راهنمای کاربری پنل 3X-UI

العربية · English · Español · فارسی · Bahasa Indonesia · 日本語 · Português · Русский
Türkçe · Українська · Tiếng Việt · 简体中文 · 繁體中文

نسخه 3.4.2 :3X-UI. این راهنما بر اساس این نسخه تهیه شده و برای آن به‌روز است. خلاصه تغییرات 3.4.2 نسبت به 3.4.1 در بخش «تازه‌های نسخه 3.4.2» آمده است.

> راهنمای جامع فارسی برای وب‌پنل **3X-UI** (مدیریت Xray-core): قابلیت‌ها، پیکربندی و بهره‌برداری، با توضیح تک‌تک فیلدها و کلیدهای تبدیل در رابط کاربری.
>
> نام‌ها و برچسب‌ها با رابط کاربری پنل مطابقت دارند. واژه‌های *inbound* و *outbound* ترجمه نمی‌شوند.

فهرست مطالب

- تازه‌های نسخه 3.4.2
- 1. معرفی، پیش‌نیازها و نصب
  - 1.1. 3X-UI چیست
  - 1.2. سیستم‌عامل‌ها و معماری‌های پشتیبانی‌شده
  - 1.3. روش‌های نصب
  - 1.4. اولین اجرا و اعتبارنامه‌های پیش‌فرض
  - 1.5. محل فایل‌ها
  - 1.6. دستور مدیریتی `x-ui` (منوی اسکریپت)
  - 1.7. زیردستورهای `x-ui` (بدون منوی تعاملی)
  - 1.8. مهاجرت از SQLite به PostgreSQL
- 2. ورود به پنل و امنیت دسترسی
  - 2.1. فرم ورود
  - 2.2. احراز هویت دومرحله‌ای (2FA / TOTP)
  - 2.3. محدودسازی تلاش‌های ورود (login limiter / محافظت در برابر حدس زدن)
  - 2.4. تغییر لاگین و رمز عبور مدیر
  - 2.5. مسیر مخفی (مسیر webBasePath / URI) و پورت پنل
  - 2.6. طول عمر نشست (تایم‌اوت)
  - 2.7. LDAP (همگام‌سازی و احراز هویت)
- 3. نمای کلی / داشبورد
  - 3.1. اصول کلی جمع‌آوری داده
  - 3.2. پردازنده (CPU)
  - 3.3. حافظه (RAM)
  - 3.4. حافظه مجازی (Swap)
  - 3.5. دیسک (Storage)
  - 3.6. مدت زمان اجرای سیستم (Uptime)
  - 3.7. میانگین بار سیستم (Load average)
  - 3.8. شبکه: سرعت و حجم کل ترافیک
  - 3.9. آدرس‌های IP سرور
  - 3.10. اتصالات TCP/UDP
  - 3.11. وضعیت Xray و مدیریت فرآیند
  - 3.12. به‌روزرسانی پنل (3X-UI)
  - 3.13. به‌روزرسانی فایل‌های جغرافیایی (GeoIP / GeoSite)
  - 3.14. پشتیبان‌گیری و بازیابی پایگاه داده
  - 3.15. عناصر اضافی رابط کاربری
- 4. Inboundها: ایجاد و پارامترهای عمومی
  - 4.1. فیلدهای عمومی فرم
  - 4.2. Sniffing (اسنیفینگ)
  - 4.3. Allocate (استراتژی توزیع پورت‌ها)
  - 4.4. External Proxy (پروکسی خارجی)
  - 4.5. Fallbacks (فال‌بک‌ها)
  - 4.6. بازنشانی دوره‌ای ترافیک
  - 4.7. JSON ورودی (پیشرفته)
  - 4.8. عملیات روی inbound: QR / Edit / Reset / Delete و آمار

- 5. پروتکل‌ها
  - 5.1. فهرست پروتکل‌های پشتیبانی‌شده
  - 5.2. کدام پروتکل‌ها از TLS / REALITY / انتقال پشتیبانی می‌کنند
  - 5.3. VLESS
  - 5.4. VMess
  - 5.5. Trojan
  - 5.6. Shadowsocks
  - 5.7. Dokodemo-door / Tunnel (فوروارد شفاف)
  - 5.8. SOCKS / HTTP (پروتکل `mixed`)
  - 5.9. WireGuard (inbound)
  - 5.10. Hysteria (به‌طور پیش‌فرض نسخه 2)
  - 5.11. MTProto (پروکسی برای تلگرام)
  - 5.12. راهنمای سریع انتخاب پروتکل
- 6. ترانسپورت (Stream Settings)
  - 6.1. انتخاب شبکه انتقال
  - 6.2. RAW / TCP (`tcpSettings`)
  - 6.3. mKCP (`kcpSettings`)
  - 6.4. WebSocket (`wsSettings`)
  - 6.5. gRPC (`grpcSettings`)
  - 6.6. HTTPUpgrade (`httpupgradeSettings`)
  - 6.7. XHTTP / SplitHTTP (`xhttpSettings`)
  - 6.8. ترانسپورت Hysteria (`hysteriaSettings`)
  - 6.9. پارامترهای همراه
- 7. امنیت اتصال: XTLS، TLS و REALITY
  - 7.1. تفاوت: TLS در مقابل XTLS در مقابل REALITY
  - 7.2. حالت «هیچ» (`none`)
  - 7.3. حالت TLS
  - 7.4. حالت REALITY
  - 7.5. توصیه‌های عملی برای پیکربندی
- 8. کلاینت‌ها
  - 8.1. فیلدهای کلاینت
  - 8.2. اتصال به inbound
  - 8.3. عملیات روی کلاینت
  - 8.4. عملیات دسته‌جمعی
  - 8.5. جستجو، فیلتر و مرتب‌سازی
  - 8.6. نشانه‌ها و وضعیت‌ها
- 9. گروه‌های کلاینت
  - 9.1. گروه کلاینت چیست و چرا لازم است
  - 9.2. ارتباط گروه با کلاینت‌ها، inboundها، نودها و پروتکل‌ها
  - 9.3. فهرست گروه‌ها و گروه‌های «خالی»
  - 9.4. فیلدها و ستون‌های گروه
  - 9.5. ایجاد گروه
  - 9.6. تغییر نام گروه
  - 9.7. افزودن کلاینت‌ها به گروه
  - 9.8. حذف کلاینت‌ها از گروه (بدون حذف خود کلاینت‌ها)
  - 9.9. ریست ترافیک گروه

- 9.10. حذف گروه و حذف کلاینت‌های گروه
- 9.11. ارتباط با صفحه «Клиенты»
- 9.12. خلاصه اندپوینت‌های API
- 9.13. ترافیک به تفکیک گروه

- 10. اشتراک‌ها (Subscription)
  - 10.1. subId چیست و لینک چگونه ساخته می‌شود
  - 10.2. تنظیمات سرور اشتراک
  - 10.3. قالب‌های خروجی
  - 10.4. صفحه اطلاعات اشتراک و QR-کدها
  - 10.5. قالب‌های سفارشی صفحه اشتراک
- 11. Xray: مسیریابی، DNS، outbounds و افزونه‌ها
  - 11.1. ساختار ویرایشگر: تب‌ها/حالت‌ها
  - 11.2. تنظیمات اصلی (General)
  - 11.3. قوانین مسیریابی (routing)
  - 11.4. Outbounds (اتصالات خروجی)
  - 11.5. Balancerها (Balancers)
  - 11.6. DNS
  - 11.7. Fake DNS
  - 11.8. WireGuard / WARP / NordVPN
  - 11.9. پروکسی معکوس و TUN
  - 11.10. لاگ‌ها و آمار (Stats, metrics)
  - 11.11. ذخیره، راه‌اندازی مجدد و تبدیل‌های خودکار
  - 11.12. Outbound از subscription (با به‌روزرسانی خودکار)
  - 11.13. چرخش IP در WARP
- 12. گره‌ها (چندپانل، master/slave)
  - 12.1. خلاصه بالای فهرست
  - 12.2. افزودن و ویرایش گره
  - 12.3. بررسی TLS (برای گره‌های https)
  - 12.4. اطلاعات نمایش داده‌شده برای هر گره
  - 12.5. اقدامات روی گره
  - 12.6. تاریخچه معیارها
  - 12.7. نحوه همگام‌سازی inbound ها و کلاینت‌ها
  - 12.8. زنجیره گره‌ها (زیرگره‌ها / گره‌های ترانزیتی)
  - 12.9. گره‌ها: جدیدها در نسخه 3.3.0
- 13. تنظیمات پنل
  - 13.1. ذخیره و راه‌اندازی مجدد پنل
  - 13.2. تنظیمات عمومی (زبانهٔ «پنل» / *General*)
  - 13.3. دسترسی به پنل: IP، پورت، مسیر، دامنه، گواهی
  - 13.4. نشست، پراکسی پنل و پراکسی‌های مورد اعتماد (زبانهٔ «پراکسی و سرور» / *Proxy and Server*)
  - 13.5. ربات Telegram (زبانهٔ «ربات Telegram» / *Telegram Bot*)
  - 13.6. تاریخ و زمان (زبانهٔ «تاریخ و زمان» / *Date and Time*)
  - 13.7. ترافیک خارجی و رفتار Xray (زبانهٔ «ترافیک خارجی» / *External Traffic*)
  - 13.8. متفرقه: قالب پیکربندی Xray و URL بررسی
  - 13.9. حساب مدیر و توکن‌های API
  - 13.10. تغییرات API در 3.3.0 (مهم برای یکپارچه‌سازی‌ها)

- 14. ربات تلگرام
  - 14.1. فعال‌سازی و پیکربندی ربات
  - 14.2. منوی اصلی و دکمه‌ها
  - 14.3. دستورات ربات
  - 14.4. مدیریت کلاینت (فقط مدیر)
  - 14.5. اعلان‌ها و گزارش‌ها
  - 14.6. پشتیبان‌گیری و لاگ‌ها
  - 14.7. ویژگی‌های عملکرد
- 15. پایگاه‌های جغرافیایی (geoip / geosite و سفارشی)
  - 15.1. geoip.dat و geosite.dat چه هستند
  - 15.2. فایل‌های جغرافیایی استاندارد و به‌روزرسانی آن‌ها
  - 15.3. به‌روزرسانی خودکار داده‌های جغرافیایی با ابزارهای Xray (Geodata Auto-Update)
  - 15.4. اعتبارسنجی و محدودیت‌ها
  - 15.5. بررسی خودکار هنگام راه‌اندازی پنل
  - 15.6. استفاده از پایگاه‌های جغرافیایی در قوانین مسیریابی
- 16. بهره‌برداری: پشتیبان‌گیری، لاگ‌ها، به‌روزرسانی، CLI
  - 16.1. پشتیبان‌گیری و بازیابی پایگاه داده
  - 16.2. مشاهده لاگ‌ها
  - 16.3. سطح و پیکربندی لاگ‌گیری Xray
  - 16.4. مدیریت Xray: توقف و راه‌اندازی مجدد
  - 16.5. راه‌اندازی مجدد و به‌روزرسانی پنل
  - 16.6. وظایف دوره‌ای (cron)
  - 16.7. منوی کنسول و CLI (`x-ui`)
  - 16.8. حذف پنل
  - 16.9. دستور `x-ui migrateDB`

تازه‌های نسخه 3.4.2

نسخه 3.4.2 یک به‌روزرسانی بزرگ است: WireGuard به مدل چندکلاینتی منتقل شده، REALITY یک اسکنر زنده هدف به دست آورده، بالانسرها تب‌های Observatory/Burst Observatory گرفته‌اند، و تأیید تنظیمات حساس با کد 2FA اضافه شده است. در ادامه تغییرات نسبت به 3.4.1، دسته‌بندی‌شده بر اساس بخش‌های راهنما، آمده است.

تغییرات بخش 1 — معرفی، پیش‌نیازها و نصب

- در منوی کناری (و در کشوی موبایل) دکمه **«مستندات»** (آیکن کتاب) اضافه شده است — مستندات رسمی `https://docs.sanaei.dev/` را باز می‌کند.
- حداقل نسخه Xray که پنل به آن به‌روزرسانی می‌کند به **26.6.27** افزایش یافته است (هسته Xray 26.6.27 همراه آن عرضه می‌شود).

تغییرات بخش 2 — ورود به پنل و امنیت دسترسی

- هنگامی که 2FA فعال است، تغییر لاگین/رمز عبور مدیر و غیرفعال کردن 2FA اکنون نیازمند **وارد کردن کد فعلی** از برنامه احرازکننده است (تأیید تغییرات حساس).
- LDAP: کلید جدید **«رد کردن بررسی گواهی TLS»** (`ldapInsecureSkipVerify`) — بررسی گواهی هنگام LDAPS را غیرفعال می‌کند؛ تنها زمانی در دسترس است که «استفاده از TLS (LDAPS)» فعال باشد.

تغییرات بخش 3 — نمای کلی / داشبورد

- دکمه نسخه پنل اکنون همیشه پنجره به‌روزرسانی را باز می‌کند (به بخش 16 — کانال dev مراجعه کنید).
- بهبود سراسری **دسترس‌پذیری**: برچسب‌های aria برای آیکن‌ها و فعال‌سازی عناصر با Enter/Space (برای صفحه‌خوان‌ها و پیمایش با صفحه‌کلید).

تغییرات بخش 4 — Inboundها: ایجاد و پارامترهای عمومی

- اقدام **«صادر کردن همه لینک‌ها»** اکنون لینک‌ها را از طریق موتور اشتراک می‌سازد — قالب ریمارک را برای هر کلاینت اعمال می‌کند و اندپوینت‌های Host مدیریت‌شده را ترجیح می‌دهد (پیش از این ریمارک ثابت `inbound-email` بود).

تغییرات بخش 5 — پروتکل‌ها

- **WireGuard به مدل چندکلاینتی منتقل شد.** peerها اکنون کلاینت‌های معمولی هستند (با تخصیص خودکار آدرس در تونل، پشتیبانی از اشتراک‌ها، محدودیت‌های ترافیک/مدت و گروه‌ها)؛ لیست درون‌خطی «Peerها» از فرم inbound حذف شده است.
- به inbound WireGuard فیلد قابل‌تنظیم **DNS** (به‌طور پیش‌فرض `1.1.1.1, 1.0.0.1`) و **کارت پیکربندی کلاینت** اضافه شده است — کپی/دانلود/QR از `.conf` کامل و لینک `wg:///wireguard:///` .

تغییرات بخش 6 — ترانسپورت (Stream Settings)

- در XHTTP برای inboundهای جدید، پارامتر `maxConnections` در **xmux** اکنون به‌طور پیش‌فرض **6** است (قبلاً `0` — بدون محدودیت). inboundهای موجود مقدار خود را حفظ می‌کنند.

تغییرات بخش 7 — امنیت اتصال: XTLS، TLS و REALITY

- یک **اسکنر زنده هدف REALITY** اضافه شده است: دکمه‌های **«اسکن»** (بررسی «زنده» هدف فعلی) و **«یافتن هدف‌ها»** (اسکن یک دامنه یا بازه **IP/CIDR** و انتخاب هدف‌های مناسب بر اساس گواهی‌های آن‌ها). فیلدهای «هدف» و SNI هنگام اولین انتخاب REALITY اکنون خالی هستند.

تغییرات بخش 8 — کلاینت‌ها

- تمدید مدت/سهمیه از طریق `bulkAdjust` اکنون کلاینتی را که فقط به‌دلیل اتمام منابع غیرفعال شده (مدتش منقضی شده یا از سهمیه عبور کرده) **به‌طور خودکار فعال می‌کند**، اگر تمدید او را به محدوده مجاز بازگرداند. کلاینت‌های دستی غیرفعال‌شده یا همچنان تمام‌شده، غیرفعال باقی می‌مانند.

تغییرات بخش 9 — گروه‌های کلاینت

- **«بازنشانی ترافیک»** گروه اکنون **فقط شمارنده خود گروه** را صفر می‌کند؛ شمارنده‌ها، سهمیه‌ها و وضعیت کلاینت‌های جداگانه تحت تأثیر قرار نمی‌گیرند، و راه‌اندازی مجدد Xray لازم نیست. این تغییری نسبت به رفتار قبلی است (قبلاً ترافیک همه کلاینت‌های گروه بازنشانی می‌شد).

تغییرات بخش 10 — اشتراک‌ها (Subscription)

- در **هاست‌های مدیریت‌شده** فیلد **VLESS route** بازتعریف شده است: اکنون یک مقدار واحد `0–65535` است (نه فهرستی از پورت‌ها)، و واقعاً در UUID هر اشتراک «دوخته» می‌شود (raw/JSON/Clash).
- متغیر `{{EMAIL}}` (و مترادف آن `{{USERNAME}}`) در قالب ریمارک اکنون فقط روی **اولین لینک** کلاینت نمایش داده می‌شود — مانند بلوک ترافیک/مدت.

تغییرات بخش 11 — Xray: مسیریابی، outboundها، DNS و افزونه‌ها

- **بالانسرها**: صفحه به تب‌های **«تنظیمات بالانسر»** و **«Observatory»** تقسیم شده است؛ به جای JSON خام — فرم‌های Observatory و Burst Observatory (به Burst فیلد **«متد HTTP»** اضافه شده است). بالانسر Random/Round-robin با `fallbackTag` اکنون به‌طور خودکار Burst Observatory ایجاد می‌کند.
- هنگام حذف یک outbound یا بالانسر، پنل خود لینک‌های مرتبط در مسیریابی را پاک می‌کند و در دیالوگ تأیید **پیش‌نمایش پیامدها** را نشان می‌دهد.
- در قوانین مسیریابی معیار شبکه **L4** با حروف کوچک در کانفیگ نوشته می‌شود (`tcp` / `udp`)، و در جدول با حروف بزرگ نمایش داده می‌شود.
- خطاهای فرم افزودن/ویرایش بالانسر اکنون تا اولین لمس فیلد یا تلاش برای ذخیره به تعویق می‌افتند.

تغییرات بخش 12 — نودها (چندپنل، master/slave)

- اعلان «به‌صورت محلی ذخیره شد، نود آفلاین است — بعداً همگام می‌شود» اکنون فقط زمانی نشان داده می‌شود که نود واقعاً آفلاین یا خاموش باشد (قبلاً — در هر ذخیره روی نود آنلاین).

تغییرات بخش 16 — بهره‌برداری: پشتیبان‌گیری، لاگ‌ها، به‌روزرسانی، CLI

- نام فایل‌های پشتیبان اکنون شامل آدرس سرور و **تاریخ-زمان** است: `{host}_YYYY-MM-DD_HHMMSS.db` (`.dump` برای PostgreSQL)، مثلاً `panel.example.com_2026-06-27_000000.db` — هم هنگام دانلود از پنل و هم در پشتیبان‌هایی که ربات تلگرام می‌فرستد.

- می‌توان **کانال dev** به‌روزرسانی‌ها را از یک ساخت پایدار فعال کرد: دکمه نسخه همیشه پنجره به‌روزرسانی را باز می‌کند، و کلید **«کانال Dev»** با هشداری درباره ناپایداری و نبود بازگشت خودکار اضافه شده است.

