA (IPv6) запись. Cloudflare умный, и с включенными проксированием заполнит и поля A, и поля AAAA для домена своими серверами, и будет проксировать IPv4 в IPv6, если у клиента нет IPv6.

Далее, на вкладке "SSL", выберите режим "Flexible". При нем Cloduflare будет принимать запросы от вашего прокси по TLS, автоматически сгенерировав для домена валидный сертификат и подписав его своим корневым сертификатом, а вот к вашему серверу будет идти по обычному HTTP по IPv6. Мы так сделали для упрощения настройки сервера, это только для трафика между CF и вашим сервером, цензоры его никогда и не увидят.

✓

Your SSL/TLS encryption mode is Flexible

This setting was last changed a few seconds ago

Browser

Cloudflare

Origin Server

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

В заключение на вкладке "Network" проверьте, что включены IPv6 и вебсокеты:

IPv6 Compatibility

Enable IPv6 support and gateway.

This setting was last changed 2 years ago

✓

API

Help

WebSockets

Allow WebSockets connections to your origin server.

Concurrent connection guidelines for your plan: low. [Learn more.](#)

✓

API

Help

https://habr.com/ru/articles/799751/

32/41

На этом настройка окончена. Можете попробовать зайти браузером на свой домен, который вы прописали в CF - должна отобразиться 404 ошибка (это нормально), но без ошибок сертификатов.

Теперь переходим к **Amazon Cloud**. Регистрируетесь, привязываете карточку (можно иметь на ней нулевой баланс, деньги не списывают). Идете в **Cloudfront**, там нажимаете "Create distribution" и заполняете поля:

Origin

Origin domain

Choose an AWS origin, or enter your origin's domain name.

Protocol | Info

- ☒ HTTP only
- ☐ HTTPS only
- ☐ Match viewer

HTTP port

Enter your origin's HTTP port. The default is port 80.

HTTPS port

Enter your origin's HTTPS port. The default is port 443.

Minimum origin SSL protocol | Info

The minimum SSL protocol that CloudFront uses with the origin.

- ☒ TLSv1.2
- ☐ TLSv1.1
- ☐ TLSv1
- ☐ SSLv3

В поле "Origin domain" Amazon требует именно домен, просто IP-адрес ввести нельзя. У вас домена может не быть, он особо и не нужен. Идите на dynu.org или любой другой DynDNS-сервис, зарегистрируйтесь там, создайте бесплатный dynamic dns домен, направив его на IPv4-адрес вашего сервера - Amazon схавает такой бесплатный домен вообще без проблем.

В поле HTTP Port впишите не 80 порт (на картинке неправильно), а тот нестандартный порт, который вы выбрали на сервере в новом inbound. Дальше пройдите по

параметрам, отключите всевозможные Origin Shield, firewall, кэширование (CachingDisabled), завершите создание домена кнопкой "Create distribution" и подождите 5-10 минут. Amazon создаст для вас домен типа *.cloudfront.net, который вы будете использовать для подключения к вашему прокси.

Осталось сгенерировать ссылку для подключения к таким прокси (ее структура будет одинаковая и для Cloudflare, и для Amazon):

```
vless://ваш_UUID@ваш_домен:443/?  
encryption=none&type=ws&sni=ваш_домен&host=ваш_домен&path=%2Fваш_path%3Fed%3D2  
048&ed=2048&eh=Sec-WebSocket-  
Protocol&security=tls&fp=chrome&security=tls&packetEncoding=xudp
```

▶ [пример и вид в Nekobox](#)

Тут все так же, UUID, домен (ваш или от Amazon, он в ссылке встречается три раза), ваш_path - это то, что вы указали как "path" в конфиге (в примере "PtnPnh"), и в ссылке к этому path добавляется %3Fed%3D2048 - это нужно для активации websocket early data в клиентах на базе XRay, при расшифровке оно будет видно как "?ed=2048). IP-адреса сервера в ссылке нет и не должно быть. Порт всегда 443.

