Настройка протокола mKCP в 3X-UI и X-UI для маскировки трафика прокси-сервера под BitTorrent, FaceTime и WebRTC

**Простой**

**5 мин**

**2.3K**

[Информационная безопасность\*](https://habr.com/ru/hubs/infosecurity/)[Системное администрирование\*](https://habr.com/ru/hubs/sys_admin/)[Сетевые технологии\*](https://habr.com/ru/hubs/network_technologies/)

Туториал

Сегодня мы поговорим о настройке подключения к прокси-серверу по протоколу mKCP в известных web-панелях X-UI и 3X-UI. Про mKCP, как и про многие другие актуальные на сегодняшний день прокси- и VPN-протоколы я недавно рассказывал в статье ["Надежный обход блокировок в 2024: протоколы, клиенты и настройка сервера от простого к сложному"](https://habr.com/ru/articles/799751/) (она заблокирована Роскомнадзором на территории РФ, но открывается на Хабре через иностранный прокси или VPN). Про сервера (так же в народе называемые "панелями") X-UI и 3X-UI рассказывал всем известный юзер MiraclePtr в своей статье ["3X-UI: Shadowsocks-2022 & XRay (XTLS) сервер с простой настройкой и приятным интерфейсом"](https://habr.com/ru/articles/735536/). Ну а мы сегодня совместим и то и то.

Первый вопрос, который может возникнуть: Зачем?  
Все популярные недавние статьи описывали настройку протокола VLESS c XTLS-Reality, суть которых в том, что прокси-сервер искусно маскируется под какой-то популярный сайт, имитируя его поведение и представляясь его настоящим TLS-сертификатом. На сегодняшний день это по-прежнему самое мощное и продвинутое средство обхода блокировок с помощью прокси. Но в мире ничто не идеально, и этот вариант тоже имеет недостатки:

1. В случае блокировки ресурса, под который вы маскируетесь, или просто каких-либо технических проблем на нем, перестанет работать и ваш прокси (например, популярный у многих для этих целей microsoft.com на некоторое время отключал TLS 1.3, в итоге многие люди не могли подключиться к своим проксикам).
2. Из-за использования TLS хендшейк при установлении соединения здесь ощутимо дольше, чем в других протоколах, что влияет на комфортность серфинга. В некоторых случаях помогает смена маскировочного домена (тот сервер, под который вы маскируетесь, должен находиться в той же стране, где и ваш VPS, а в идеале в том же дата-центре), но для некоторых время установления соединения может быть критичным.
3. VLESS с XTLS *всегда* должен работать на 443 порту. В некоторых случаях (вы делите сервер с кем-то еще, или подключаетесь через строгий корпоративный прокси который делает MitM) это может вам не подойти.

Для решения проблем 1 и 2 в качестве запасного варианта часто в инструкциях даются советы по настройке дополнительного подключения по протоколу ShadowSocks-2022, чтобы вы всегда могли подключиться к своему прокси-серверу, даже если с маскировкой через XTLS-Reality что-то случится. Shadowsocks - отличная штука, но как показало время, в случае блокировок по белым спискам протоколов (чем развлекался Роскомнадзор некоторое время назад, и что иногда делают в Китае согласно последнему GFW-Report), он тоже перестанет работать - просто потому, что Shadowsocks не похож ни на один из существующих протоколов, и в белом списке его точно не будет.

И тут нам на помощь приходит mKCP. mKCP - это вариант протокола KCP, который вообще изначально был придуман для работы через плохие каналы связи с большим количеством потерь пакетов (например, когда вы сидите через 3G/4G и в вашем районе сильно перегружена сеть в пиковые часы). Для нас же он ценен другими своими свойствами:

1. Он очень прост в настройке. Не надо искать никаких маскировочных доменов, думать о сертификатах, и т.д.
2. Он работает не через TCP, а через UDP. Как, опять же, показывает практика, нередко цензоры сосредотачиваются именно на фильтрации TCP, а про UDP забывают, поэтому очень хорошо иметь такую альтернативу;
3. Он может маскировать заголовки своих пакетов под uTP (BitTorrent), DTLS (используется в WebRTC, например, для звонков в мессенджерах), SRTP (Facetime) и WeChat.
4. Как бонус (на самом деле это было его основное предназначения), при тонкой настройке на сервере и на клиенте (параметры uplink/downlink) он может сильно выручить при работе через нестабильный канал связи.

Неплохо, да?

Недостатки, правда, тоже есть. mKCP поддерживается только в клиентах, основанных на XRay.  
Что будет работать: v2RayN (Windows), Nekoray (с ядром XRay, его можно выбрать в настройках), v2RayNG (Android), Streisand (iPhone) и другие клиенты, основанные на XRay. Что *не* будет работать: Hiddify-Next, Nekobox for Android, и другие клиенты на базе Sing-box.

Ну и да, нужно уточнить, что mKCP в нашем случае - это лишь транспортный протокол, а внутри него аутентификация и обмен данными производится по уже всем привычному VLESS.