1. معرفی، پیش‌نیازها و نصب

1.1. 3X-UI چیست

3X-UI یک پنل مدیریت وب متن‌باز برای سرورهای Xray-core است. این پنل یک رابط کاربری وب چندزبانه یکپارچه برای استقرار، پیکربندی و نظارت بر مجموعه گسترده‌ای از پروتکل‌های پروکسی و VPN ارائه می‌دهد: از یک VPS تکی تا پیکربندی‌های توزیع‌شده با چند گره (نود).

3X-UI یک فورک توسعه‌یافته از پروژه اصلی X-UI است. در مقایسه با آن، پشتیبانی از پروتکل‌های بیشتر، پایداری بالاتر، آمارگیری ترافیک به‌ازای هر کلاینت و قابلیت‌های کاربردی متعددی اضافه شده است.

قابلیت‌های اصلی:

- **inbound با پروتکل‌های مختلف** — VLESS، VMess، Trojan، Shadowsocks، WireGuard، Hysteria2، HTTP، SOCKS (Mixed)، TUN، Dokodemo-door / Tunnel و **MTProto** (پروکسی تلگرام، افزوده‌شده در نسخه 3.3.0).
- **انتقال‌ها و رمزنگاری مدرن** — HTTPUpgrade، gRPC، WebSocket، mKCP، TCP (Raw) و XHTTP، با امنیت XTLS، TLS و REALITY.
- **Fallback** — سرویس‌دهی چند پروتکل روی یک پورت (مثلاً VLESS و Trojan روی 443) از طریق fallback در Xray.
- **مدیریت به‌ازای هر کلاینت** — سهمیه ترافیک، تاریخ انقضا، محدودیت IP، نمایش وضعیت «آنلاین»، لینک دعوت با یک کلیک، QR کد و سابسکریپشن.
- **آمار ترافیک** — به‌ازای هر inbound، کلاینت و outbound، با امکان ریست کردن.
- **پشتیبانی از چند گره (نود)** — مدیریت و مقیاس‌پذیری روی چند سرور از یک پنل واحد.
- **outbound و مسیریابی** — WARP، NordVPN، قوانین مسیریابی سفارشی، بالانسرهای بار، زنجیره پروکسی.
- **سرور سابسکریپشن داخلی** با چند فرمت خروجی.
- **ربات تلگرام** برای نظارت و مدیریت از راه دور.
- **REST API** با مستندات Swagger داخلی.
- **ذخیره‌سازی انعطاف‌پذیر** — SQLite (پیش‌فرض) یا PostgreSQL.
- **13 زبان رابط کاربری**، تم تاریک و روشن.
- **یکپارچه‌سازی با Fail2ban** برای اعمال محدودیت IP به‌ازای هر کلاینت.

> نکته مهم: این پروژه تنها برای استفاده شخصی طراحی شده است. استفاده از آن برای اهداف غیرقانونی یا در محیط‌های تولیدی توصیه نمی‌شود.

1.2. سیستم‌عامل‌ها و معماری‌های پشتیبانی‌شده

سیستم‌عامل‌ها

اسکریپت نصب، توزیع لینوکس را از فیلد `ID` در `/etc/os-release` (یا `/usr/lib/os-release`) تشخیص می‌دهد. توزیع‌های پشتیبانی‌شده به‌صورت رسمی:

Ubuntu، Debian، Armbian، Fedora، CentOS، RHEL، AlmaLinux، Rocky Linux، Oracle Linux، Amazon Linux، Alpine، openSUSE (Tumbleweed / Leap)، Parch، Manjaro، Arch، Virtuozzo و Windows.

در سیستم‌های مبتنی بر Alpine از سرویس OpenRC استفاده می‌شود (`rc-service` / `rc-update`)، و در بقیه از systemd. برای CentOS 7 بسته‌ها از طریق `yum` و برای نسخه‌های جدیدتر از طریق `dnf` نصب می‌شوند. اگر توزیع شناسایی نشود، اسکریپت به‌صورت پیش‌فرض سعی می‌کند از مدیر بسته `apt-get` استفاده کند.

معماری‌های پردازنده

معماری از خروجی `uname -m` تشخیص داده می‌شود و به یکی از مقادیر پشتیبانی‌شده زیر نگاشت می‌گردد:

مقدار `uname -m`	معماری 3X-UI
`amd64` , `x64` , `x86_64`	`amd64`
`x86` , `i*86`	`386`
`armv8*` , `arm64` , `aarch64`	`arm64`
`armv7*` , `arm`	`armv7`
`armv6*`	`armv6`
`armv5*`	`armv5`
`s390x`	`s390x`

اگر معماری در این فهرست نباشد، اسکریپت پیام «Unsupported CPU architecture!» را نمایش داده و نصب را متوقف می‌کند.

وابستگی‌های پایه

پیش از نصب پنل، اسکریپت به‌صورت خودکار مجموعه‌ای از بسته‌های پایه را نصب می‌کند (نام بسته‌ها بسته به توزیع متفاوت است): `cronie` / `dcron` ، `curl` ، `tar` ، `tzdata` / `timezone` ، `socat` ، `ca-certificates` ، `openssl` / `cron` .

1.3. روش‌های نصب

روش 1. اسکریپت نصب (توصیه‌شده)

نصب با یک دستور به‌عنوان root انجام می‌شود:

```
bash <(curl -Ls https://raw.githubusercontent.com/mhsanaei/3x-ui/master/install.sh)
```

این اسکریپت الزاماً به دسترسی root نیاز دارد: اگر بدون root اجرا شود، پیام «Please run this script with root privilege» نمایش داده شده و با خطا خارج می‌شود.

مراحل کاری نصب‌کننده:

1. سیستم‌عامل و معماری را تشخیص می‌دهد.
2. وابستگی‌های پایه را نصب می‌کند.
3. آرشیو ریلیز `x-ui-linux-<arch>.tar.gz` را دانلود کرده و در مسیر `/usr/local/x-ui` استخراج می‌کند.
4. اسکریپت مدیریتی `x-ui.sh` را دانلود و به‌عنوان دستور `/usr/bin/x-ui` نصب می‌کند.
5. پوشه لاگ `/var/log/x-ui` را ایجاد می‌کند.
6. پیکربندی اولیه را اجرا می‌کند: انتخاب پایگاه داده، تولید اعتبارنامه، انتخاب پورت، پیکربندی اختیاری SSL.
7. سرویس راه‌اندازی خودکار را نصب و فعال می‌کند (یونیت systemd با نام `x-ui.service` یا اسکریپت init مربوط به OpenRC برای Alpine).

انتخاب پایگاه داده هنگام نصب. نصب‌کننده گزینه‌های زیر را پیشنهاد می‌دهد:

- `1) SQLite` (پیش‌فرض، برای تعداد کلاینت کمتر از 500 توصیه می‌شود) — یک فایل واحد در `/etc/x-ui/x-ui.db` ، نیاز به پیکربندی ندارد.
- `2) PostgreSQL` (برای تعداد زیاد کلاینت یا چند نود توصیه می‌شود). می‌توان PostgreSQL را به‌صورت محلی نصب کرد (یک کاربر و پایگاه داده اختصاصی با نام `xui` ایجاد می‌شود) یا DSN یک سرور موجود را وارد کرد. پارامترهای اتصال در فایل محیط سرویس (`/etc/default/x-ui` ، `/etc/conf.d/x-ui` یا `/etc/sysconfig/x-ui` بسته به توزیع) به‌صورت متغیرهای `XUI_DB_TYPE` و `XUI_DB_DSN` ذخیره می‌شوند.

مثال: ذخیره پارامترهای PostgreSQL در فایل محیط سرویس. پس از انتخاب PostgreSQL و وارد کردن DSN، نصب‌کننده خطوطی مشابه زیر را به فایل محیط اضافه می‌کند:

```
XUI_DB_TYPE=postgres
XUI_DB_DSN=postgres://xui:S3cretPass@127.0.0.1:5432/xui?sslmode=disable
```

در اینجا `xui` نام کاربر و پایگاه داده است، `127.0.0.1:5432` آدرس و پورت سرور است، و `sslmode=disable` برای اتصال محلی مناسب است (برای سرور راه دور معمولاً از `require` استفاده می‌شود).

نصب نسخه خاص (قدیمی‌تر). می‌توان یک تگ نسخه مشخص را تعیین کرد — نصب‌کننده ریلیز مربوطه را دانلود می‌کند:

```
bash <(curl -Ls https://raw.githubusercontent.com/mhsanaei/3x-ui/v2.4.0/install.sh)
v2.4.0
```

حداقل نسخه مجاز برای این روش نصب `v2.3.5` است؛ اگر نسخه قدیمی‌تری مشخص شود، پیام «Please use a newer version (at least v2.3.5)» نمایش داده می‌شود.

نصب نسخه dev. علاوه بر تگ نسخه، نصب‌کننده آرگومان `dev-latest` (نام مستعار `dev`) را هم می‌پذیرد — این گزینه آخرین نسخه rolling dev بر اساس آخرین کامیت شاخه `main` را نصب می‌کند:

```
bash <(curl -Ls https://raw.githubusercontent.com/mhsanaei/3x-ui/master/install.sh) dev-
latest
```

نسخه dev یک پیش‌ریلیز per-commit (تگ `dev-latest`) است، نه نسخه پایدار، بنابراین بررسی حداقل نسخه مجاز برای آن انجام نمی‌شود. هنگام اجرا هشداری نمایش داده می‌شود: «Installing the rolling dev build (tag: dev-latest). This is a per-commit pre-release, not a stable version.». بدون آرگومان، نصب‌کننده آخرین ریلیز پایدار را نصب می‌کند. استفاده از نسخه dev تنها برای بررسی رفع اشکالاتی که هنوز منتشر نشده‌اند توجیه دارد؛ در استفاده روزمره، نسخه‌های پایدار را نصب کنید.

روش 2. Docker

اجرا با پایگاه داده SQLite پیش‌فرض:

```
docker compose up -d
```

برای اجرا با سرویس PostgreSQL داخلی، باید خطوط `XUI_DB_*` را در `docker-compose.yml` از حالت کامنت خارج کنید و با پروفایل مربوطه اجرا کنید:

```
docker compose --profile postgres up -d
```

ایمیج شامل Fail2ban است (به‌صورت پیش‌فرض فعال) برای اعمال محدودیت IP کلاینت‌ها. Fail2ban با `iptables` بلاک می‌کند که نیازمند قابلیت `NET_ADMIN` است. در `docker-compose.yml` این قابلیت از قبل از طریق `cap_add` تأمین شده است. در اجرای دستی از طریق `docker run` باید قابلیت‌ها را خودتان اضافه کنید، وگرنه بلاک‌ها تنها در لاگ ثبت می‌شوند اما اعمال نمی‌شوند:

مثال: دستور کامل `docker run`. نمونه مینیمال با فوروارد کردن پورت پنل، قابلیت‌های شبکه و یک ولوم دائمی برای پایگاه داده:

```
docker run -d \
  --name 3x-ui \
  --restart unless-stopped \
  --cap-add=NET_ADMIN --cap-add=NET_RAW \
  -v $PWD/db:/etc/x-ui \
  -v $PWD/cert:/root/cert \
  -p 2053:2053 \
  ghcr.io/mhsanaei/3x-ui:latest
```

ولوم `/etc/x-ui` فایل `x-ui.db` را بین ری‌استارت‌های کانتینر حفظ می‌کند، در غیر این صورت تنظیمات و اکانت‌ها از دست می‌روند.

```
docker run -d --cap-add=NET_ADMIN --cap-add=NET_RAW ... ghcr.io/mhsanaei/3x-ui
```

در Docker، پنل فرآیند اصلی کانتینر است: راه‌اندازی خودکار توسط سیاست ری‌استارت کانتینر مدیریت می‌شود (مثلاً `restart: unless-stopped`)، نه یک سرویس داخل کانتینر.

1.4. اولین اجرا و اعتبارنامه‌های پیش‌فرض

در اولین نصب (وقتی هنوز از اعتبارنامه‌های پیش‌فرض استفاده می‌شود)، نصب‌کننده **مقادیر تصادفی** برای نام کاربری، رمز عبور، مسیر وب و پورت **تولید می‌کند**:

پارامتر	نحوه تعیین هنگام نصب	توضیح
نام کاربری (Username)	رشته تصادفی 10 کاراکتری	به‌صورت خودکار تولید می‌شود
رمز عبور (Password)	رشته تصادفی 10 کاراکتری	به‌صورت خودکار تولید می‌شود
مسیر وب پنل (WebBasePath)	رشته تصادفی 18 کاراکتری	پنل را از شناسایی از طریق URL ریشه محافظت می‌کند
پورت پنل (Port)	به‌صورت پیش‌فرض یک پورت تصادفی در بازه 1024–62000؛ در صورت تمایل می‌توان به‌صورت دستی تعیین کرد	مقدار «کارخانه‌ای» `webPort` برابر `2053` است اما نصب‌کننده آن را بازنویسی می‌کند

در پایان نصب، اسکریپت یک خلاصه نهایی نمایش می‌دهد: نام کاربری، رمز عبور، پورت، مسیر وب، توکن API و لینک آماده ورود (Access URL) به‌شکل:

```
https://<دامنه-یا-IP>:<مسیر-وب>/<پورت>
```

اگر گواهینامه SSL پیکربندی نشده باشد، لینک از طریق `http://` خواهد بود و اسکریپت هشداری درباره نیاز به پیکربندی SSL (گزینه منو 19) نمایش می‌دهد.

> تغییر اجباری اعتبارنامه‌ها. از آنجا که نام کاربری و رمز عبور به‌صورت تصادفی تولید می‌شوند، باید **بلافاصله پس از نصب آن‌ها را ذخیره کنید**. می‌توانید در هر زمان آن‌ها را از طریق گزینه منو «Reset Username & Password» (به زیر مراجعه کنید) یا از رابط وب در تنظیمات پنل تغییر دهید. پس از ریست، اسکریپت یادآوری می‌کند: «Please use the new login username and password to access the X-UI panel. Also remember them!».

پس از نصب، برای باز کردن منوی مدیریت از دستور `x-ui` استفاده کنید (به بخش 1.6 مراجعه کنید).

1.5. محل فایل‌ها

مسیر	کاربرد
`/usr/local/x-ui/`	پوشه نصب پنل (باینری `x-ui` ، اسکریپت `x-ui.sh`)
`/usr/local/x-ui/bin/xray-linux-<arch>`	باینری Xray-core (روی armv5/armv6/armv7 به `xray-linux-arm` تغییر نام می‌یابد)
`/usr/bin/x-ui`	اسکریپت مدیریتی (دستور `x-ui`)
`/etc/x-ui/x-ui.db`	فایل پایگاه داده SQLite (پیش‌فرض)
`/var/log/x-ui/`	پوشه لاگ‌های پنل
`/etc/systemd/system/x-ui.service`	یونیت systemd سرویس (نه برای Alpine)
`/etc/init.d/x-ui`	اسکریپت init مربوط به OpenRC (فقط Alpine)
`/etc/default/x-ui` · `/etc/conf.d/x-ui` · `/etc/sysconfig/x-ui`	فایل متغیرهای محیط سرویس (مسیر بسته به توزیع متفاوت است)؛ `XUI_DB_DSN` / `XUI_DB_TYPE` در اینجا نوشته می‌شوند

پوشه پایگاه داده را می‌توان با متغیر محیط `XUI_DB_FOLDER` بازتعریف کرد (پیش‌فرض `/etc/x-ui`)، و پوشه باینری‌های Xray را با متغیر `XUI_BIN_FOLDER` (پیش‌فرض `bin` نسبت به پوشه پنل). نام فایل پایگاه داده `x-ui.db` است.

مثال: انتقال پایگاه داده به دیسک جداگانه. برای ذخیره `x-ui.db` نه در `/etc/x-ui` بلکه مثلاً روی دیسک مانت‌شده `/data` ، متغیر را در فایل محیط سرویس تعریف کنید و پنل را ری‌استارت کنید:

```
echo 'XUI_DB_FOLDER=/data/x-ui' >> /etc/default/x-ui
mkdir -p /data/x-ui
systemctl restart x-ui
```

مسیر کامل پایگاه داده `/data/x-ui/x-ui.db` خواهد شد.

متغیرهای محیط اصلی

متغیر	کاربرد	پیش‌فرض
`XUI_DB_TYPE`	بک‌اند پایگاه داده: `sqlite` یا `postgres`	`sqlite`
`XUI_DB_DSN`	رشته اتصال PostgreSQL (هنگام `XUI_DB_TYPE=postgres`)	–
`XUI_DB_FOLDER`	پوشه فایل پایگاه داده SQLite	`/etc/x-ui`
`XUI_INIT_WEB_BASE_PATH`	مسیر URI اولیه پنل وب (فقط در اولین راه‌اندازی)	`/`

متغیر	کاربرد	پیش‌فرض
`XUI_DB_MAX_OPEN_CONNS`	حداکثر اتصالات باز (پول PostgreSQL)	—
`XUI_DB_MAX_IDLE_CONNS`	حداکثر اتصالات بیکار (پول PostgreSQL)	—
`XUI_ENABLE_FAIL2BAN`	فعال‌سازی اعمال محدودیت IP از طریق Fail2ban	`true`
`XUI_LOG_LEVEL`	سطح لاگ‌گیری (`debug` ، `info` ، `warning` ، `error`)	`info`
`XUI_DEBUG`	حالت دیباگ	`false`

مثال: فعال‌سازی موقت لاگ‌گیری دقیق. برای تشخیص مشکل، سطح لاگ را به `debug` افزایش داده و سرویس را ری‌استارت کنید:

```
echo 'XUI_LOG_LEVEL=debug' >> /etc/default/x-ui
systemctl restart x-ui
x-ui log    # مشاهده لاگ دیباگ
```

پس از عیب‌یابی مقدار را به `info` برگردانید تا لاگ بیش از حد رشد نکند.

مسیر اولیه پنل وب از طریق محیط. متغیر `XUI_INIT_WEB_BASE_PATH` مسیر URI پنل وب (`webBasePath`) را در راه‌اندازی اولیه تنظیم می‌کند. این در هنگام استقرار در Docker یا از طریق systemd مفید است تا مسیر ورود به پنل از همان ابتدا ثابت باشد. مقدار به‌صورت خودکار نرمال‌سازی می‌شود — اسلش‌های ابتدا و انتها در صورت نیاز اضافه می‌شوند، و مقدار خالی یا شامل فضای خالی نادیده گرفته می‌شود (در این صورت مسیر پیش‌فرض `/` اعمال می‌شود). این متغیر **تنها بر راه‌اندازی اولیه تأثیر می‌گذارد:** اگر تنظیمات از قبل ایجاد شده باشند، مسیر از طریق رابط وب یا گزینه منو «Reset Web Base Path» تغییر می‌کند.

1.6. دستور مدیریتی `x-ui` (منوی اسکریپت)

پس از نصب، دستور `x-ui` (اجرا به‌عنوان root) منوی تعاملی «3X-UI Panel Management Script» را باز می‌کند. انتخاب گزینه با وارد کردن شماره آن (بازه 0–27) انجام می‌شود. بسیاری از گزینه‌ها به‌صورت زیردستور نیز برای استفاده در اسکریپت‌ها در دسترس هستند (به بخش 1.7 مراجعه کنید).

منو به بلوک‌های موضوعی تقسیم شده است.