Список литературы и полезные ссылки

- Документация по XRay, там очень много интересного: [Configurations | Project X](https://xtls.github.io/) (xtls.github.io)
- Примеры конфигурации XRay для разных протоколов: <https://github.com/XTLS/Xray-examples>
- Статьи MiraclePtr про блокировки: раз, два, три, четыре, пять, шесть, семь, восемь, девять, десять
- No Thought is a Crime, форум энтузиастов-исследователей
- net4people - то же самое, но не в российском, а в мировом масштабе

Заключение

Вот так.

Я благодарю MiraclePtr, чьи статьи очень многому меня научили и вдохновили, ChatGPT за исправление моих кривых фраз, кота Чачу, который отвлекал меня в процессе написания статьи, а также... хотя не, про еще одного человека и что он делал я лучше здесь писать не буду, потому что это будет уже пропаганда нетрадиционных отношений, а это в современной России запрещено (как, впрочем, и распространение знаний об обходе блокировок, стоит об этом задуматься).

Теги: обход блокировок, цензура, прокси, xray, vless, sing-box, shadowsocks, vpn

Хабы: Настройка Linux, Информационная безопасность, Системное администрирование, Сетевые технологии

Редакторский дайджест

Присылаем лучшие статьи раз в месяц

Электронпочта



64

260.5

Карма

Рейтинг

@UranusExplorer


Копая глубоко и упорно

 Комментарии 68

Публикации

ЛУЧШИЕ ЗА СУТКИ


ПОХОЖИЕ





UranusExplorer

12 часов назад

Надежный обход блокировок в 2024: протоколы, клиенты и настройка сервера от простого к сложному

 Простой

 36 мин

 26K

Тutorial

https://habr.com/ru/articles/799751/

35/41

 +192 440 68

Telnov_WIKI

19 часов назад

Исследование деградации Li-ion аккумуляторов в результате “быстрой” зарядки



Простой



4 мин



19K

Аналитика

 +41 48 39

Lunathecat

вчера в 12:00

115 лет прогресса: от механического осциллографа до самодельного цифрового



Простой



9 мин



6.6K

Ретроспектива

 +37 42 13

Exosphere

вчера в 12:27

Инженеры, мы в ваших руках



Простой



6 мин



4.4K

 +29 25 13

DAN_SEA

20 часов назад

Ещё один шаг в сторону оптических наушников



Средний



10 мин



8K

Обзор

 +27 23 50

madyouth

22 часа назад

Как мы создаём редакторы документов. Ядро и его роль в кроссплатформенной разработке

🕒 10 мин 👁 1.1K

📌 +27

🔖 19

💬 1



iMonin

19 часов назад

Энергетика большой страны. Почему мы все не можем отапливаться электричеством?

🕒 28 мин 👁 10K

📌 +25

🔖 36

💬 64



sacredtree

22 часа назад

Почему рациональный выбор невозможен

🕒 8 мин 👁 3.1K

📌 +18

🔖 38

💬 7



it_union

3 часа назад

Снижение зарплат в ИТ

🕒 2 мин 👁 9.8K

📌 +17

🔖 8

💬 9



WizAlx

19 часов назад

Интеграция нативных SDK во Flutter-приложение

🕒 9 мин 👁 846

Тutorial

Перевод

📌 +15

🔖 33

💬 1

А если всё рухнет при первой атаке? Почему нужно вкладываться в ИБ-обучение сотрудников

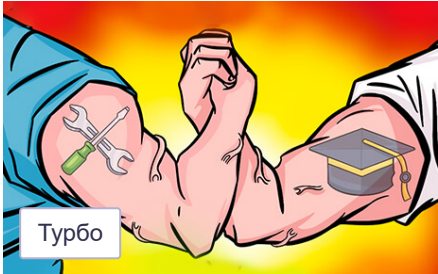
Турбо

Показать еще

МИНУТОЧКУ ВНИМАНИЯ



Выбирайте команду тестирования по вайбам




Когда сотрудник — барьер для киберпреступника





Планируй своё время и его хватит на IT-ивенты из Календаря


КУРСЫ

 L101 Администрирование Linux. Базовый курс
8 апреля 2024 · Hi-TECH Academy

 KL 008.11.6 Kaspersky Endpoint Security. Encryption - Шифрование
18 марта 2024 · Hi-TECH Academy