Предвижу вопрос: можно ли одноврменно пользоваться разными протоколами на сервере? Ответ: да, можно, но лучше не смешивайте XTLS-Reality с остальным. То есть если вы планируете использовать XTLS-Reality, то подключайтесь всегда именно через него, а Shadowsocks и mKCP оставьте на черный день, если вдруг что. Или наоборот, если Shadowsocks и mKCP у вас работают стабильнее и отзывчивее, используйте один из них, а Reality пусть будет про запас.

Настройка подключения в панели

Настраивается все не просто, а очень просто.

Дано: у вас есть установленная панель 3X-UI или X-UI, например, [по инструкции от MiraclePtr](https://habr.com/ru/articles/735536/). В ней уже у вас могут быть созданы inbounds для VLESS+Reality или Shadowsocks, а теперь нам надо добавить новый inbound для mKCP. Я буду показывать на примере 3X-UI, в X-UI все должно быть аналогично, если вдруг вы обнаружите, что что-то сильно отличается, напишите в комментариях, разберемся.

На вкладке inbounds нажимаем Add Inbound и заполняем:

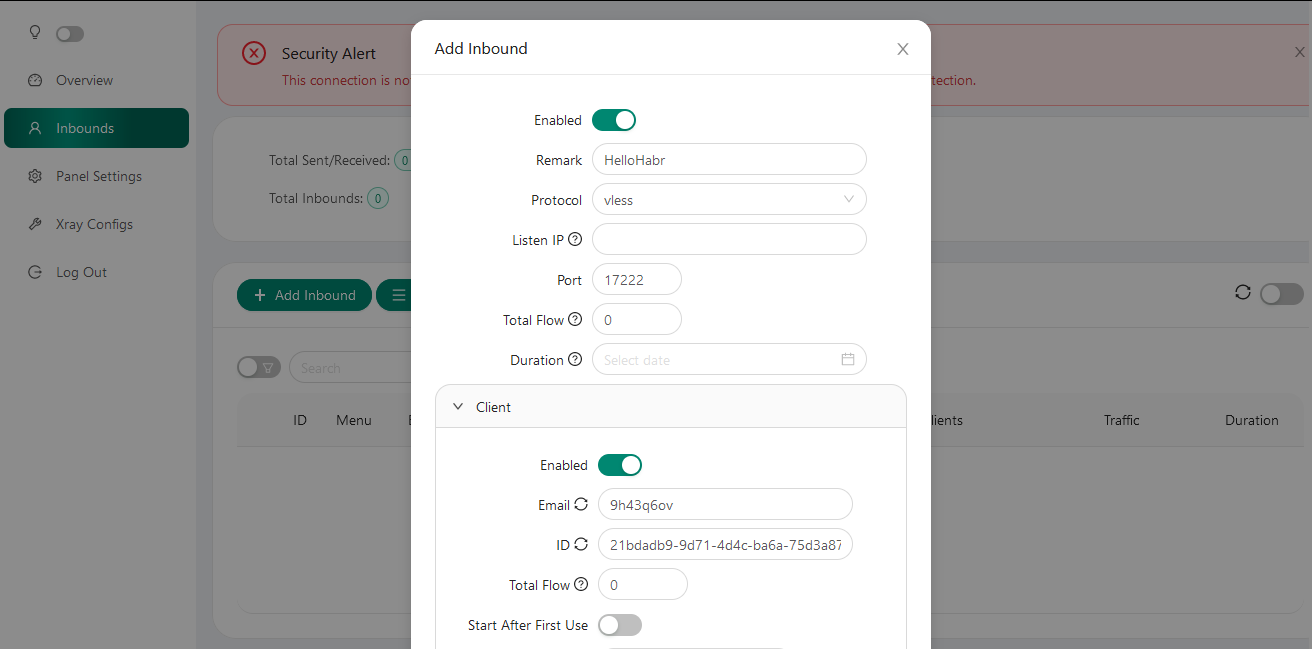
**Remark** - название, может быть все что угодно (у меня HelloHabr)

**Protocol** - "vless" (это протокол для аутентификации, он будет завернут в mKCP)

**Listen IP** - можно оставить пустым (слушать на всех адресах), или указать конкретный IP-адрес вашего сервера, на котором надо принимать подключения

**Port** - панель его сгененрирует автоматически, можно использовать любой. Одно но: Tor c транспортом Snowflake использует порты 30000-60000, поэтому если вы будете маскировать под WebRTC, выберите порт пониже, ну так, на всякий случай, мало ли что, вдруг у роскомпозора обострение шизы будет :)

**Client** - панель создаст вам нового пользователя с UUID.



Продолжаем:

**Transmission** - вот тут важно, выбираем mKCP

**Obfuscation** - выбираем, под что маскироваться, например uTP (это Bittorrent) или DTLS (это WebRTC), или еще что. Конкретного совета "под что лучше" я вам дать не могу, но никто не запрещает создать несколько inbound'ов на разных портах с разными типами обфускации. Главное не выбирать "wireguard" :)