نصب و به‌روزرسانی

- **1. Install** — نصب پنل (اجرای `install.sh`). پیش از نصب بررسی می‌شود که پنل هنوز نصب نشده باشد.
- **2. Update** — به‌روزرسانی تمام اجزای x-ui به آخرین نسخه. داده‌ها از دست نمی‌روند؛ پس از به‌روزرسانی پنل به‌صورت خودکار ری‌استارت می‌شود. نیاز به تأیید دارد.
- **3. Update Menu** — به‌روزرسانی تنها اسکریپت مدیریتی (`x-ui.sh` / دستور `x-ui`) به نسخه جدید بدون نصب مجدد پنل.
- **4. Legacy Version** — نصب نسخه مشخص (قدیمی‌تر) پنل. اسکریپت شماره نسخه (مثلاً `2.4.0`) را پرسیده و ریلیز مربوطه را دانلود می‌کند.
- **5. Uninstall** — حذف کامل پنل **به همراه Xray**. سرویس متوقف و غیرفعال می‌شود، پوشه‌های `/etc/x-ui/` و `/usr/local/x-ui/`، فایل محیط سرویس و اسکریپت مدیریتی حذف می‌شوند. نیاز به تأیید دارد (پیش‌فرض «خیر»).

اعتبارنامه‌ها و تنظیمات

- 6. **Reset Username & Password** — ریست نام کاربری و رمز عبور پنل. می‌توان مقادیر دلخواه وارد کرد یا آن‌ها را خالی گذاشت تا به‌صورت تصادفی تولید شوند (نام تصادفی 10 کاراکتر، رمز تصادفی 18 کاراکتر). همچنین پیشنهاد می‌شود احراز هویت دو مرحله‌ای (2FA) در صورت فعال بودن غیرفعال شود. پس از ریست، پنل ری‌استارت می‌شود.
- 7. **Reset Web Base Path** — ریست مسیر وب پنل: یک مسیر تصادفی جدید (18 کاراکتر) تولید شده و پنل ری‌استارت می‌شود. در صورتی که مسیر قبلی لو رفته یا فراموش شده باشد استفاده می‌شود.
- 8. **Reset Settings** — ریست تمام تنظیمات پنل به مقادیر پیش‌فرض. **اعتبارنامه‌ها (نام کاربری و رمز عبور) و داده‌های اکانت‌ها از دست نمی‌روند.** نیاز به تأیید دارد؛ پس از ریست پنل ری‌استارت می‌شود.
- 9. **Change Port** — تغییر پورت پنل وب. شماره پورت (1–65535) پرسیده می‌شود؛ پس از تنظیم برای اعمال شدن پورت، ری‌استارت لازم است.
- 10. **View Current Settings** — مشاهده تنظیمات جاری (`x-ui setting -show`). از جمله بک‌اند پایگاه داده (SQLite یا PostgreSQL با رمز ماسک‌شده در DSN) و لینک دسترسی آماده (Access URL) را نمایش می‌دهد. اگر SSL پیکربندی نشده باشد، پیشنهاد صدور گواهینامه Let's Encrypt برای آدرس IP داده می‌شود.

مدیریت سرویس

- 11. **Start** — راه‌اندازی سرویس پنل. اگر پنل در حال اجرا باشد، پیامی مبنی بر عدم نیاز به راه‌اندازی مجدد نمایش داده می‌شود.
- 12. **Stop** — توقف سرویس پنل.
- 13. **Restart** — ری‌استارت سرویس پنل.
- 14. **Restart Xray** — ری‌استارت تنها هسته Xray-core بدون ری‌استارت خود پنل (از طریق `systemctl reload x-ui` ، در Docker با سیگنال `USR1` به فرآیند پنل).
- 15. **Check Status** — بررسی وضعیت سرویس (`systemctl status x-ui` یا `rc-service x-ui status`).
- 16. **Logs Management** — مدیریت لاگ‌ها: مشاهده لاگ دیباگ (Debug Log، از طریق `journalctl`) و، به جز Alpine، پاک کردن تمام لاگ‌ها (Clear All logs).

راه‌اندازی خودکار

- 17. **Enable Autostart** — فعال‌سازی راه‌اندازی خودکار پنل هنگام بوت سیستم‌عامل (`systemctl enable x-ui` یا `rc-update add`).
- 18. **Disable Autostart** — غیرفعال‌سازی راه‌اندازی خودکار هنگام بوت سیستم‌عامل.

در Docker، راه‌اندازی خودکار توسط سیاست ری‌استارت کانتینر مدیریت می‌شود، بنابراین این گزینه‌ها تنها راهنمای مربوطه را نمایش می‌دهند.

امنیت و شبکه

- 19. **SSL Certificate Management** — مدیریت گواهینامه‌های SSL از طریق acme.sh: صدور گواهینامه برای دامنه، ابطال، تمدید اجباری، مشاهده دامنه‌های موجود، تعیین مسیر گواهینامه برای پنل، و همچنین صدور گواهینامه کوتاه‌مدت (~6 روزه، با تمدید خودکار) برای آدرس IP.
- 20. **Cloudflare SSL Certificate** — صدور گواهینامه SSL از طریق اعتبارسنجی DNS کلودفلر.
- 21. **IP Limit Management** — مدیریت محدودیت تعداد IP به‌ازای هر کلاینت (بر اساس Fail2ban): مشاهده و رفع بلاک‌ها و غیره.
- 22. **Firewall Management** — مدیریت فایروال (باز/بستن پورت‌ها و مشاهده قوانین).
- 23. **SSH Port Forwarding Management** — پیکربندی فورواردینگ پورت SSH برای دسترسی به پنل از دستگاه محلی از طریق تونل SSH.

عملکرد و نگهداری

- **24. Enable BBR** — فعال/غیرفعال‌سازی الگوریتم کنترل ازدحام TCP BBR (زیرمنو با گزینه‌های Enable BBR / Disable BBR).
- **25. Update Geo Files** — به‌روزرسانی پایگاه داده‌های جغرافیایی (فایل‌های `.dat`) با انتخاب منبع: Loyalsoldier (`geoip.dat`، `geosite.dat`)، chocolate4u (`geoip_IR.dat`، `geosite_IR.dat`)، runetfreedom (`geoip_RU.dat`، `geosite_RU.dat`) یا All (همه با هم). پس از به‌روزرسانی پنل ری‌استارت می‌شود.
- **26. Speedtest by Ookla** — اجرای تست سرعت شبکه از طریق Speedtest by Ookla.
- **27. PostgreSQL Management** — مدیریت نمونه PostgreSQL داخلی/متصل (فعال‌سازی و عملیات مرتبط).
- **0. Exit Script** — خروج از منو.

1.7. زیردستورهای `x-ui` (بدون منوی تعاملی)

برای استفاده در اسکریپت‌ها، دستور `x-ui` از زیردستورهای مستقیم پشتیبانی می‌کند (اجرای `x-ui` بدون آرگومان منو را باز می‌کند):

دستور	عملکرد
`x-ui`	باز کردن منوی مدیریت
`x-ui start`	راه‌اندازی پنل
`x-ui stop`	توقف پنل
`x-ui restart`	ری‌استارت پنل
`x-ui restart-xray`	ری‌استارت Xray
`x-ui status`	وضعیت جاری سرویس
`x-ui settings`	تنظیمات جاری
`x-ui enable`	فعال‌سازی راه‌اندازی خودکار هنگام بوت سیستم‌عامل
`x-ui disable`	غیرفعال‌سازی راه‌اندازی خودکار
`x-ui log`	مشاهده لاگ‌ها
`x-ui banlog`	مشاهده لاگ‌های بلاک Fail2ban
`x-ui update`	به‌روزرسانی پنل
`x-ui update-all-geofiles`	به‌روزرسانی تمام فایل‌های geo
`x-ui migrateDB [file]`	تبدیل `.db` ⇄ `.dump` (SQLite)
`x-ui legacy`	نصب نسخه قدیمی
`x-ui install`	نصب پنل
`x-ui uninstall`	حذف پنل

1.8. مهاجرت از SQLite به PostgreSQL

یک نصب موجود روی SQLite را می‌توان به PostgreSQL منتقل کرد:

```
x-ui migrate-db --dsn "postgres://xui:password@127.0.0.1:5432/xui?sslmode=disable"
# تنظیم کنید و ری‌استارت کنید /etc/default/x-ui را در XUI_DB_DSN و XUI_DB_TYPE سپس:
systemctl restart x-ui
```

فایل SQLite اصلی دست‌نخورده باقی می‌ماند — آن را تنها پس از تأیید کارکرد صحیح بک‌اند جدید به‌صورت دستی حذف کنید.

مثال: تأیید تغییر به PostgreSQL. پس از مهاجرت، با دستور مشاهده تنظیمات مطمئن شوید که پنل واقعاً روی بک‌اند جدید کار می‌کند — در خروجی باید PostgreSQL نمایش داده شود (رمز در DSN ماسک می‌شود):

```
x-ui settings | grep -i -E 'db|dsn'
```

اگر پنل باز شد و اکانت‌ها سرجایشان بودند، می‌توانید `x-ui.db` اصلی را حذف کنید.

---

2. ورود به پنل و امنیت دسترسی

این بخش همه چیزهای مرتبط با احراز هویت مدیر پنل 3X-UI را شرح می‌دهد: فرم ورود، احراز هویت دومرحله‌ای (TOTP)، محافظت در برابر حدس زدن رمز عبور، تغییر اطلاعات حساب، تغییر مسیر مخفی و پورت پنل، طول عمر نشست، و همچنین همگام‌سازی/احراز هویت از طریق LDAP.

2.1. فرم ورود

صفحه ورود از ریشه مسیر مخفی پنل (`webBasePath`) ارائه می‌شود. اگر کاربر از پیش احراز هویت شده باشد، به‌طور خودکار به `/…panel/` هدایت می‌شود. در صفحه یک کلید تغییر تم، انتخاب زبان رابط کاربری و خودِ فرم وجود دارد.

فیلدهای فرم:

فیلد	راهنما/عنوان (RU)	اجباری	توضیحات
نام کاربری	«Имя пользователя»	بله	لاگین مدیر. مقدار خالی همان در سمت کلاینت رد می‌شود و در سمت سرور — با پیام «Введите имя пользователя».
رمز عبور	«Пароль»	بله	رمز عبور مدیر. مقدار خالی با پیام «Введите пароль» رد می‌شود.
کد 2FA	«Код 2FA»	فقط هنگام فعال بودن 2FA	این فیلد **فقط** زمانی ظاهر می‌شود که احراز هویت دومرحله‌ای روی پنل فعال باشد. کد 6 رقمی از برنامه احراز هویت‌گر.

دکمه **«Войти»** فرم را به `POST /login` ارسال می‌کند.

رفتار و پیام‌ها:

- در صورت ورود موفق، پیام «Вход выполнен успешно» نمایش داده می‌شود و انتقال به `/…panel/` رخ می‌دهد.
- در هر خطای اطلاعات حساب یا کد 2FA نادرست، سرور یک پیام **یکپارچه** بازمی‌گرداند: «Неверные данные учетной записи.» (انگلیسی: *Invalid username or password or two-factor code*.). این کار عمدی است — پنل اشاره نمی‌کند که دقیقاً چه چیزی نادرست است (لاگین، رمز عبور یا کد) تا حدس زدن را آسان نکند.
- فیلد «Код 2FA» را پنل بر اساس درخواست `POST /getTwoFactorEnable` نمایش می‌دهد یا مخفی می‌کند؛ این درخواست وضعیت فعلی 2FA را پیش از احراز هویت بازمی‌گرداند.
- اگر نشست سمت سرور منقضی شده باشد، در درخواست بعدی پیام «Сессия истекла. Войдите в систему снова» نمایش داده می‌شود و کاربر به صفحه ورود هدایت می‌شود.

> یادداشت درباره CSRF: پیش از ارسال فرم، کلاینت یک توکن CSRF دریافت می‌کند (`GET /csrf-token`)؛ درخواست‌های `/login` و `/logout` با بررسی CSRF محافظت می‌شوند.

مثال: ورود از طریق API. زمانی که 2FA غیرفعال است، لاگین و رمز عبور کافی است؛ هنگام فعال بودن 2FA فیلد `twoFactorCode` اضافه می‌شود:

```
# Без 2FA
curl -i -X POST https://panel.example.com:2053/мой-секрет/login \
  -H 'Content-Type: application/x-www-form-urlencoded' \
  --data 'username=admin&password=ВашПароль'

# С включённой 2FA — добавляется 6-значный код
curl -i -X POST https://panel.example.com:2053/мой-секрет/login \
  -H 'Content-Type: application/x-www-form-urlencoded' \
  --data 'username=admin&password=ВашПароль&twoFactorCode=123456'
```

در صورت موفقیت، سرور یک `Set-Cookie` با کوکی نشست بازمی‌گرداند — همان را باید در درخواست‌های بعدی به `/panel/api/…` ارسال کرد.

2.2. احراز هویت دومرحله‌ای (2FA / TOTP)

2FA در 3X-UI بر اساس استاندارد **TOTP** پیاده‌سازی شده و با هر برنامه احراز هویت‌گری (Google Authenticator, Aegis, FreeOTP و غیره) سازگار است. پارامترها به‌صورت ثابت تعیین شده‌اند: الگوریتم **SHA1**، **6** رقم، دوره **30** ثانیه، صادرکننده (issuer) `3x-ui` ، برچسب `Administrator` .

مثال: otpauth-URI که QR-کد را کدگذاری می‌کند. اگر برنامه احراز هویت‌گر با دوربین اسکن نمی‌کند، توکن را می‌توان به‌صورت دستی با چنین پیوندی اضافه کرد (به‌جای `JBSWY3DPEHPK3PXP` سکرت Base32 خود را قرار دهید):

```
otpauth://totp/3x-ui:Administrator?secret=JBSWY3DPEHPK3PXP&issuer=3x-ui&algorithm=SHA1&digits=6&period=30
```

پارامترهای `algorithm=SHA1` ، `digits=6` ، `period=30` با مقادیر ثابت پنل مطابقت دارند — نیازی به تغییر آن‌ها نیست.

تنظیمات در بخش **Учетная запись** → **Настройки**، زبانه **«Двухфакторная аутентификация»** قرار دارد.

عنصر	متن (RU)	توضیحات
کلید تغییر وضعیت	«Включить 2FA»	احراز هویت دومرحله‌ای را فعال/غیرفعال می‌کند.
توضیح	«Добавляет дополнительный уровень аутентификации для повышения безопасности.»	راهنمای زیر کلید.

چگونه 2FA را فعال کنیم

هنگام فعال کردن کلید، پنل **به‌صورت محلی یک سکرت جدید تولید می‌کند** — یک رشته تصادفی با کدگذاری Base32 (الفبای `A–Z` و `2–7`). پنجره «Включить двухфакторную аутентификацию» با دستورالعمل گام‌به‌گام باز می‌شود:

1. **«Отсканируйте этот QR-код в приложении для аутентификации или скопируйте токен рядом с QR-кодом и вставьте его в приложение».** زیر QR-کد، خودِ سکرت به‌صورت متنی نمایش داده می‌شود — با کلیک روی QR-کد، سکرت در حافظه موقت کپی می‌شود (پیام «Скопировано» ظاهر می‌شود).
2. **«Введите код из приложения»** — باید کد 6 رقمی‌ای را که برنامه تولید کرده وارد کنید. کد **در سمت مرورگر** بررسی می‌شود: پنل خودش TOTP فعلی را بر اساس سکرتی که تازه تولید شده محاسبه می‌کند و با مقدار وارد شده مقایسه می‌کند. اگر کد نادرست باشد — «Неверный код»؛ این فیلد فقط دقیقاً 6 رقم را می‌پذیرد.

تنها پس از تأیید موفق، سکرت و پرچم فعال‌سازی ذخیره می‌شوند. هنگام ذخیره، پیام «Двухфакторная аутентификация была успешно установлена» نمایش داده می‌شود.

مهم: تغییرات در بخش تنظیمات با دکمه عمومی **«Сохранить»** اعمال می‌شوند، که پس از آن معمولاً نیاز به راه‌اندازی مجدد پنل است («Сохраните изменения и перезапустите панель для их применения»). هنگام نخستین فعال‌سازی 2FA، سرور به‌علاوه **همه نشست‌های فعال را باطل می‌کند** («login epoch» را افزایش می‌دهد)، بنابراین پس از اعمال تنظیمات، ورود مجدد لازم خواهد بود — این بار با کد 2FA.

چگونه 2FA را غیرفعال کنیم

فشردن دوباره کلید، پنجره «Отключить двухфакторную аутентификацию» را با راهنمای «Введите код из приложения, чтобы отключить двухфакторную аутентификацию.» باز می‌کند. پس از وارد کردن کد درست (از 3.4.2 این کد **روی سرور** بررسی می‌شود)، پرچم و سکرت پاک می‌شوند و پیام «Двухфакторная аутентификация была успешно удалена» نمایش داده می‌شود.

بررسی کد هنگام ورود

هنگام ورود، سرور سکرت ذخیره‌شده را برمی‌دارد و TOTP فعلی را با کد 2FA ارسال‌شده مقایسه می‌کند. عدم تطابق به‌عنوان ورود ناموفق تلقی می‌شود، اما به کاربر همان پیام یکپارچه «Неверные данные учетной записи.» نمایش داده می‌شود.

بازیابی دسترسی (recovery)

سازوکار جداگانه‌ای برای «کدهای بازیابی» در 3X-UI **وجود ندارد**. اگر دسترسی به برنامه احراز هویت‌گر از دست برود، بازگرداندن ورود از طریق رابط پنل امکان‌پذیر نیست. تنها راه — غیرفعال کردن 2FA به‌طور مستقیم در پایگاه داده روی سرور است: تنظیم کلید `twoFactorEnable` به `false` (و در صورت نیاز پاک کردن `twoFactorToken`) در جدول تنظیمات، و سپس راه‌اندازی مجدد پنل. به همین دلیل توصیه می‌شود سکرت (توکن Base32) را هنگام فعال‌سازی 2FA در جایی مطمئن نگه دارید.

مثال: غیرفعال‌سازی اضطراری 2FA روی سرور. پس از دسترسی به سرور از طریق SSH، پنل را متوقف کنید، کلیدها را در جدول تنظیمات بازنشانی کنید و پنل را دوباره اجرا کنید:

```
x-ui stop
sqlite3 /etc/x-ui/x-ui.db "UPDATE settings SET value='false' WHERE
key='twoFactorEnable';"
sqlite3 /etc/x-ui/x-ui.db "UPDATE settings SET value='' WHERE key='twoFactorToken';"
x-ui start
```

پس از این، ورود فقط با لاگین و رمز عبور انجام می‌شود و در صورت تمایل می‌توان 2FA را دوباره پیکربندی کرد.

> ارتباط با تغییر اطلاعات حساب: هنگام تغییر لاگین/رمز عبور (نگاه کنید به 2.4)، 2FA **به‌صورت خودکار غیرفعال می‌شود** تا سکرت قدیمی دسترسی تحت حساب جدید را مسدود نکند.

2.3. محدودسازی تلاش‌های ورود (login limiter / محافظت در برابر حدس زدن)

پنل یک محدودکننده داخلی برای ورودهای ناموفق دارد (مشابه fail2ban در سطح برنامه). پارامترها در کد تعیین شده‌اند و از طریق رابط کاربری **قابل تنظیم نیستند**:

پارامتر	مقدار	کاربرد
حداکثر تعداد ناموفق	**5**	چند تلاش ناموفق در محدوده پنجره مجاز است.
پنجره شمارش	**5 دقیقه**	پنجره لغزان که در آن ناموفق‌ها انباشته می‌شوند (موارد قدیمی‌تر کنار گذاشته می‌شوند).
مسدودسازی (cooldown)	**15 دقیقه**	کلید پس از عبور از آستانه برای چه مدتی مسدود می‌شود.