 KL302B Комплексный курс Лаборатории Касперского Kaspersky Endpoint Security – Продвинутый уровень
18 марта 2024 · Hi-TECH Academy

 KL 302.11 Bundle Комплексный курс Лаборатории Касперского
19 марта 2024 · Hi-TECH Academy

 KL 302.11 Kaspersky Security Center. Масштабирование
19 марта 2024 · Hi-TECH Academy

Больше курсов на Хабр Карьере

ЧИТАЮТ СЕЙЧАС

Надежный обход блокировок в 2024: протоколы, клиенты и настройка сервера от простого к сложному

26K

69

Снижение зарплат в ИТ

9.8K

9

Голодные игры начались. Развитие ИИ приведёт к естественному отбору населения

42K

195

Исследование деградации Li-ion аккумуляторов в результате “быстрой” зарядки

19K

39

Сколько мы заработали за год на 1 товаре из Китая. Продаем коврики для ноутбука на маркетплейсах

39K

65

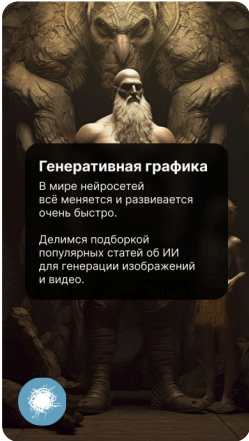
А если всё рухнет при первой атаке? Почему нужно вкладываться в ИБ-обучение сотрудников

Турбо

ИСТОРИИ



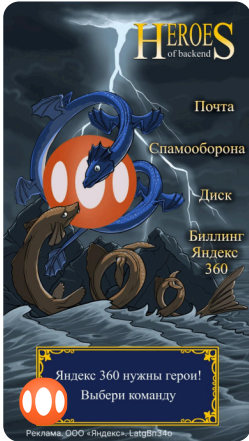
GitVerse: открой вселенную кода



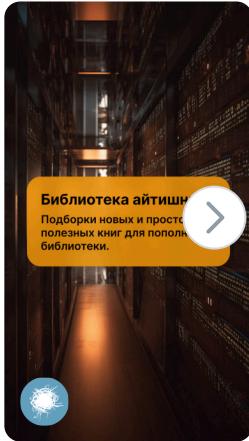
Нейросети: интересное



Что умеет калькулятор зарплат в IT



Яндекс 360 призывает героев бэкэнда



Полезные книги для библиотеки айтишника

РАБОТА

DevOps инженер
28 вакансий

Специалист по информационной безопасности


140 вакансий

Системный администратор

92 вакансии


Все вакансии

БЛИЖАЙШИЕ СОБЫТИЯ





Как решать алгоритмический блок при найме в IT?


Разбираемся на Тренировках по алгоритмам 5.0 от Яндекса



Серия занятий «Тренировки по алгоритмам 5.0» от Яндекса


 1 марта – 19 апреля

 19:00


 Онлайн


Подробнее в календаре


Хабр Карьера



Тестировщики, выбирайте себе команду по вайбам на Хабр Карьере

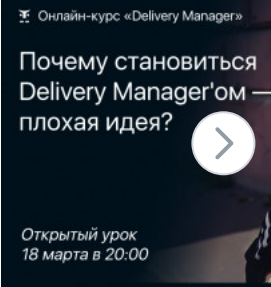
 18 – 24 марта

 09:00 – 23:00


 Онлайн


Подробнее в календаре


Онлайн-курс «Delivery Manager»



Открытый урок «Почему становиться Delivery Manager'ом — плохая идея?»

 18 марта



 Онлайн

Подробнее в календаре

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
	Авторы	Соглашение	программы
	Песочница	Конфиденциальность	Стартапам



Настройка языка

Техническая поддержка

© 2006–2024, Habr