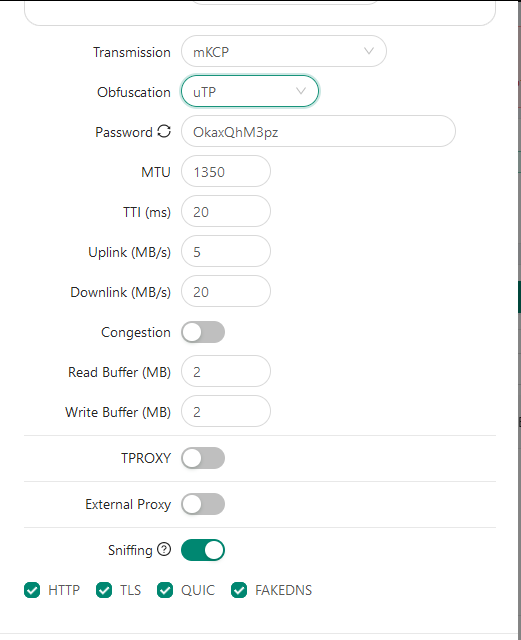
Также в поле Obfuscation можно не выбирать ничего - в таком случае у вас на прокси будет трафик типа "ни на что не похожий набор байт по UDP" (типа как в Hysteria с обфускацией), в каких-то случаях это может оказаться вполне работоспособным вариантом - мы же не знаем, что там отечественным государственным цензорам в голову взбредёт.

**Password** - панель сгенерирует пароль автоматически, он будет использовать как ключ для шифрования передаваемых данных.

**MTU, TTI** - оставляем по умолчанию. MTU можно будет потом попробовать уменьшить, если будут проблемы с доступом к каким-то некоторым ресурсам, но стандартное 1350 должно работать без проблем.

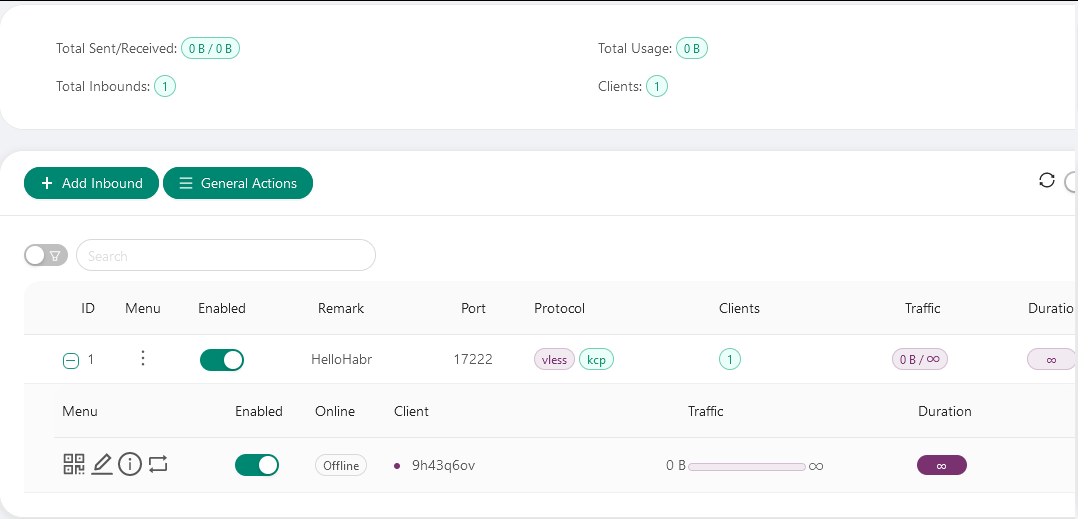
**Uplink/Downlink** - для начала можно оставить по умолчанию. По идее, там должны быть значения, соответствующие реальному каналу сервера, и - важно - в мега*байтах*, а не в мега*битах*! То есть если у вас у сервера канал 100 мегабит, должно быть 12/12. Но, как я говорил, у меня и с настройками по-умолчанию 5/20 все отлично работало, а изменять эти значения есть смысл только если у вас все работает совсем медленно или нестабильно.  
Если вам важна работа через плохие каналы связи, то на стороне клиента нужно будет настроить эти же параметры в соответствия с параметрами канала связи и возможно поиграться с параметром Congestion. Если у вас нет проблем с подключением, то не трогаем эти значения и на клиенте тоже.

**Read Buffer / Write Buffe**r - можно начать со значения по умолчанию в 2 мегабайта. Чем больше значения, тем выше может быть скорость передачи данных, но тем больше памяти будет потреблять прокси-сервер.



Все остальное по умолчанию, как на скриншоте.

Сохраняем настройки, и видим, что у нас создался наш новый inbound:



В некоторых версиях панели нужно будет еще перезагрузить XRay или саму панель.

После этого можно раскрыть поле с клиентами для нашего нового inbound, и точно так же как и для VLESS/Reality и Shadowsocks, там можно получить строку подключения или QR-код, которые можно вбить/отсканировать в ваш клиент с поддержкой mKCP и подключиться к прокси.