نحوه کار:

- کلید مسدودسازی از **ترکیب «IP + لاگین»** ساخته می‌شود (لاگین به حروف کوچک تبدیل و فاصله‌ها حذف می‌شوند). یعنی مسدودسازی روی یک جفت مشخص «آدرس + نام کاربری» اعمال می‌شود، نه روی کل پنل.
- در هر تلاش ناموفق (لاگین/رمز عبور نادرست یا کد 2FA نادرست) شمارنده افزایش می‌یابد. با رسیدن به **5** ناموفق در **5 دقیقه**، کلید برای **15 دقیقه** مسدود می‌شود. در طول مسدودسازی، هر تلاش این جفت بلافاصله با همان پیام «Неверные данные учетной записи.» رد می‌شود، حتی اگر داده‌ها درست باشند.
- **ورود موفق بلافاصله** شمارنده را بازنشانی و مسدودسازی این جفت را برمی‌دارد.
- آدرس IP کلاینت با درنظرگرفتن پراکسی‌های مورد اعتماد (نگاه کنید به `trustedProxyCIDRs`) تعیین می‌شود: هدرهای `X-Real-IP` و `X-Forwarded-For` فقط در صورتی پذیرفته می‌شوند که درخواست از آدرس مورد اعتماد آمده باشد. در غیر این صورت از آدرس واقعی اتصال استفاده می‌شود، و در صورت ناممکن بودن استخراج آن — رشته `unknown` .

همه تلاش‌ها در لاگ ثبت می‌شوند. برای موارد ناموفق، یک هشدار در لاگ سرور با نام کاربری، IP، علت و — در صورت مسدودسازی — زمان `blocked_until` نوشته می‌شود. اگر اطلاع‌رسانی ورود از طریق ربات تلگرام فعال باشد (`tgNotifyLogin` — «Уведомление о входе»)، مدیر به‌علاوه نام کاربری، IP و زمان تلاش‌های موفق، ناموفق و مسدودشده را دریافت می‌کند.

مثال: اطلاع‌رسانی ورود در تلگرام. هنگام فعال بودن `tgNotifyLogin` ، پس از هر تلاش مدیر پیامی تقریباً به این شکل دریافت می‌کند:

```
Уведомление о входе
Пользователь: admin
IP: 203.0.113.45
Время: 2026-06-10 14:32:07
Статус: успешно
```

برای جفت مسدودشده «IP + لاگین» در وضعیت قید می‌شود که تلاش توسط محدودکننده رد شده است.

2.4. تغییر لاگین و رمز عبور مدیر

بخش **Учетная запись** → **Настройки**، زبانه **«Учетные данные администратора»**. فیلدها:

فیلد	متن (RU)	توضیحات
لاگین فعلی	«Текущий логин»	نام کاربری جاری باید با لاگین فعلی مطابقت داشته باشد، در غیر این صورت تغییر رد می‌شود.
رمز عبور فعلی	«Текущий пароль»	رمز عبور جاری برای تأیید هویت.
لاگین جدید	«Новый логин»	نام کاربری جدید. نمی‌تواند خالی باشد.
رمز عبور جدید	«Новый пароль»	رمز عبور جدید. نمی‌تواند خالی باشد.

تغییر با دکمه **«Подтвердить»** اعمال و به `POST /panel/setting/updateUser` ارسال می‌شود.

منطق و پیام‌های سرور:

- اگر «Текущий логин» با مقدار واقعی مطابقت نداشته باشد یا «Текущий пароль» نادرست باشد — «Произошла ошибка при изменении учетных данных администратора.» با توضیح «Неверное имя пользователя или пароль».
- اگر لاگین جدید یا رمز عبور جدید خالی باشد — توضیح «Новое имя пользователя и новый пароль должны быть заполнены».
- در صورت موفقیت — «Вы успешно изменили учетные данные администратора.». رمز عبور به‌صورت هش bcrypt ذخیره می‌شود.

تأیید با کد 2FA (از 3.4.2). اگر روی پنل احراز هویت دومرحله‌ای فعال باشد، تغییر لاگین/رمز عبور علاوه بر این نیازمند **وارد کردن کد فعلی** از برنامه احرازکننده است. پنجره **«تغییر اعتبارنامه‌ها»** با راهنمای «کد را از برنامه وارد کنید تا اعتبارنامه‌های مدیر تغییر کند.» باز می‌شود؛ کد (`twoFactorCode`) روی سرور بررسی می‌شود، و در صورت نادرست یا خالی بودن کد، تغییر اعمال نمی‌شود. همین تأیید اکنون هنگام غیرفعال کردن 2FA نیز لازم است (به 2.2 مراجعه کنید).

مثال: تغییر اطلاعات حساب از طریق API. درخواست به یک کوکی نشست معتبر (که هنگام ورود به‌دست می‌آید) و تأیید لاگین/رمز عبور فعلی نیاز دارد:

```
curl -X POST https://panel.example.com:2053/мой-секрет/panel/setting/updateUser \
  -b 'session=ВАША_СЕССИОННАЯ_COOKIE' \
  -H 'Content-Type: application/x-www-form-urlencoded' \
  --data 'oldUsername=admin&oldPassword=СтарыйПароль&newUsername=root&newPassword=НовыйС
ложныйПароль'
```

پس از موفقیت، نشست جاری باطل می‌شود — لازم است با داده‌های جدید دوباره وارد شوید.

اثرات مهم تغییر اطلاعات حساب:

- **همه نشست‌های موجود باطل می‌شوند** (شمارنده `login_epoch` کاربر افزایش می‌یابد)، بنابراین پس از تغییر، پنل به‌طور خودکار خروج را انجام می‌دهد و به صفحه ورود هدایت می‌کند — باید دوباره وارد شد.
- اگر در لحظه تغییر **2FA فعال بوده باشد، به‌صورت خودکار غیرفعال می‌شود** (پرچم و سکرت بازنشانی می‌شوند). احراز هویت دومرحله‌ای را پس از تغییر لاگین/رمز عبور باید دوباره پیکربندی کرد.

اگر 2FA فعال باشد، پیش از ارسال فرم پنجره «Изменить учетные данные» با راهنمای «Введите код из приложения, чтобы изменить учетные данные администратора.» باز می‌شود — تغییر اطلاعات حساب فقط با تأیید کد 2FA فعلی ممکن است.

2.5. مسیر مخفی (مسیر URI / webBasePath) و پورت پنل

این پارامترها در بخش **Панель** → **Настройки** قرار دارند و مستقیماً بر «پنهان‌بودن» و دسترس‌پذیری پنل تأثیر می‌گذارند. پس از ذخیره و **راه‌اندازی مجدد پنل** اعمال می‌شوند.

فیلد	متن (RU)	مقدار پیش‌فرض	توضیحات
پورت پنل	«Порт панели» (`panelPort`)، راهنما «Порт, на котором работает панель»	**2053**	پورت TCP رابط وب.
مسیر URI	«URI-путь» (`panelUrlPath`)، راهنما «Должен начинаться с '/' и заканчиваться '/'»	**/**	مسیر پایه مخفی (`webBasePath`). پنل فقط از طریق آن در دسترس است (مثلاً `/мой-секрет/`).

فیلد	متن (RU)	مقدار پیش‌فرض	توضیحات
آدرس IP برای مدیریت پنل	«IP-адрес для управления панелью» (`panelListeningIP`)، راهنما «Оставьте пустым для подключения с любого IP»	خالی	آدرسی که پنل روی آن گوش می‌دهد. خالی = همه رابطها.
دامنه پنل	«Домен панели» (`panelListeningDomain`)، راهنما «Оставьте пустым для подключения с любых доменов и IP».	خالی	محدودسازی دسترسی بر اساس دامنه (Host).
مسیر کلید عمومی گواهی پنل	`publicKeyPath`، راهنما «Введите полный путь, начинающийся с '/'»	خالی	گواهی TLS برای دسترسی HTTPS به پنل.
مسیر کلید خصوصی گواهی پنل	`privateKeyPath`، همان راهنما	خالی	کلید خصوصی TLS.

رفتار مسیر پایه (`webBasePath`):

- مقدار به‌طور خودکار نرمال‌سازی می‌شود: اگر با `/` شروع نشود، این نویسه به ابتدا افزوده می‌شود؛ اگر به `/` ختم نشود، به انتها افزوده می‌شود. یعنی عملاً مسیر همیشه به شکل `/.../` است.
- مسیر پایه به خودِ پنل، به دارایی‌ها (ассеты) و به کوکی نشست اعمال می‌شود (کوکی فقط برای این مسیر صادر می‌شود).

> توصیه‌های امنیتی (بخش «Предупреждения безопасности»): پنل خودش هشدارهایی نمایش می‌دهد اگر پیکربندی «بیش از حد عمومی» باشد:
>
> - «Панель работает по обычному HTTP — настройте TLS для продакшна».
> - «Стандартный порт 2053 широко известен — измените его на случайный».
> - «Базовый путь по умолчанию "/" широко известен — измените его на случайный».
>
> به‌عبارت دیگر، برای یک سرور عملیاتی باید **پورت غیراستاندارد، مسیر URI غیرپیش‌پاافتاده** و **گواهی TLS** تعیین کرد.

مثال: پیکربندی «پنهان» پنل برای محیط عملیاتی. در بخش **Панель** → **Настройки** تقریباً چنین مقادیری را تعیین کنید:

فیلد	مقدار
پورت پنل	`34571` (تصادفی، به‌جای 2053)
مسیر URI	`/aXf9Qm2/` (غیرپیش‌پاافتاده، با `/` شروع و ختم می‌شود)
مسیر کلید عمومی گواهی پنل	`/etc/letsencrypt/live/panel.example.com/fullchain.pem`
مسیر کلید خصوصی گواهی پنل	`/etc/letsencrypt/live/panel.example.com/privkey.pem`

پس از ذخیره و راه‌اندازی مجدد، پنل فقط از طریق `https://panel.example.com:34571/aXf9Qm2/` در دسترس خواهد بود و هشدارهای امنیتی ناپدید می‌شوند.

2.6. طول عمر نشست (تایم‌اوت)

فیلد **«Продолжительность сессии»** (`sessionMaxAge`) در میان تنظیمات پنل/بازه‌ها قرار دارد.

فیلد	متن (RU)	مقدار پیش‌فرض	واحد	توضیحات
طول عمر نشست	«Продолжительность сессии», راهنما «Продолжительность сессии в системе (значение: минута)»	**360**	دقیقه	طول عمر کوکی نشست مدیر.

رفتار:

- مقدار بر حسب **دقیقه** تعیین می‌شود (به‌طور پیش‌فرض 360 دقیقه = 6 ساعت) و هنگام تنظیم کوکی به ثانیه تبدیل می‌شود.
- اگر مقدار **بزرگ‌تر از 0** باشد، برای کوکی نشست `MaxAge` متناظر تنظیم می‌شود. پس از پایان این مدت کوکی از اعتبار می‌افتد و در درخواست بعدی کاربر پیام «Сессия истекла. Войдите в систему снова» را دریافت می‌کند.
- نشست همچنین هنگام تغییر اطلاعات حساب یا نخستین فعال‌سازی 2FA پیش از موعد بی‌اعتبار می‌شود (به‌واسطه سازوکار `login_epoch`، نگاه کنید به 2.4 و 2.2) و هنگام خروج صریح (`POST /logout`).
- کوکی نشست با `HttpOnly` و سیاست `SameSite=Lax` علامت‌گذاری می‌شود؛ پرچم `Secure` هنگام دسترسی مستقیم HTTPS به پنل تنظیم می‌شود.

علاوه بر خودِ تایم‌اوت، یک اطلاع‌رسانی مرتبط وجود دارد: **«Задержка уведомления об истечении сессии»** (`expireTimeDiff`، راهنما «Получение уведомления об истечении срока действия сессии до достижения порогового значения (значение: день)»، به‌طور پیش‌فرض `0`) — امکان دریافت هشدار از پیش را فراهم می‌کند.

2.7. LDAP (همگام‌سازی و احراز هویت)

بخش LDAP دو قابلیت می‌دهد: (1) احراز هویت ورود مدیر از طریق LDAP، اگر رمز عبور محلی مطابقت نکند، و (2) همگام‌سازی دوره‌ای وضعیت کلاینت‌ها (پرچم فعال/غیرفعال VLESS) از دایرکتوری.

نحوه استفاده هنگام ورود: سرور ابتدا هش bcrypt محلی رمز عبور را بررسی می‌کند. اگر **مطابقت نکرد** و LDAP فعال باشد، پنل تلاش می‌کند کاربر را در دایرکتوری احراز هویت کند: در صورت تعیین `Bind DN`، یک bind سرویسی انجام می‌شود، سپس بر اساس فیلتر و صفت، رکورد کاربر جستجو می‌شود و تلاش می‌شود تحت DN یافت‌شده با رمز عبور وارد شده bind انجام شود. موفقیت به‌معنای ورود است. (پس از احراز هویت موفق LDAP، اگر 2FA فعال باشد، کد TOTP همچنان بررسی می‌شود.)

فیلدهای بخش:

فیلد	متن (RU)	مقدار پیش‌فرض	توضیحات
فعال‌سازی همگام‌سازی LDAP	«Включить LDAP-синхронизацию» (`enable`)	**false**	کلید اصلی یکپارچه‌سازی LDAP.
میزبان LDAP	«LDAP-хост» (`host`)	خالی	آدرس سرور LDAP.
پورت LDAP	«Порт LDAP» (`port`)	**389**	پورت برای LDAPS معمولاً 636.
استفاده از TLS (LDAPS)	«Использовать TLS (LDAPS)» (`useTls`)	**false**	هنگام فعال‌سازی، طرح `ldaps://` استفاده می‌شود (به‌طور پیش‌فرض — با بررسی گواهی سرور).

فیلد	متن (RU)	مقدار پیش‌فرض	توضیحات
رد کردن بررسی گواهی TLS	«رد کردن بررسی گواهی TLS» (`ldapInsecureSkipVerify`)	**false**	افزوده‌شده در 3.4.2. بررسی گواهی سرور هنگام LDAPS را غیرفعال می‌کند — برای CAهای داخلی/خودامضا. راهنما: «ناامن — بررسی گواهی سرور را غیرفعال می‌کند. فقط با CAهای داخلی/نامعتبر استفاده کنید». این کلید **تنها** زمانی در دسترس است که «استفاده از TLS (LDAPS)» فعال باشد.
Bind DN	«Bind DN» (`bindDn`)	خالی	DN حساب سرویسی برای bind/جستجوی اولیه. اگر خالی باشد — bind انجام نمی‌شود (جستجوی ناشناس).
رمز عبور bind	راهنماها: «Настроено; оставьте пустым, чтобы сохранить текущий пароль.» / «Настроено» / «Не настроено. — введите новое значение для замены»	خالی	رمز عبور برای `Bind DN`. به‌صورت جداگانه ذخیره می‌شود؛ برای حفظ مقدار قبلی، فیلد را خالی می‌گذارند.
Base DN	«Base DN» (`baseDn`)	خالی	ریشه زیردرختی که جستجو در آن انجام می‌شود (جستجو بازگشتی، در کل زیردرخت).
فیلتر کاربر	«Фильтр пользователя» (`userFilter`)	`(objectClass=person)`	فیلتر LDAP برای انتخاب حساب‌ها. هنگام احراز هویت، لاگین با کاراکترگریزی (экранирование) در فیلتر قرار داده می‌شود.
صفت کاربر (username/email)	«Атрибут пользователя (username/email)» (`userAttr`)	`mail`	صفتی که با لاگین/شناسه کلاینت تطبیق داده می‌شود (مثلاً `mail` یا `uid`).
صفت پرچم VLESS	«Атрибут VLESS-флага» (`vlessField`)	`vless_enabled`	صفتی که تعیین می‌کند آیا دسترسی VLESS کلاینت باید فعال باشد.
صفت پرچم عمومی (اختیاری)	«Общий атрибут флага (опц.)» (`flagField`)، راهنما «Если задано, переопределяет флаг VLESS — напр. shadowInactive.»	خالی	اگر تعیین شود، به‌جای `vless_enabled` استفاده می‌شود.
مقادیر Truthy	«Truthy-значения» (`truthyValues`)، راهنما «Через запятую; по умолчанию: true,1,yes,on»	`true,1,yes,on`	فهرست مقادیر صفت پرچم که به‌عنوان «فعال» تلقی می‌شوند.
معکوس کردن پرچم	«Инвертировать флаг» (`invertFlag`)، راهنما «Включите, когда атрибут означает «отключено» (напр. shadowInactive).»	**false**	معنای پرچم را معکوس می‌کند.
زمان‌بندی همگام‌سازی	«Расписание синхронизации» (`syncSchedule`)، راهنما «Строка типа cron, напр. @every 1m»	`@every 1m`	تناوب همگام‌سازی در قالب شبه‌cron.

فیلد	متن (RU)	مقدار پیش‌فرض	توضیحات
تگ‌های inbound	«Теги входящих» (`inboundTags`)، راهنما «Входящие, на которых LDAP-синхронизация может авто-создавать или авто-удалять клиентов.»	خالی	محدود می‌کند که روی کدام inbound عملیات خودکار مجاز است. اگر inbound وجود نداشته باشد: «Входящие не найдены. Сначала создайте входящий.»
ایجاد خودکار کلاینت‌ها	«Авто-создание клиентов» (`autoCreate`)	**false**	ایجاد کلاینت در inbound مشخص‌شده، اگر در دایرکتوری ظاهر شده باشد.
حذف خودکار کلاینت‌ها	«Авто-удаление клиентов» (`autoDelete`)	**false**	حذف کلاینت، اگر از دایرکتوری ناپدید شده باشد.
حجم پیش‌فرض (گیگابایت)	«Объём по умолчанию (ГБ)» (`defaultTotalGb`)	**0**	محدودیت ترافیک برای کلاینت‌های ایجادشده خودکار (0 = بدون محدودیت).
مدت پیش‌فرض (روز)	«Срок по умолчанию (дни)» (`defaultExpiryDays`)	**0**	مدت اعتبار برای کلاینت‌های ایجادشده خودکار (0 = بدون انقضا).
محدودیت IP پیش‌فرض	«Лимит IP по умолчанию» (`defaultIpLimit`)	**0**	محدودیت تعداد IP همزمان (0 = بدون محدودیت).

ویژگی‌های منطق پرچم همگام‌سازی: هنگام خواندن صفت پرچم (`flagField` ، به‌طور پیش‌فرض `vless_enabled`)، مقدار «فعال» در نظر گرفته می‌شود اگر در فهرست مقادیر truthy باشد؛ هنگام فعال بودن معکوس‌سازی، نتیجه به مقدار مخالف تغییر می‌کند. صفت کاربر (`userAttr`) به‌عنوان کلید تطبیق (email/نام) استفاده می‌شود — رکوردهای بدون مقدار این صفت نادیده گرفته می‌شوند.

> امنیت: توصیه می‌شود **TLS (LDAPS)** را فعال کنید تا رمزهای عبور bind و رمزهای عبور بررسی‌شده به‌صورت متن آشکار منتقل نشوند، و برای `Bind DN` از حسابی با حداقل حقوق خواندن لازم استفاده کنید.

مثال: پیکربندی معمول همگام‌سازی LDAP (Active Directory). پر کردن فیلدهای بخش برای دایرکتوری‌ای که وضعیت دسترسی در صفتی شبیه پرچم `userAccountControl` نگه‌داری می‌شود و تطبیق بر اساس ایمیل انجام می‌شود:

فیلد	مقدار
میزبان LDAP	`ldap.example.com`
پورت LDAP	`636`
استفاده از TLS (LDAPS)	فعال
Bind DN	`CN=svc-3xui,OU=Service,DC=example,DC=com`
Base DN	`OU=Users,DC=example,DC=com`
فیلتر کاربر	`(objectClass=person)`
صفت کاربر (username/email)	`mail`
صفت پرچم VLESS	`vless_enabled`
مقادیر Truthy	`true,1,yes,on`
زمان‌بندی همگام‌سازی	`@every 5m`

با چنین تنظیماتی، هر 5 دقیقه پنل زیردرخت `OU=Users` را پیمایش می‌کند، کلاینت‌ها را بر اساس `mail` تطبیق می‌دهد و دسترسی VLESS را بر اساس مقدار `vless_enabled` فعال/غیرفعال می‌کند.

---

3. نمای کلی / داشبورد

داشبورد («داشبورد»، در رابط انگلیسی — *Overview*) صفحه شروع پنل است. این صفحه وضعیت سرور و فرآیند Xray را به‌صورت زنده نشان می‌دهد. تمام مقادیر از سمت سرور ارسال می‌شوند. زمان‌بند پس‌زمینه هر **2 ثانیه** یک اسنپ‌شات جدید می‌سازد و از طریق WebSocket به تمام تب‌های باز ارسال می‌کند؛ هر یک دقیقه یک‌بار ردیف‌های متریک انباشته‌شده روی دیسک ذخیره می‌شوند. اندپوینت HTTP با مسیر `GET /status` آخرین اسنپ‌شات کش‌شده را برمی‌گرداند.

در ادامه هر مقدار و هر عنصر کنترلی صفحه بررسی شده است.

در منوی کناری پنل (و در کشوی موبایل) از نسخه 3.4.2 دکمه **«مستندات»** (آیکن کتاب) وجود دارد — مستندات رسمی `https://docs.sanaei.dev/` را باز می‌کند. همچنین در 3.4.2 یک بهبود سراسری **دسترس‌پذیری** انجام شده است: دکمه‌های آیکنی و عناصر قابل کلیک برچسب‌های aria و فعال‌سازی با Enter/Space دریافت کرده‌اند (برای صفحه‌خوان‌ها و پیمایش با صفحه‌کلید).

3.1. اصول کلی جمع‌آوری داده

- اسنپ‌شات توسط کتابخانه `gopsutil` ساخته می‌شود. اگر یک اندازه‌گیری خاص ناموفق باشد، فیلد مربوطه صفر می‌ماند و یک هشدار در لاگ نوشته می‌شود (`get cpu percent failed`، `get uptime failed` و غیره) — این کل داشبورد را از کار نمی‌اندازد، فقط کارت مربوطه مقدار N-A/0 نشان می‌دهد.
- سرعت‌های «لحظه‌ای» (درصد CPU، شبکه، دیسک I/O) به‌صورت اختلاف بین اسنپ‌شات فعلی و قبلی تقسیم‌بر فاصله زمانی بر حسب ثانیه محاسبه می‌شوند. به همین دلیل در اولین بارگذاری صفحه ممکن است مقادیر سرعت صفر باشند تا اندازه‌گیری دوم انجام شود.
- تاریخچه را می‌توان در بخش «تاریخچه سیستم» (*System History*) مشاهده کرد — نمودارها بر اساس همان ردیف‌های داده‌ای که در ادامه توضیح داده می‌شوند ساخته می‌شوند (بخش 3.12 را ببینید).

3.2. پردازنده (CPU)

کارت «پردازنده» (*CPU*) میزان بار فعلی پردازنده را به درصد، به‌همراه مشخصات خود پردازنده نشان می‌دهد.

مقدار	توضیح
بار CPU، %	سهم زمان پردازنده‌ای که در آخرین بازه استفاده شده است. با میانگین نمایی (EMA، ضریب `alpha = 0.3`) هموار می‌شود تا نوسانات ناگهانی شاخص را تکان ندهند. مقدار همیشه در بازه 0 تا 100 درصد محدود می‌شود. در اولین اندازه‌گیری مقدار 0 برگردانده می‌شود (مقداردهی اولیه نقطه پایه).
پردازنده‌های منطقی	تعداد هسته‌های منطقی — یعنی با احتساب Hyper-Threading.
هسته‌های فیزیکی	تعداد هسته‌های فیزیکی.
فرکانس	فرکانس پایه پردازنده بر حسب MHz. به‌صورت تنبل درخواست می‌شود و کش می‌شود: اولین اندازه‌گیری موفق ذخیره می‌شود، درخواست مجدد حداکثر هر 5 دقیقه یک‌بار انجام می‌شود و خود درخواست با تایم‌اوت 1.5 ثانیه محدود شده است (در برخی سیستم‌ها درخواست فرکانس کند پاسخ می‌دهد).

بار CPU از نظر الگوریتمی این‌گونه محاسبه می‌شود: اگر پیاده‌سازی بومی پلتفرم وجود داشته باشد از آن استفاده می‌شود، در غیر این صورت از دلتای شمارنده‌های زمان پردازنده (busy / total) محاسبه می‌شود. زمان Guest و GuestNice برای جلوگیری از دوبار شمردن حذف می‌شوند.

3.3. حافظه (RAM)

کارت «حافظه» (*RAM*) مقدار استفاده‌شده و کل را نشان می‌دهد. به‌صورت «استفاده‌شده / کل» و/یا درصد پر شدن نمایش داده می‌شود. در تاریخچه درصد ثبت می‌شود.

3.4. حافظه مجازی (Swap)

کارت «حافظه مجازی» (*Swap*) مقدار استفاده‌شده و کل را نشان می‌دهد. اگر فایل/پارتیشن swap پیکربندی نشده باشد (کل = 0)، مقدار صفر است؛ در صورت نبود swap، مقدار 0 در ردیف تاریخچه ثبت می‌شود.

3.5. دیسک (Storage)

کارت «دیسک» (*Storage*) مقدار استفاده‌شده و کل را نشان می‌دهد و **فقط پارتیشن ریشه** `/` را در نظر می‌گیرد. درصد پر بودن در تاریخچه «استفاده از دیسک» (*Disk Usage*) ثبت می‌شود. ورودی/خروجی دیسک (خواندن / نوشتن، بایت/ثانیه) نیز به‌صورت جداگانه به‌عنوان دلتای شمارنده‌ها در بازه جمع‌آوری می‌شود — این مقادیر در تب «دیسک I/O» تاریخچه نمایش داده می‌شوند.

3.6. مدت زمان اجرای سیستم (Uptime)

مقدار «مدت زمان اجرای سیستم» (*Uptime*). این زمان از لحظه راه‌اندازی **کل سرور** (بر حسب ثانیه) است، نه زمان اجرای پنل یا Xray. مدت زمان اجرای فرآیند Xray به‌صورت جداگانه ذخیره می‌شود (بخش 3.9 را ببینید)، همچنین تعداد تردهای پنل (در ترجمه — «رشته‌ها» / *Threads*).

حافظه اشغال‌شده توسط پنل

در کنار مقادیر فرآیند پنل، مقدار حافظه RAM که خود فرآیند 3X-UI اشغال کرده نمایش داده می‌شود. این مقدار از RSS واقعی فرآیند (آن‌گونه که سیستم‌عامل می‌بیند) گرفته می‌شود و با آنچه ابزارهای سیستمی نشان می‌دهند مطابقت دارد. عدد با آزاد شدن حافظه کاهش می‌یابد. پیش از این، پنل از شمارنده داخلی Go استفاده می‌کرد که مصرف حافظه را بیش از حد نشان می‌داد (مثلاً ~300 مگابایت روی سرور بیکار با یک کلاینت) و هرگز کاهش نمی‌یافت — اکنون این مشکل وجود ندارد. علاوه بر این، یک فرآیند پس‌زمینه دوره‌ای حافظه بلااستفاده را به سیستم‌عامل بازمی‌گرداند تا مقدار نمایشی مصرف واقعی را منعکس کند.

3.7. میانگین بار سیستم (Load average)

بلوک «بار سیستم» (*System Load*) — آرایه‌ای از سه عدد `[Load1, Load5, Load15]`. راهنما: «میانگین بار سیستم در 1، 5 و 15 دقیقه گذشته» (*System load average for the past 1, 5, and 15 minutes*). نمودار تاریخچه «میانگین بار سیستم (1 / 5 / 15 دقیقه)» نام دارد. مقادیر به‌صورت جداگانه در ردیف‌های تاریخچه ثبت می‌شوند: `load1`، `load5`، `load15`.

این یک مقدار استاندارد یونیکس است: میانگین تعداد فرآیندهایی که در صف اجرا هستند. معیار مقایسه با تعداد هسته‌هاست: بار ثابتی که از تعداد هسته‌های فیزیکی بیشتر باشد نشان‌دهنده اضافه‌بار است.

3.8. شبکه: سرعت و حجم کل ترافیک

فقط رابطهای فیزیکی در نظر گرفته می‌شوند. رابطهای مجازی و تونل حذف می‌شوند: `lo` / `lo0`، و هر چیزی که با `loopback`، `docker`، `br-`، `veth`، `virbr`، `tun`، `tap`، `wg`، `tailscale`، `zt` شروع شود. مقادیر روی تمام رابطهای باقی‌مانده جمع می‌شوند.

سرعت کلی (*Overall Speed*) — سرعت لحظه‌ای، دلتای شمارنده‌ها در بازه:

مقدار	توضیح
آپلود (برچسب «آپلود» / *Upload*)	سرعت ارسال، بایت/ثانیه.
دانلود (برچسب «دانلود» / *Download*)	سرعت دریافت، بایت/ثانیه.

حجم کل ترافیک (*Total Data*) — شمارنده‌های انباشته از زمان راه‌اندازی سیستم:

مقدار	توضیح
ارسال‌شده (برچسب «ارسال‌شده» / *Sent*)	مجموع بایت‌های ارسال‌شده.
دریافت‌شده (برچسب «دریافت‌شده» / *Received*)	مجموع بایت‌های دریافت‌شده.

علاوه بر این، سرعت بسته‌ها (بسته/ثانیه) و شمارنده‌های کل بسته‌ها نیز جمع‌آوری می‌شوند — در تب «بسته‌های شبکه» (*Network Packets*) تاریخچه نمایش داده می‌شوند. ردیف‌های تاریخچه شبکه: `netUp` ، `netDown` ، `pktUp` ، `pktDown` .

3.9. آدرس‌های IP سرور

بلوک «آدرس‌های IP سرور» (*IP Addresses*) مقادیر `IPv4` و `IPv6` را نشان می‌دهد. آدرس‌های خارجی از طریق سرویس‌های شخص ثالث تشخیص داده می‌شوند (`api4.ipify.org` ، `ipv4.icanhazip.com` ، `v4.api.ipinfo.io/ip` ، `ipv4.myexternalip.com/raw` ، `4.ident.me` ، `check-host.net/ip` برای IPv4 و مشابه برای IPv6). لیست به ترتیب تا اولین پاسخ موفق بررسی می‌شود؛ تایم‌اوت هر درخواست 3 ثانیه است.

ویژگی‌ها:

- نتیجه در طول عمر فرآیند **کش** می‌شود: آدرس تشخیص داده‌شده مجدداً درخواست نمی‌شود.
- اگر هیچ سرویسی پاسخ ندهد، فیلد `N/A` باقی می‌ماند. برای IPv6 در اولین `N/A` درخواست‌های IPv6 به‌طور کامل غیرفعال می‌شوند تا در شبکه‌های بدون IPv6 زمان هدر نرود.
- یک دکمه «چشم» برای پنهان/نمایش آدرس‌ها وجود دارد — راهنما: «پنهان یا نمایش آدرس‌های IP سرور» (*Toggle visibility of the IP*). این فقط یک پنهان‌سازی بصری در رابط است (مثلاً برای تصاویر اسکرین‌شات) و روی خود آدرس‌ها تأثیری ندارد.

3.10. اتصالات TCP/UDP

بلوک «تعداد اتصالات» (*Connection Stats*) تعداد کل اتصالات TCP و UDP فعال روی سرور را نشان می‌دهد (در کل سیستم، نه فقط Xray). نمودار تاریخچه — «اتصالات فعال (TCP / UDP)» (*Active Connections*)، ردیف‌های `tcpCount` ، `udpCount` .

3.11. وضعیت Xray و مدیریت فرآیند

کارت «Xray» وضعیت فرآیند Xray-core را نشان می‌دهد و امکان مدیریت آن را فراهم می‌کند.

وضعیت‌ها

مقدار	برچسب	ترجمه	زمان تنظیم
`running`	«در حال اجرا»	*Running*	فرآیند Xray در حال اجراست.
`stop`	«متوقف»	*Stopped*	فرآیند در حال اجرا نیست و خطای راه‌اندازی ثبت‌شده‌ای وجود ندارد.

مقدار	برچسب	ترجمه	زمان تنظیم
`error`	«خطا»	*Error*	فرآیند در حال اجرا نیست، اما خطای راه‌اندازی ثبت شده است. متن خطا در پنجره‌ای با عنوان «خطا در حین اجرای Xray» (*An error occurred while running Xray*) نمایش داده می‌شود.
—	«ناشناخته»	*Unknown*	در حالی که وضعیت هنوز دریافت نشده نمایش داده می‌شود.

نسخه Xray نیز در کنار وضعیت نشان داده می‌شود.

دکمه‌های کنترل

- **توقف** (*Stop*). فراخوانی `POST /stopXrayService`. در صورت موفقیت، پنل از طریق WebSocket وضعیت جدید `stop` و اعلان «Xray با موفقیت متوقف شد» (*Xray service has been stopped*) را ارسال می‌کند، در صورت خطا — وضعیت `error` همراه با متن خطا. مهم: اگر پنل از طریق خود Xray در دسترس است، متوقف کردن Xray ممکن است ارتباط با پنل را قطع کند — در اتصال مستقیم به پنل مشکلی وجود ندارد.
- **راه‌اندازی مجدد** (*Restart*). فراخوانی `POST /restartXrayService`. قبل از عمل، تأییدیه «راه‌اندازی مجدد xray؟» با توضیح «سرویس xray را با پیکربندی ذخیره‌شده مجدداً راه‌اندازی می‌کند» نشان داده می‌شود. در صورت موفقیت — وضعیت `running` و اعلان «Xray با موفقیت راه‌اندازی مجدد شد» (*Xray service has been restarted successfully*). راه‌اندازی مجدد پیکربندی ذخیره‌شده فعلی را اعمال می‌کند — پس از تغییر تنظیمات از آن استفاده کنید.

> توجه. در این فورک برای داشبورد مدیریت کامل Start / Stop / Restart برای همه انواع مجوز اضافه شده است؛ در رابط اصلی 3x-ui دکمه جداگانه «استارت» وجود ندارد — راه‌اندازی از طریق راه‌اندازی مجدد انجام می‌شود.

دکمه مشاهده لاگ‌های Xray

در کارت Xray دکمه مشاهده لاگ‌های Xray (*Logs*) وجود دارد. این دکمه فقط زمانی نمایش داده می‌شود که access-log در پیکربندی Xray تنظیم شده باشد: مشاهده‌گر داخلی دقیقاً این فایل را می‌خواند، بنابراین بدون access-log دکمه پنهان است. نمایش دکمه به ویژگی جداگانه `accessLogEnable` وابسته است و دیگر به محدودیت IP بستگی ندارد — لیست آنلاین و محدودیت آدرس‌های IP حتی بدون access-log کار می‌کنند (بخش 8 را ببینید).

انتخاب نسخه Xray

بخش «انتخاب نسخه» (*Version*) امکان تغییر Xray-core به نسخه دیگری را فراهم می‌کند. لیست نسخه‌ها از طریق `GET /getXrayVersion` بارگذاری می‌شود:

- منبع — GitHub API مخزن `XTLS/Xray-core` (`/releases`). درخواست‌ها به مدت **15 دقیقه** کش می‌شوند؛ در صورت خرابی GitHub آخرین لیست دریافت‌شده با موفقیت برگردانده می‌شود تا پیکر خالی نباشد.
- فقط نسخه‌هایی با قالب `X.Y.Z` و **نه قدیمی‌تر از 26.4.25** در لیست قرار می‌گیرند.

راهنماها: «نسخه مورد نظر را انتخاب کنید» (*Choose the version you want to switch to*.) و هشدار «مهم: نسخه‌های قدیمی ممکن است از تنظیمات فعلی پشتیبانی نکنند» (*Choose carefully, as older versions may not be compatible with current configurations*.).

تغییر نسخه: `POST /installXray/:version`. سناریو:

مثال. تغییر به نسخه خاصی از Xray-core (کوکی نشست باید از قبل توسط احراز هویت دریافت شده باشد):

```
curl -X POST 'https://panel.example.com:2053/xpanel/installXray/v25.6.8' \
  -b cookie.txt
```

در اینجا `v25.6.8` یک تگ از لیستی است که `GET /getXrayVersion` برمی‌گرداند. نسخه باید در این لیست وجود داشته باشد، در غیر این صورت پنل درخواست را رد می‌کند. از نسخه 3.4.2 حداقل نسخه قابل‌نصب Xray به **26.6.27** افزایش یافته است (هسته Xray 26.6.27 همراه آن عرضه می‌شود)، بنابراین ساخت‌های قدیمی‌تر برای به‌روزرسانی در دسترس نیستند.

1. نسخه انتخاب‌شده در لیست نسخه‌های فعلی بررسی می‌شود (در غیر این صورت — رد درخواست).
2. Xray متوقف می‌شود.
3. آرشیو `Xray-<os>-<arch>.zip` برای سیستم‌عامل و معماری فعلی از GitHub دانلود می‌شود (پشتیبانی از arm64، amd64/64-، s390x، 386/32، arm32-v7a/v6/v5، v8a؛ برای Windows — `xray.exe`). حجم آرشیو و باینری به 200 مگابایت محدود شده است.
4. باینری به‌صورت اتمیک جایگزین می‌شود (از طریق فایل موقت + تغییر نام) و به‌عنوان اجرایی علامت‌گذاری می‌شود.
5. Xray مجدداً راه‌اندازی می‌شود.

قبل از تغییر، گفتگوی «تغییر نسخه Xray» (*Do you really want to change the Xray version?*) با توضیح «این نسخه Xray را به #version# تغییر خواهد داد» نشان داده می‌شود. در صورت موفقیت — اعلان «Xray با موفقیت به‌روزرسانی شد» (*Xray updated successfully*).

3.12. به‌روزرسانی پنل (3X-UI)

بلوک بررسی به‌روزرسانی پنل داده‌ها از طریق `GET /getPanelUpdateInfo` دریافت می‌شوند:

فیلد	توضیح
نسخه فعلی پنل	نسخه پنل نصب‌شده.
آخرین نسخه پنل	آخرین نسخه 3x-ui دریافت‌شده از GitHub.
به‌روزرسانی موجود است	نشانه‌ای که آخرین نسخه جدیدتر از نسخه فعلی است. اگر به‌روزرسانی لازم نباشد — «پنل به‌روز است» / «به‌روزشده» نمایش داده می‌شود.

دکمه **«به‌روزرسانی پنل»** (*Update Panel*) `POST /updatePanel` را اجرا می‌کند. راهنما: «این 3X-UI را به آخرین نسخه به‌روزرسانی می‌کند و سرویس پنل را مجدداً راه‌اندازی می‌کند». قبل از شروع — تأییدیه «آیا واقعاً می‌خواهید پنل را به‌روزرسانی کنید؟» با متن «این 3X-UI را به نسخه #version# به‌روزرسانی می‌کند و سرویس پنل را مجدداً راه‌اندازی می‌کند».

ویژگی‌ها و محدودیت‌ها:

- به‌روزرسانی خودکار **فقط در Linux** پشتیبانی می‌شود (در سیستم‌عامل‌های دیگر خطا برگردانده می‌شود).
- اسکریپت به‌روزرسان از مخزن رسمی دانلود می‌شود (`raw.githubusercontent.com/MHSanaei/3x-ui/main/update.sh`، محدودیت 2 مگابایت) و از طریق `bash` اجرا می‌شود، در صورت امکان به‌صورت ایزوله از طریق `systemd-run`.
- در صورت راه‌اندازی موفق «به‌روزرسانی پنل شروع شد» (*Panel update started*) نمایش داده می‌شود؛ اگر بررسی به‌روزرسانی ناموفق بود — «بررسی به‌روزرسانی پنل ناموفق بود». در حین نصب هشدار «نصب در حال انجام است. صفحه را تازه‌سازی نکنید» نمایش داده می‌شود.

3.13. به‌روزرسانی فایل‌های جغرافیایی (GeoIP / GeoSite)

دکمه/گفتگوی به‌روزرسانی پایگاه‌های داده جغرافیایی `POST /updateGeofile` (همه فایل‌ها) یا `POST /updateGeofile/:fileName` (یک فایل) را فراخوانی می‌کند. به‌روزرسانی بر اساس لیست سفید سختگیرانه‌ای از نام‌ها و منابع کار می‌کند:

فایل	منبع
`geoip.dat` ، `geosite.dat`	`Loyalsoldier/v2ray-rules-dat` (latest)
`geoip_IR.dat` ، `geosite_IR.dat`	`chocolate4u/Iran-v2ray-rules` (latest)
`geoip_RU.dat` ، `geosite_RU.dat`	`runetfreedom/russia-v2ray-rules-dat` (latest)

رفتار:

- نام فایل اعتبارسنجی می‌شود: `..` ، اسلش، مسیرهای مطلق مجاز نیستند؛ فقط `[a-zA-Z0-9._-]+.dat` مجاز است. فایل‌های خارج از لیست سفید دانلود نمی‌شوند.
- درخواست مشروط `If-Modified-Since` استفاده می‌شود: اگر فایل روی سرور منبع تغییر نکرده باشد (HTTP 304)، مجدداً دانلود نمی‌شود، فقط تاریخ تغییر به‌روزرسانی می‌شود.
- پس از بارگذاری Xray **مجدداً راه‌اندازی می‌شود** (تا پایگاه‌های داده جدید را بارگیری کند).

مثال. به‌روزرسانی فقط پایگاه‌های داده جغرافیایی روسیه، بدون تغییر سایر فایل‌ها:

```
curl -X POST 'https://panel.example.com:2053/xpanel/updateGeofile/geoip_RU.dat' -b
cookie.txt
curl -X POST 'https://panel.example.com:2053/xpanel/updateGeofile/geosite_RU.dat' -b
cookie.txt
```

برای به‌روزرسانی همه فایل‌های لیست سفید به‌طور همزمان — `POST /updateGeofile` را بدون نام فایل فراخوانی کنید.

- گفتگوها: «آیا واقعاً می‌خواهید فایل جغرافیایی را به‌روزرسانی کنید؟» با «این فایل #filename# را به‌روزرسانی خواهد کرد» برای یک فایل و «این همه فایل‌های جغرافیایی را به‌روزرسانی خواهد کرد» برای دکمه «به‌روزرسانی همه». موفقیت — «فایل‌های جغرافیایی با موفقیت به‌روزرسانی شدند».

3.14. پشتیبان‌گیری و بازیابی پایگاه داده

بلوک «پشتیبان‌گیری و بازیابی» (*Backup & Restore*). رفتار به DBMS مورد استفاده بستگی دارد (SQLite به‌صورت پیش‌فرض یا PostgreSQL).

صدور پایگاه داده (پشتیبان‌گیری)

دکمه «صدور پایگاه داده» / «پشتیبان‌گیری» (*Back Up*) `GET /getDb` را فراخوانی می‌کند. فایل به‌عنوان پیوست ارسال می‌شود:

- **SQLite**: ابتدا checkpoint انجام می‌شود (تخلیه WAL)، سپس فایل `x-ui.db` دانلود می‌شود. راهنما: «برای دانلود فایل .db حاوی نسخه پشتیبان پایگاه داده فعلی کلیک کنید...».
- **PostgreSQL**: یک dump با فرمت سفارشی `x-ui.dump` دانلود می‌شود (`pg_dump --format=custom --no-owner --no-privileges`). ابزارهای کلاینت PostgreSQL باید روی سرور نصب باشند؛ در غیر این صورت — خطای مربوط به نبود `pg_dump` .

وارد کردن پایگاه داده (بازیابی)

دکمه «وارد کردن پایگاه داده» / «بازیابی» (*Restore*) فایل را از طریق `POST /importDB` (فیلد فرم `db`) بارگذاری می‌کند. راهنما: «برای انتخاب و بارگذاری فایل .db... جهت بازیابی پایگاه داده از نسخه پشتیبان کلیک کنید».

سناریوی **SQLite** ایمن، با بازگشت به حالت قبل:

1. فایل از نظر قالب SQLite بررسی می‌شود و در یک فایل موقت ذخیره می‌شود، سپس یکپارچگی آن بررسی می‌شود.
2. Xray متوقف می‌شود، پایگاه داده فعلی بسته می‌شود و به `*.backup` تغییر نام می‌یابد (fallback).
3. فایل جدید جای پایگاه داده کاری را می‌گیرد، مقداردهی اولیه و مهاجرت انجام می‌شود. اگر مشکلی پیش آمد — fallback بازیابی می‌شود.
4. Xray مجدداً راه‌اندازی می‌شود.

برای **PostgreSQL** فایل `.dump` بارگذاری می‌شود (امضای `PGDMP` بررسی می‌شود) و از طریق `pg_restore --clean --if-exists --single-transaction …` اعمال می‌شود. راهنما صریحاً هشدار می‌دهد: «این تمام داده‌های فعلی را جایگزین خواهد کرد».

پیام‌ها: «پایگاه داده با موفقیت وارد شد»، «خطایی در حین وارد کردن پایگاه داده رخ داد»، «...در حین خواندن پایگاه داده»، «...در حین دریافت پایگاه داده».

فایل مهاجرت (بین SQLite و PostgreSQL)

دکمه «دانلود فایل مهاجرت» (*Download Migration*) `GET /getMigration` را فراخوانی می‌کند و یک صدور قابل انتقال برای اجرای پنل روی DBMS دیگری تولید می‌کند:

- روی **SQLite** فایل `x-ui.dump` (dump متنی SQL) دانلود می‌شود.
- روی **PostgreSQL** فایل `x-ui.db` دانلود می‌شود — یک پایگاه داده SQLite آماده که از داده‌های PostgreSQL ساخته شده است.

3.15. عناصر اضافی رابط کاربری

- **شاخص کلاینت‌های آنلاین.** داشبورد ردیف `online` (*Online Clients* / «کلاینت‌های آنلاین») را نگه می‌دارد — تعداد کلاینت‌هایی که اتصال فعال دارند. در حین اجرای Xray محاسبه می‌شود (در غیر این صورت 0) و با همان چرخه 2 ثانیه‌ای در تاریخچه ثبت می‌شود. نمودار — تب «آنلاین».
- **تاریخچه سیستم (نمودارها).** دکمه/بخش «نمودارها» → «تاریخچه سیستم» با تب‌ها: «پهنای باند»، «بسته‌ها»، «دیسک I/O»، «آنلاین»، «بار»، «اتصالات»، «استفاده از دیسک». داده‌ها از طریق `GET /history/:metric/:bucket` دریافت می‌شوند؛ بازه‌های مجاز تجمیع (bucket، ثانیه): **2, 30, 60, 180, 360, 720, 1440, 2880, 10080**، تا 60 نقطه به هر تب می‌رسد. در انتخابگر بازه زمانی روی صفحه دکمه‌های **2m, 1h, 3h, 6h, 12h, 24h, 2d, 7d** در دسترس هستند (bucketهای `2, 60, 180, 360, 720, 1440, 2880, 10080` به ترتیب). در بازه‌های طولانی **2d** و **7d** برچسب‌های زمانی روی محور با تاریخ به فرمت `MM-DD HH:MM` تکمیل می‌شوند. ذخیره‌سازی با سه سطح کاهش‌دهنده (rollup) سازماندهی شده است: داده‌های تازه با گام 2 ثانیه برای **یک ساعت** گذشته نگه داشته می‌شوند، سپس به گام 1 دقیقه برای **48 ساعت** و به گام 10 دقیقه برای **7 روز** میانگین‌گیری می‌شوند. بنابراین نمودارها (RAM، CPU، ترافیک، بسته‌ها، اتصالات، دیسک، آنلاین، بار) را می‌توان برای **تا 7 روز** گذشته مشاهده کرد (قبلاً تا 48 ساعت بود)، و هرچه به گذشته دورتر بروید جزئیات درشت‌تر می‌شود. متریک‌های مجاز: `cpu, mem, swap, netUp, netDown, pktUp, pktDown, diskRead, diskWrite, diskUsage, tcpCount, udpCount, online, load1, load5, load15`. برچسب «2 دقیقه اخیر» متناظر با bucket = 2 (حالت زمان واقعی) است.

مثال. دریافت ردیف بار CPU برای ~2 دقیقه اخیر (bucket = 2 ثانیه، تا 60 نقطه) و همان ردیف با تجمیع 5 دقیقه‌ای (bucket = 300 ثانیه):

```
curl 'https://panel.example.com:2053/xpanel/history/cpu/2' -b cookie.txt
curl 'https://panel.example.com:2053/xpanel/history/cpu/300' -b cookie.txt
```

متریک را می‌توان با هر متریک مجاز جایگزین کرد (`mem`، `netUp`، `tcpCount`، `load1` و غیره). bucket خارج از لیست سفید `2, 30, 60, 180, 360, 720, 1440, 2880, 10080` رد خواهد شد.

- **متریک‌های Xray** — یک بلوک جداگانه با مصرف حافظه و garbage collection مربوط به Xray (ردیف‌های `xrAlloc, xrSys, xrHeapObjects, xrNumGC, xrPauseNs`) و «Observatory» (وضعیت اتصالات outbound). فقط در صورتی کار می‌کنند

که بلوک `metrics` در پیکربندی Xray تنظیم شده باشد (`listen 127.0.0.1:11111` ، تگ `metrics_out`)؛ در غیر این صورت «اندپوینت متریک Xray پیکربندی نشده است» نمایش داده می‌شود. در پنجره متریک‌های Xray انتخابگر بازه زمانی خاص خود با دکمه‌های **2m, 1h, 3h, 6h, 12h** (bucketهای `2, 60, 180, 360, 720`) وجود دارد.

مثال بلوکی که کارت متریک‌های Xray را فعال می‌کند. در بخش تنظیمات Xray باید همزمان `metrics` (با تگ) و inbound که این تگ را گوش می‌دهد وجود داشته باشد:

```
{
  "metrics": {
    "tag": "metrics_out"
  },
  "inbounds": [
    {
      "listen": "127.0.0.1",
      "port": 11111,
      "protocol": "dokodemo-door",
      "settings": { "address": "127.0.0.1" },
      "tag": "metrics_out"
    }
  ]
}
```

آدرس `127.0.0.1:11111` عمداً در معرض عموم قرار نمی‌گیرد — پنل آن را به‌صورت محلی بررسی می‌کند.

- **سوئیچ تم تاریک.** در منوی کلی/هدر قرار دارد، نه در خود داشبورد. گزینه‌ها: «تم» (*Theme*) با گزینه‌های «تاریک» و «خیلی تاریک» (*Ultra Dark*). این یک تنظیم بصری صرف است و روی عملکرد پنل تأثیری ندارد.
- **سایر لینک‌ها** در اطراف داشبورد (از منو/نوار پایین): «لاگ‌ها»، «پیکربندی» — مشاهده JSON نهایی Xray (`GET /getConfigJson`)، «مستندات».

---

4. Inboundها: ایجاد و پارامترهای عمومی

بخش **«ورودی‌ها»** (به انگلیسی *Inbounds*) فهرستی از همه نقاط ورود Xray است که کلاینت‌ها از طریق آن‌ها متصل می‌شوند. هر inbound هم فیلدهای «پنلی» (یادداشت، محدودیت ترافیک، زمان‌بندی بازنشانی) و هم بلوک‌های خام JSON پیکربندی Xray (`settings` ، `streamSettings` ، `sniffing`) را نگه می‌دارد.

ایجاد با دکمه **«ایجاد اتصال»** (*Add Inbound*) و ویرایش با **«ویرایش اتصال»** (*Modify Inbound*) انجام می‌شود. هر دو عملیات به اندپوینت‌های API یعنی `POST /add` و `POST /update/:id` ارسال می‌شوند.

در ادامه همه فیلدهای فرم بررسی شده‌اند که **به** تنظیمات پروتکل خاص (کلاینت‌ها، رمزنگاری، REALITY/TLS) ربطی **ندارند** و **به** ترابری/جریان (زبانه‌های **«جریان»**، **«امنیت»**) نیز مربوط **نیستند** — این‌ها موضوع بخش‌های جداگانه‌اند.

4.1. فیلدهای عمومی فرم

Remark (یادداشت)

پارامتر	مقدار
فیلد	`remark`
نوع	رشته
پیش‌فرض	خالی

نام قابل خواندن برای انسان از inbound که در فهرست و در عنوان دیالوگ‌ها («حذف اتصال "{remark}"؟» و مانند آن) نمایش داده می‌شود. برچسب فیلد — **«یادداشت»**. روی کارکرد Xray تأثیری ندارد و فقط برای راحتی مدیریت لازم است؛ توصیه می‌شود نام‌های یکتا و معنادار تعیین کنید، زیرا در نام فایل‌های صادرشده و در تأیید عملیات گروهی درج می‌شوند.

Protocol (پروتکل)

پارامتر	مقدار
فیلد	`protocol`
برچسب	**«پروتکل»**
اعتبارسنجی	`required,oneof=vmess vless trojan shadowsocks wireguard hysteria http mixed tunnel tun`

فهرست بازشوی پروتکل inbound. مقادیر مجاز:

مقدار	یادداشت
`vmess`	
`vless`	
`trojan`	
`shadowsocks`	
`wireguard`	

مقدار	یادداشت
`hysteria`	Hysteria v2 همان `hysteria` با `streamSettings.version = 2` است؛ پروتکل جداگانه‌ای وجود ندارد
`http`	
`mixed`	socks/http روی یک پورت
`tunnel`	
`tun`	توسط اعتبارسنج پذیرفته می‌شود؛ ثابت پروتکل جداگانه‌ای ندارد

فیلد اجباری است (`required`). انتخاب پروتکل تعیین می‌کند که کدام فیلدهای تنظیمات کلاینت‌ها و کدام ترابری در دسترس خواهند بود (به بخش‌های مخصوص پروتکل مراجعه کنید).

> مهم: هنگام ذخیره، سرویس `streamSettings` را نرمال‌سازی می‌کند. تنظیمات ترابری فقط برای پروتکل‌های `vmess` ، `vless` ، `trojan` ، `shadowsocks` ، `hysteria` باقی می‌مانند؛ برای بقیه (`http` ، `mixed` ، `tunnel` ، `wireguard` ، `tun`) فیلد `streamSettings` **به‌اجبار پاک می‌شود**.

برای inbound از نوع `tunnel` /TProxy که بلوک `streamSettings` آن کلید `security` ندارد (حالت بدون‌ترابری)، فرم بدون خطای اعتبارسنجی `streamSettings.security Invalid input` باز و ذخیره می‌شود.

Listen IP (IP گوش‌دادن)

پارامتر	مقدار
فیلد	`listen`
نوع	رشته
پیش‌فرض	خالی → Xray روی `0.0.0.0` (همه IPها) گوش می‌دهد

آدرس IP که inbound روی آن اتصال‌ها را می‌پذیرد. راهنمای فیلد:

> «برای گوش‌دادن به همه آدرس‌های IP خالی بگذارید».

هنگام تولید پیکربندی Xray، مقدار خالی با `0.0.0.0` جایگزین می‌شود. علاوه بر IP، فیلد **مسیر Unix-socket** را نیز می‌پذیرد — راهنما:

> «همچنین می‌توانید مسیر Unix-socket (مثلاً /run/xray/in.sock) یا نام سوکت انتزاعی با پیشوند @ (مثلاً @xray/in.sock) را وارد کنید تا به‌جای پورت TCP به سوکت گوش داده شود — در این حالت پورت را 0 قرار دهید».

بنابراین فیلد دو شکل از Unix-socket را می‌پذیرد: مسیر در سیستم فایل (`/run/xray/in.sock`) و نام انتزاعی سوکت با پیشوند `@` (`@xray/in.sock`). در هر دو حالت `Port` را برابر `0` قرار دهید.

این فیلد را زمانی تغییر می‌دهند که لازم باشد inbound به یک رابط محدود شود (مثلاً `127.0.0.1` برای inboundی که فقط به‌عنوان هدف fallback پشت Nginx کار می‌کند) یا زمانی که inbound به Unix-socket گوش می‌دهد.

مثال. inboundی که فقط به رابط محلی گوش می‌دهد (هدف معمول fallback پشت Nginx) و Unix-socket:

```
listen = 127.0.0.1   порт = 8443
listen = /run/xray/in.sock   порт = 0
```

Port (پورت)

پارامتر	مقدار
فیلد	`port`
برچسب	**«پورت»**
اعتبارسنجی	`gte=0,lte=65535`
پیش‌فرض	— (توسط کاربر تعیین می‌شود)

پورت TCP/UDP برای گوش‌دادن. مقادیر از `0` تا `65535` مجاز هستند. مقدار `0` فقط همراه با گوش‌دادن روی Unix-socket استفاده می‌شود (به بالا مراجعه کنید).

هنگام ذخیره، سرویس تداخل پورت را بررسی می‌کند: دو inbound نمی‌توانند همزمان `listen:port` های متداخل را برای یک ترابری یکسان (TCP/UDP) اشغال کنند. ترابری از روی پروتکل و `settings` / `streamSettings` محاسبه می‌شود: برای مثال، `hysteria` و `wireguard` همیشه UDP را اشغال می‌کنند، `kcp` / `quic` نیز UDP، و بیشتر بقیه — TCP. در صورت تداخل، ذخیره با خطا رد می‌شود.

به‌طور جداگانه، پنل اجازه نمی‌دهد **پورت رزروشده API داخلی Xray** (تگ `api` ، به‌صورت پیش‌فرض `62789` روی `127.0.0.1`) اشغال شود: یک inbound محلی TCP که آدرس گوش‌دادن آن با این پورت روی loopback تداخل دارد، با همان خطای تداخل پورت رد می‌شود. پورت واقعی API از قالب پیکربندی Xray خوانده می‌شود (با مقدار جایگزین `62789`). روی گره‌ها (nodes) این محدودیت اعمال نمی‌شود — آن‌ها Xray مخصوص خود را دارند.

> تگ Xray (`Tag` ، یکتا) به‌صورت خودکار از روی پورت و ترابری در قالب `in-<порт>-<tcp|udp|tcpudp|any>` تولید می‌شود؛ برای inboundی که روی یک گره مستقر شده، پیشوند `n<nodeId>-` افزوده می‌شود. در صورت تصادم، به تگ `-2` ، `-3` و غیره اضافه می‌شود. کاربر معمولاً تگ را ویرایش نمی‌کند.

Total traffic (کل ترافیک، GB)

پارامتر	مقدار
فیلد	`total` (به **بایت**)
برچسب	**«مصرف کل»**
پیش‌فرض	`0`

محدودیت کل ترافیک inbound. در فرم مقدار به گیگابایت وارد می‌شود و در پایگاه داده به بایت ذخیره می‌شود. راهنمای فیلد:

> «= نامحدود. (واحد: GB)».

یعنی `0` **به‌معنای نامحدود است**. این محدودیت در سطح کل inbound است (نه کلاینت‌های جداگانه)؛ ترافیک واقعی مصرف‌شده در فیلدهای `up` (ارسال‌شده) و `down` (دریافت‌شده) ذخیره و با `total` مقایسه می‌شود.

Expiry date / Duration (تاریخ پایان / مدت)

پارامتر	مقدار
فیلد	`expiryTime` (برچسب زمانی Unix)
برچسب	**«تاریخ پایان»** (به انگلیسی *Duration*)
پیش‌فرض	خالی / `0`

مدت اعتبار inbound. راهنما:

> «برای بی‌نهایت بودن خالی بگذارید».

مقدار خالی (`0`) به‌معنای inbound بدون انقضا است. مقدار به‌صورت برچسب زمانی Unix ذخیره می‌شود؛ فرم اجازه می‌دهد هم یک تاریخ مشخص و هم یک مدت بر حسب روز (شمارش نسبی از لحظه کنونی — برچسب انگلیسی فیلد *Duration*) تعیین کنید.

Enabled (فعال‌سازی)

پارامتر	مقدار
فیلد	`enable`
برچسب	**«فعال‌سازی»** (به انگلیسی *Enabled*)
پیش‌فرض	هنگام ایجاد تعیین می‌شود

نشانگر فعال بودن inbound. تغییر این پرچم در فهرست توسط یک اندپوینت «سبک» جداگانه `POST /setEnable/:id` پردازش می‌شود، نه به‌روزرسانی کامل — این کار به‌طور خاص انجام شده تا هنگام هر کلیک روی کلید فعال‌سازی یک inbound با هزاران کلاینت، کل بلوک `settings` (همه کلاینت‌ها) دوباره سریالایز نشود. هنگام غیرفعال‌سازی، inbound از Xray در حال اجرا حذف می‌شود و هنگام فعال‌سازی دوباره افزوده می‌شود.

Node / Deploy to (گره / استقرار روی)

پارامتر	مقدار
فیلد	`nodeId`
برچسب	**«استقرار روی»، «پنل محلی»**
پیش‌فرض	خالی (پنل محلی)

انتخاب اینکه inbound فیزیکی روی کجا کار می‌کند: روی پنل محلی یا روی یکی از گره‌های ثبت‌شده. ویژگی پیاده‌سازی: `nodeId = 0` به `nil` نرمال می‌شود، زیرا `0` یک شناسه معتبر گره نیست بلکه آرتیفکت پیوند فرم است؛ `0` / `nil` به‌معنای پنل محلی است. هنگام ذخیره inbound روی یک گره آفلاین، ممکن است پیام «تغییر هنگام اتصال مجدد گره همگام‌سازی می‌شود» نمایش داده شود. از نسخه 3.4.2 این پیام فقط زمانی نشان داده می‌شود که گره واقعاً آفلاین یا خاموش باشد (قبلاً ممکن بود هنگام ذخیره روی گره آنلاین نیز ظاهر شود).

استراتژی آدرس برای لینک‌ها (Share address strategy)

پارامتر	مقدار
فیلد	استراتژی + (اختیاری) آدرس سفارشی
برچسب	**«استراتژی آدرس برای لینک‌ها»** (به انگلیسی *Share address strategy*)
پیش‌فرض	**«آدرس گوش‌دادن inbound»** (*Inbound listen*)

فهرست بازشو تعیین می‌کند که کدام آدرس در **لینک‌های اشتراک‌گذاری صادرشده و کدهای QR** این inbound درج شود. مقادیر:

مقدار	برچسب	چه چیزی درج می‌شود
`node`	**«آدرس گره»** (*Node address*)	آدرس گره‌ای که inbound روی آن کار می‌کند
`listen`	**«آدرس گوش‌دادن inbound»** (*Inbound listen*)	آدرس گوش‌دادن خود inbound
`custom`	**«سفارشی»** (*Custom*)	آدرس خودتان از فیلد **«آدرس سفارشی برای لینک‌ها»** (*Custom share address*)

با انتخاب **«سفارشی»**، فیلد **«آدرس سفارشی برای لینک‌ها»** ظاهر می‌شود؛ در آن میزبان یا IP را **بدون اسکیم و پورت** وارد می‌کنند (مقدار اعتبارسنجی می‌شود). گزینه **«آدرس گره»** فقط زمانی در فهرست نشان داده می‌شود که یک گره فعال وجود داشته باشد که این inbound بتواند روی آن کار کند؛ در غیر این صورت پنهان می‌شود و مقدار به **«آدرس گوش‌دادن inbound»** تبدیل می‌شود.

این استراتژی **فقط** روی لینک‌های مستقیم اشتراک‌گذاری و کدهای QR تأثیر می‌گذارد. روی صدور اشتراک (subscription) تأثیری **ندارد** — آنجا آدرس همچنان با منطق معمول پنل تعیین می‌شود.

4.2. Sniffing (اِسنیفینگ)

زبانه **«اسنیفینگ»** بلوک `sniffing` پیکربندی Xray را که به‌صورت JSON خام ذخیره می‌شود ویرایش می‌کند. Sniffing به Xray اجازه می‌دهد نام دامنه/پروتکل واقعی درون اتصال را برای مقاصد مسیریابی «استراق سمع» کند.

زیرفیلد	برچسب	کاربرد
`enabled`	(کلید زبانه)	sniffing را برای inbound فعال/غیرفعال می‌کند
`destOverride`	—	فهرست پروتکل‌هایی که آدرس مقصد برایشان استخراج می‌شود: `http`، `tls`، `quic`، `fakedns`
`metadataOnly`	**«فقط متادیتا»**	استفاده فقط از متادیتای اتصال، بدون خواندن محموله
`routeOnly`	**«فقط مسیریابی»**	اعمال نتیجه sniffing فقط برای مسیریابی، بدون بازنویسی آدرس مقصد
`domainsExcluded`	**«دامنه‌های مستثنی»**	دامنه‌هایی که از sniffing مستثنی می‌شوند
(IPهای مستثنی)	**«IPهای مستثنی»**	آدرس‌های IP که از sniffing مستثنی می‌شوند

- **`destOverride`** — مجموعه اسنیفرها: `http` (دامنه را از سرآیند Host در HTTP تشخیص می‌دهد)، `tls` (از SNI)، `quic` (از QUIC ClientHello)، `fakedns` (تطبیق با مخزن FakeDNS). معمولاً برای تشخیص دامنه `http` و `tls` فعال می‌شوند.

مثال بلوک `sniffing` (تشخیص دامنه بر اساس HTTP و TLS، استفاده از نتیجه فقط برای مسیریابی، بدون دست‌زدن به شبکه محلی):

```
{
  "enabled": true,
  "destOverride": ["http", "tls"],
  "routeOnly": true,
  "domainsExcluded": ["courier.push.apple.com"]
}
```

- **`metadataOnly`** — وقتی فعال باشد، Xray محتوای اولین بسته را نمی‌خواند و فقط به متادیتا تکیه می‌کند؛ برای آنکه پروتکل‌هایی که نمی‌توان داده‌هایشان را «استراق سمع» کرد مختل نشوند مفید است.
- **`routeOnly`** — نتیجه sniffing فقط توسط قواعد مسیریابی استفاده می‌شود؛ آدرس اتصال در outbound در این حالت به دامنه شناسایی‌شده بازنویسی نمی‌شود.

> نکته: پنل `sniffing` را به‌صورت یک بلوک JSON مات نگه می‌دارد و هنگام ذخیره چیزی به آن نمی‌افزاید — همه مقادیر پیش‌فرض این تیک‌ها در سمت برنامه کلاینت ساخته می‌شوند. این بلوک را می‌توان در حالت خام از طریق بخش «JSON ورودی» ویرایش کرد (به پایین مراجعه کنید).

4.3. Allocate (استراتژی توزیع پورت‌ها)

بلوک `allocate` در `streamSettings` نحوه توزیع پورت‌های گوش‌دادن توسط Xray را مدیریت می‌کند. این بخشی از پیکربندی Xray است؛ پنل آن را به‌عنوان بخشی از `streamSettings` JSON/ ورودی نگه می‌دارد و منتقل می‌کند. پارامترها (طبق اصطلاحات Xray-core):

زیرفیلد	کاربرد	مقادیر / پیش‌فرض
`strategy`	استراتژی تخصیص پورت‌ها	`always` — همیشه به پورت تعیین‌شده گوش بده (پیش‌فرض)؛ `random` — به‌صورت دوره‌ای پورت‌های گوش‌داده‌شده را درون یک بازه تغییر بده
`refresh`	بازه تعویض پورت‌ها (دقیقه) در حالت `random`	عدد صحیح دقیقه (مقدار توصیه‌شده 5؛ حداقل — 2)
`concurrency`	چند پورت همزمان باز نگه داشته شود در حالت `random`	عدد صحیح (پیش‌فرض 3؛ نه بیشتر از یک‌سوم پهنای بازه پورت‌ها)

`strategy: always` inbound را روی یک پورت نگه می‌دارد (حالت استاندارد). `strategy: random` برای سناریوهای anti-blocking لازم است، زمانی که inbound به‌صورت دوره‌ای روی یک بازه از پورت‌ها «می‌پرد»؛ در این حالت `refresh` و `concurrency` معنا پیدا می‌کنند. این مقادیر را فقط هنگام استفاده آگاهانه از حالت پورت‌های تصادفی تغییر دهید.

مثال بلوک `allocate` در `streamSettings` (حالت پورت‌های تصادفی: نگه‌داشتن 3 پورت باز، تعویض هر 5 دقیقه):

```
{
  "allocate": {
    "strategy": "random",
    "refresh": 5,
    "concurrency": 3
  }
}
```

برای آنکه این کار کند، `port` ورودی را به‌صورت بازه تعیین می‌کنند (مثلاً `20000-20100`).

4.4. External Proxy (پروکسی خارجی)

فیلد **«External Proxy»** به تنظیمات ساخت لینک‌های دعوت مربوط است و در `streamSettings` ورودی ذخیره می‌شود. این فیلد فهرستی از آدرس‌های خارجی جایگزین (میزبان/پورت، در صورت نیاز با TLS اجباری — **«TLS اجباری»**) را تعیین می‌کند که به‌جای `listen:port` واقعی ورودی در لینک‌های کلاینت درج می‌شوند.

زمانی استفاده می‌شود که کلاینت‌ها باید نه مستقیم به سرور، بلکه از طریق یک پروکسی/ریورس/CDN خارجی متصل شوند: در این صورت در لینک‌های مشترک، آدرس عمومی چنین فرانت‌اندی درج می‌شود. روی فرایند پذیرش اتصال‌ها توسط Xray تأثیری ندارد — این فقط «آرایش» لینک‌های تولیدشده است. فیلدهای مرتبط فرم: **«TLS اجباری»**، **«Fingerprint»**، برچسب هر رکورد.

4.5. Fallbacks (فال‌بک‌ها)

بخش **«فال‌بک‌ها»** قواعد هدایت اتصال‌هایی را که با هیچ کلاینت ورودی تطبیق نداشته‌اند تعیین می‌کند. برای master-inbound روی ترابری TLS (VLESS/Trojan TCP-TLS) در دسترس است. از طریق اندپوینت‌های `GET /:id/fallbacks` / `POST /:id/fallbacks` مدیریت می‌شود.

راهنمای بخش:

> «هنگامی که یک اتصال روی این inbound با هیچ کلاینتی تطبیق نکند، به جای دیگری هدایت می‌شود. یک inbound فرزند را در پایین انتخاب کنید تا فیلدهای مسیریابی (SNI / ALPN / Path / xver) به‌صورت خودکار از ترابری آن پر شوند، یا انتخاب را خالی بگذارید و Dest را مستقیم تعیین کنید (مثلاً 8080 یا 127.0.0.1:8080) تا به یک سرور خارجی مانند Nginx هدایت شود. هر inbound فرزند باید روی 127.0.0.1 با security=none گوش بدهد».

بخش فال‌بک‌ها فقط برای inbound از نوع VLESS/Trojan روی RAW (TCP) با امنیت TLS یا REALITY نشان داده می‌شود. inbound جدید با `security=none` آغاز می‌شود، بنابراین ممکن است بخش ابتدا به‌نظر غایب برسد. در این حالت (RAW/TCP، VLESS/Trojan، امنیت هنوز تنظیم نشده)، به‌جای بخش یک راهنمای داخلی نمایش داده می‌شود: فال‌بک‌ها پس از انتخاب TLS یا Reality در زبانه **«امنیت»** در دسترس می‌شوند.

فیلدهای سطر fallback

فیلد	پیش‌فرض	توضیح
(inbound فرزند)	—	انتخاب inbound فرزند (برچسب **«یک inbound انتخاب کنید»**). در صورت انتخاب، فیلدهای Name/Alpn/Path/Dest ممکن است به‌صورت خودکار از ترابری آن پر شوند
Name	خالی (= هر)	شرط تطبیق بر اساس نام (SNI/نام). برچسب «هر» — **«هر»**
Alpn	خالی	شرط تطبیق بر اساس ALPN
Path	خالی	شرط تطبیق بر اساس مسیر (برای ترابری‌های WS/HTTP در inbound فرزند)
Dest	خودکار	به کجا هدایت شود. متن جانگهدار **«خودکار (listen:پورت فرزند)»**. می‌توان پورت (`8080`) یا `host:port` (`127.0.0.1:8080`) را تعیین کرد
Xver	`0`	نسخه پروتکل PROXY (**«Xver»**): `0` — غیرفعال، `1` یا `2` — نسخه متناظر PROXY protocol
(ترتیب)	بر اساس موقعیت	ترتیب اعمال قواعد؛ با دکمه‌های **«بالا»**/**«پایین»** تعیین می‌شود

منطق ذخیره: کل فهرست فال‌بک‌های مستر به‌صورت اتمیک جایگزین می‌شود. سطری که نه inbound فرزند انتخاب‌شده (`childId <= 0`) و نه `Dest` تعیین‌شده دارد، **رد می‌شود**. اگر inbound فرزند انتخاب‌شده برابر شناسه خودِ مستر باشد، صفر می‌شود. هنگام تولید JSON نهایی: اگر `Dest` خالی باشد، از روی inbound فرزند به‌صورت `listen:port` محاسبه می‌شود، به‌طوری‌که `0.0.0.0` / `::` / `::0` با `127.0.0.1` جایگزین می‌شوند؛ فیلدهای خالی `path` / `alpn` / `name` به JSON خروجی نمی‌روند؛ `xver` فقط در صورتی افزوده می‌شود که بزرگ‌تر از 0 باشد.

مثال `settings.fallbacks` نهایی (ترافیک با `alpn=h2` به هدف WS از مسیر `/ws` می‌رود، بقیه — به Nginx محلی روی پورت 8080):

```
{
  "fallbacks": [
    { "alpn": "h2", "path": "/ws", "dest": "127.0.0.1:2001", "xver": 1 },
    { "dest": 8080 }
  ]
}
```

آخرین سطر بدون `path` / `alpn` / `name` قاعده «پیش‌فرض» است که بقیه را می‌گیرد.

دکمه‌ها و راهنماهای بخش fallbacks

- **«افزودن فال‌بک»** — افزودن سطر؛ **«هنوز فال‌بکی نیست»** — حالت خالی.
- **«افزودن سریع همه موارد مناسب» / «افزودن همه»** — برای هر inbound مناسبی که هنوز متصل نشده، یک سطر fallback می‌افزاید. نتیجه: «{n} فال‌بک افزوده شد» یا «هیچ inbound مناسب جدیدی نیست».
- **«پر کردن از فرزند»** — بازخوانی فیلدهای مسیریابی (SNI/ALPN/Path/xver) از ترابری inbound فرزند انتخاب‌شده؛ پس از اجرا — «از فرزند پر شد».
- **«ویرایش فیلدهای مسیریابی» / «پنهان‌کردن موارد پیشرفته»** — نمایش/پنهان‌کردن فیلدهای جزئی سطر.
- برچسب‌های **«مسیریابی می‌کند هنگامی که»** و **«پیش‌فرض — بقیه را می‌گیرد»** شرط فعال‌شدن هر سطر را توضیح می‌دهند.

پس از ذخیره فال‌بک‌ها، سرور Xray را راه‌اندازی مجدد می‌کند تا `settings.fallbacks` جدید اعمال شوند.

4.6. بازنشانی دوره‌ای ترافیک

بلوک **«بازنشانی ترافیک»** بازنشانی خودکار شمارنده‌های ترافیک inbound را بر اساس زمان‌بندی تنظیم می‌کند. توضیح:

> «بازنشانی خودکار شمارنده ترافیک در بازه‌های مشخص‌شده».

پارامتر	مقدار
فیلد	`trafficReset`
اعتبارسنجی	`omitempty,oneof=never hourly daily weekly monthly`
پیش‌فرض	`never`
فیلد همراه	`lastTrafficResetTime` — برچسب آخرین بازنشانی (برچسب **«آخرین بازنشانی»**)

فهرست بازشو:

مقدار	برچسب
`never`	«هرگز»
`hourly`	«ساعتی»
`daily`	«روزانه»
`weekly`	«هفتگی»
`monthly`	«ماهانه»

برای هر دوره یک کار cron ثبت شده که طبق زمان‌بندی متناظر اجرا می‌شود (`@hourly` ، `@daily` ، `@weekly` ، `@monthly`). این کار همه inboundهای دارای `trafficReset` تعیین‌شده را انتخاب می‌کند و برای هرکدام شمارنده‌های خود inbound (`up=0` ، `down=0`) و ترافیک همه کلاینت‌های آن را بازنشانی می‌کند. یعنی بازنشانی دوره‌ای هم inbound و هم کلاینت‌هایش را در بر می‌گیرد.

مثال مقدار فیلد. برای آنکه شمارنده‌ها در روز اول هر ماه صفر شوند، در فرم **«ماهانه»** را انتخاب می‌کنند که این‌گونه ذخیره می‌شود:

```
{ "trafficReset": "monthly" }
```

مقدار `never` (پیش‌فرض) بازنشانی خودکار را به‌طور کامل غیرفعال می‌کند.

4.7. JSON ورودی (پیشرفته)

بخش **«بخش‌های JSON ورودی»** دسترسی مستقیم به بلوک‌های خام JSON ورودی را فراهم می‌کند. توضیح:

> «JSON کامل ورودی و ویرایشگرهای جداگانه برای sniffing، settings و streamSettings».

ویرایشگرهای در دسترس:

زبانه	برچسب	چه چیزی را ویرایش می‌کند
همه	«شیء کامل ورودی با همه فیلدها در یک ویرایشگر»	کل شیء Inbound
تنظیمات	«پوشش بلوک settings از Xray»	فیلد `settings`
Sniffing	«پوشش بلوک sniffing از Xray»	فیلد `sniffing`
Stream	«پوشش بلوک stream از Xray»	فیلد `streamSettings`

این فیلدها به‌صورت اشیای JSON تودرتو سریالایز می‌شوند: بلوک‌های خالی به‌صورت `null` بازگردانده می‌شوند، و متنی که JSON معتبر نیست در یک رشته پیچیده می‌شود تا داده‌ها از دست نروند. خطاهای تجزیه هنگام ذخیره با پیشوند **«JSON پیشرفته»** نمایش داده می‌شوند.

پنجره مشاهده «JSON ورودی»، مانند پنجره وارد کردن inbound، از یک ویرایشگر کد تمام‌عیار با برجسته‌سازی نحو JSON استفاده می‌کند (به‌جای فیلد متنی معمولی): مشاهده پیکربندی — در حالت فقط‌خواندنی برجسته‌شده، و وارد کردن — در حالت قابل‌ویرایش، که خواندن و اصلاح را آسان‌تر می‌کند.

4.8. عملیات روی inbound: QR / Edit / Reset / Delete و آمار

در فهرست و در کارت inbound عملیات زیر در دسترس هستند (منوی **«منو»**):

آمار ترافیک

ترافیک تجمیعی inbound نمایش داده می‌شود: **«ارسال‌شده/دریافت‌شده»** (فیلدهای `down` / `up`)، **«کل ترافیک»**، **«کل اتصال‌ها»**. در کارت همچنین — **«ایجادشده»**، **«به‌روزشده»**، **«تاریخ پایان»**.

در فهرست inboundها ستونی به نام **Speed** با سرعت فعلی ترافیک هر inbound (آپلود/دانلود) وجود دارد که از افزایش شمارنده‌ها بین دو نمونه‌برداری محاسبه می‌شود؛ همان سرعت زنده در پنجره آمار inbound نیز نشان داده می‌شود. وقتی نمونه‌برداری بعدی افزایشی ندهد، مقدار سرعت بازنشانی می‌شود.

در خلاصه کلاینت‌ها در صفحه inboundها وضعیت بر اساس اولویت «مصرف‌شده/پایان‌یافته» تعیین می‌شود: کلاینت‌هایی که مدتشان به پایان رسیده یا ترافیکشان مصرف شده (و کار خودکار `enable` را از آن‌ها برداشته است) به وضعیت **«مصرف‌شده/پایان‌یافته»** (*Depleted/Ended*) تعلق دارند، نه وضعیت خاکستری **«غیرفعال»** (*Disabled*)، و دو بار شمرده نمی‌شوند. این دسته‌بندی با آنچه در کارت خود کلاینت نشان داده می‌شود مطابقت دارد و کلاینت‌های متصل به چند inbound را به‌درستی محاسبه می‌کند.

کد QR و کپی لینک‌ها

- **«جزئیات بیشتر»** — لینک‌های اتصال و اشتراک را باز می‌کند.
- کد QR کلاینت: راهنمای **«برای کپی روی کد QR کلیک کنید»**.
- **«کپی لینک»** (به انگلیسی *Copy URL*)، **«صادر کردن لینک‌ها»**. از نسخه 3.4.2 اقدام صفحه‌ای **«صادر کردن همه لینک‌ها»** (`GET /panel/api/inbounds/allLinks`)؛ پنجره «صادر کردن همه لینک‌های inbound»، فایل «All-Inbounds») لینک‌ها را از طریق موتور اشتراک می‌سازد — قالب ریمارک را برای هر کلاینت اعمال می‌کند و اندپوینت‌های Host مدیریت‌شده را ترجیح می‌دهد (پیش از این ریمارک ثابت `inbound-email` استفاده می‌شد).

Edit (ویرایش)

«ویرایش اتصال» — فرم ویرایش را باز می‌کند (`POST /update/:id`). هنگام به‌روزرسانی، سرویس سطر موجود را دوباره می‌خواند، فیلدهای تغییریافته را منتقل می‌کند، در صورت نیاز تگ را دوباره تولید می‌کند (اگر تگ قدیمی خودکار تولید شده بود) و رانتایم Xray را همگام‌سازی می‌کند. موفقیت — پیام **«اتصال با موفقیت به‌روز شد»**.

Reset Traffic (بازنشانی ترافیک)

«بازنشانی ترافیک» — شمارنده‌های `down` / `up` دقیقاً همین inbound را صفر می‌کند (`POST /:id/resetTraffic`، `up=0, down=0`) را قرار می‌دهد). تأیید:

> «بازنشانی ترافیک "{remark}"؟» / «شمارنده‌های ارسال/دریافت این اتصال را به 0 بازنشانی می‌کند».

بازنشانی ترافیک inbound شمارنده‌های کلاینت‌های آن را دست **نمی‌زند** (برای آن‌ها عملیات جداگانه «بازنشانی ترافیک کلاینت‌ها» وجود دارد). پس از بازنشانی، راه‌اندازی مجدد Xray آغاز می‌شود. موفقیت — پیام **«ترافیک ورودی بازنشانی شد»**. یک حالت گروهی نیز وجود دارد — **«بازنشانی ترافیک همه اتصال‌ها»** (`POST /resetAllTraffics`).

Delete (حذف)

«حذف اتصال» (`POST /del/:id`). تأیید:

> «حذف اتصال "{remark}"؟» / «اتصال و همه کلاینت‌های آن حذف می‌شوند. این عمل قابل بازگشت نیست».

حذف، inbound را از Xray در حال اجرا برمی‌دارد (در صورت نیاز با راه‌اندازی مجدد). موفقیت — پیام **«اتصال با موفقیت حذف شد»**. حذف گروهی — `POST /bulkDel` ، با گزارش به‌ازای هر عنصر و حداکثر یک راه‌اندازی مجدد Xray.

سایر عملیات با کلاینت‌های inbound

در منو همچنین در دسترس هستند: **«همانندسازی»** (نسخه‌ای از inbound با پورت جدید و فهرست خالی کلاینت‌ها)، **«حذف همه کلاینت‌ها»** (`POST /:id/delAllClients` — همه کلاینت‌ها را حذف می‌کند، خود inbound باقی می‌ماند)، **«حذف کلاینت‌های غیرفعال»**، **«پیوند/جداکردن کلاینت‌ها»**، **«وارد کردن»/«صادر کردن اتصال‌ها»** (`POST /import`). جزئیات عملیات کلاینتی به بخش کلاینت‌ها مربوط است.

5. پروتکل‌ها

هنگام ایجاد یک اتصال ورودی (inbound)، ابتدا **پروتکل** («Protocol») انتخاب می‌شود. پروتکل تعیین می‌کند که Xray-core چه روش احراز هویت و رمزنگاری ترافیک را برای این inbound به‌کار می‌گیرد، چه مجموعه‌ای از فیلدها در `settings` باید پر شود، و کدام انتقال‌ها (`network`) و انواع امنیت (TLS / REALITY) برای آن در دسترس هستند.

فیلد پروتکل یک‌بار هنگام ایجاد inbound تنظیم می‌شود و **هنگام ویرایش قابل تغییر نیست** (در فرم ویرایش، لیست کشویی قفل است). برای تغییر پروتکل، باید یک inbound جدید ایجاد کنید.

5.1. فهرست پروتکل‌های پشتیبانی‌شده

سرور مجموعه زیر را به عنوان مقادیر مجاز فیلد `Protocol` می‌پذیرد:

```
oneof=vmess vless trojan shadowsocks wireguard hysteria http mixed tunnel tun mtproto
```

> از نسخه **3.3.0** به بعد، مقدار `mtproto` (پروکسی تلگرام) به فهرست اضافه شده است.

مقدار در تنظیمات	کاربرد	مدل کلاینت
`vless`	پروتکل پروکسی اصلی (پیش‌فرض هنگام ایجاد inbound)	کلاینت‌ها با UUID، پشتیبانی از flow و رمزنگاری پست‌کوانتومی
`vmess`	پروتکل پروکسی کلاسیک Xray	کلاینت‌ها با UUID و پارامتر `security`
`trojan`	پروکسی که خود را به‌جای HTTPS معمولی جا می‌زند	کلاینت‌ها با رمز عبور
`shadowsocks`	پروکسی Shadowsocks (از جمله 2022-blake3 / SIP022)	یک یا چند کاربر (2022)
`wireguard`	Inbound WireGuard	Peerها (نه کلاینت‌ها)
`hysteria`	Inbound Hysteria (به‌طور پیش‌فرض نسخه 2)	کلاینت‌ها با توکن `auth`
`http`	پروکسی HTTP کلاسیک (forward proxy)	حساب‌های user/pass، بدون شمارش ترافیک
`mixed`	پروکسی ترکیبی SOCKS + HTTP	حساب‌های user/pass
`tunnel`	فوروارد شفاف (`dokodemo-door` در xray)	بدون کلاینت
`tun`	رابط TUN (فقط رندر موارد موجود)	بدون کلاینت
`mtproto`	پروکسی تلگرام (MTProto)، اضافه‌شده در 3.3.0؛ توسط پروسه جداگانه `mtg` مدیریت می‌شود، نه Xray	بدون کلاینت (دسترسی از طریق secret)

> توضیح درباره `tun`: این مقدار برای سازگاری و **نمایش** inboundهای قبلاً ذخیره‌شده در فهرست باقی مانده است، اما در نسخه فعلی backend ایجاد آن توصیه نمی‌شود — پشتیبانی منسوخ شده است. ایجاد inboundهای جدید از این نوع معنایی ندارد.

> توضیح درباره Hysteria 2: پروتکل جداگانه‌ای به نام «hysteria2» وجود ندارد. این پروتکل `hysteria` با فیلد `streamSettings.version = 2` است. طرح لینک `hysteria2://` هنگام تولید share-link به‌طور خودکار انتخاب می‌شود، وقتی که نسخه stream برابر 2 باشد.

همه پروتکل‌ها از توزیع روی گره‌ها (nodes) پشتیبانی نمی‌کنند. فقط موارد زیر را می‌توان روی گره‌ها استقرار داد: `vless`، `vmess`، `trojan`، `shadowsocks`، `hysteria`، `wireguard`. پروتکل‌های `http`، `mixed`، `tunnel`، `tun`، `mtproto` فقط روی پنل محلی کار می‌کنند.

5.2. کدام پروتکل‌ها از TLS / REALITY / انتقال پشتیبانی می‌کنند

امکان فعال‌سازی لایه امنیتی یا انتقال خاص به پروتکل و شبکه انتخابی (`streamSettings.network`) بستگی دارد:

قابلیت	در دسترس برای پروتکل‌ها	شبکه‌های مجاز (network)
TLS	`vmess`، `vless`، `trojan`، `shadowsocks` (و همیشه برای `hysteria`)	`tcp`، `ws`، `http`، `grpc`، `httpupgrade`، `xhttp`
REALITY	`vless`، `trojan`	`tcp`، `http`، `grpc`، `xhttp`
flow (xtls-rprx-vision)	فقط `vless`	فقط `tcp`، با `security = tls` یا `reality`
Stream / انتقال (تب «جریان»)	`vmess`، `vless`، `trojan`، `shadowsocks`، `hysteria`	—

برای پروتکل‌های `http`، `mixed`، `tunnel`، `tun`، `wireguard` تب انتقال در دسترس نیست — این‌ها تنظیمات stream در Xray ندارند.

5.3. VLESS

کاربرد: پروتکل پروکسی مدرن اصلی. از XTLS-Vision (`flow`)، REALITY و همچنین رمزنگاری پست‌کوانتومی در سطح خود VLESS (فیلدهای `decryption` / `encryption`) پشتیبانی می‌کند. به‌طور پیش‌فرض برای inboundهای جدید استفاده می‌شود.

فیلدهای بلوک `settings` :

فیلد	مقدار پیش‌فرض	توضیح
`clients`	`[]`	فهرست کلاینت‌ها. هر کلاینت دارای: `id` (UUID)، `email` (الزامی)، `flow`، محدودیت‌ها (`limitIp`، `totalGB`، `expiryTime`)، `enable`، `tgId`، `subId`، `comment`، `reset`
`decryption`	`none`	پارامتر رمزگشایی در سمت سرور. برچسب در UI: «Расшифрование» (انگلیسی «Decryption»)
`encryption`	`none`	پارامتر رمزنگاری جفت (در لینک کلاینت قرار می‌گیرد). برچسب: «Шифрование» (انگلیسی «Encryption»)
`fallbacks`	`[]`	فهرست fallbackها (بخش مربوط به fallbackها را ببینید)؛ در دسترس است وقتی `network = tcp` و `security` = TLS یا REALITY باشد
`testseed`	(4 عدد: 900, 500, 900, 256)	«4 — Vision testseed» عدد صحیح مثبت برای XTLS-Vision padding. فقط برای کلاینت‌هایی با flow `xtls-rprx-vision` اعمال می‌شود، در غیر این صورت نادیده گرفته می‌شود

flow (`xtls-rprx-vision`)

`flow` روی **کلاینت** تنظیم می‌شود، نه روی inbound، و یکی از سه مقدار زیر را می‌پذیرد:

مقدار	معنا
`` (خالی)	بدون XTLS-flow (پیش‌فرض)
`xtls-rprx-vision`	XTLS-Vision — حالت توصیه‌شده برای VLESS روی TCP+TLS/REALITY
`xtls-rprx-vision-udp443`	همان Vision، اما با پردازش UDP/443 (QUIC)

فیلد `flow` فقط زمانی قابل انتخاب است که تمام شرایط برقرار باشند: پروتکل `vless` ، `network = tcp` و `security` برابر `tls` یا `reality` . فیلد **Vision testseed** در فرم فقط در همان شرایط نمایش داده می‌شود.

> استثنا برای XHTTP: در VLESS روی `network = xhttp` با احراز هویت پست‌کوانتومی VLESS فعال (`encryption` / `decryption` ، flow، vlessenc) `xtls-rprx-vision` نیز مجاز است — صرف‌نظر از لایه امنیتی، از جمله با REALITY. در این حالت پنل `xtls-rprx-vision` را به درستی در share-linkها و اشتراک‌ها (از جمله فرمت Clash/Mihomo) منتقل می‌کند، بنابراین کلاینت پیکربندی را با Vision دریافت می‌کند.

رمزگشایی / رمزنگاری (احراز هویت پست‌کوانتومی VLESS)

فیلدهای `decryption` و `encryption` احراز هویت در سطح خود VLESS هستند (جداگانه از TLS/REALITY انتقالی). به‌طور پیش‌فرض هر دو برابر `none` هستند. در فرم زیر این فیلدها بلوک **«تولید کلید»** قرار دارد — یک لیست کشویی برای انتخاب حالت و دکمه **«Generate»** (در کنار آن دکمه **«Clear»**). لیست کشویی شش گزینه دارد: **X25519 (native)**، **X25519 (xorpub)**، **X25519 (random)**، **ML-KEM-768 (native)**، **ML-KEM-768 (xorpub)**، **ML-KEM-768 (random)** — یعنی دو نوع کلید (کلاسیک X25519 و پست‌کوانتومی ML-KEM-768)، هر کدام در سه حالت:

- **native** — جفت کلید پایه از نوع انتخاب‌شده؛
- **xorpub** — حالت مشتق‌شده با پردازش اضافی بخش عمومی؛
- **random** — حالت مشتق‌شده با مؤلفه تصادفی.

حالت مورد نظر را در لیست انتخاب کنید و **«Generate»** را فشار دهید: پنل **هر دو** فیلد (`decryption` و `encryption`) را با جفت مقادیر آماده برای آن حالت پر می‌کند. دکمه **«Clear»** هر دو فیلد را به `none` بازمی‌گرداند.

زیر بلوک، یک خط وضعیت **«انتخاب‌شده: ...»** نمایش داده می‌شود که از روی رشته تولیدشده هم نوع کلید (X25519 یا ML-KEM-768) و هم حالت (native / xorpub / random) را تشخیص داده و نمایش می‌دهد. فیلدهای خالی یا `none` به صورت «None» نشان داده می‌شوند.

از نظر فنی، دکمه‌ها با `GET /panel/api/server/getNewVlessEnc` ارتباط برقرار می‌کنند (تولید کلید از طریق `xray vlessenc`) و **هر دو** فیلد را با مقادیر جفت‌شده به شکل `mlkem768x25519plus.native.<rtt>.<role>` پر می‌کنند (مثلاً `decryption = mlkem768x25519plus.native.600s.server-x25519`، `encryption = mlkem768x25519plus.native.0rtt.client-x25519`). پارامتر `decryption` در سرور می‌ماند و `encryption` در لینک کلاینت قرار می‌گیرد.

> مهم: هنگام تولید پیکربندی inbound برای Xray، پنل موارد اضافی را حذف می‌کند: اگر `encryption` (که مربوط به سمت کلاینت است) در `settings` باقی بماند، از پیکربندی سرور **حذف** می‌شود. فقط `decryption` در سرور باقی می‌ماند.

چه زمانی VLESS را انتخاب کنید: این گزینه پیش‌فرض توصیه‌شده برای inbound جدید است، به‌ویژه همراه با REALITY (بدون گواهی خودی) یا با TLS + XTLS-Vision.

مثال: بلوک `settings` یک VLESS-inbound با یک کلاینت و XTLS-Vision. فیلد `flow` روی کلاینت تنظیم شده، `decryption` روی سرور باقی می‌ماند:

```
{
  "clients": [
    {
      "id": "d342d11e-d424-4583-b36e-524ab1f0afa4",
      "email": "user1",
      "flow": "xtls-rprx-vision",
      "limitIp": 2,
      "totalGB": 0,
      "expiryTime": 0,
      "enable": true
    }
  ],
  "decryption": "none"
}
```

برای ترکیب REALITY، بلوک متناظر `streamSettings` (تب «Transport» → Security: REALITY» به این شکل است:

```
{
  "network": "tcp",
  "security": "reality",
  "realitySettings": {
    "dest": "www.microsoft.com:443",
    "serverNames": ["www.microsoft.com"],
    "privateKey": "<приватный ключ X25519>",
    "shortIds": ["", "6ba85179e30d4fc2"]
  }
}
```

5.4. VMess

کاربرد: پروتکل پروکسی کلاسیک Xray. احراز هویت از طریق UUID؛ روش رمزنگاری محموله (`security`) روی کلاینت پیکربندی می‌شود.

فیلدهای بلوک `settings` :

فیلد	مقدار پیش‌فرض	توضیح
`clients`	`[]`	فهرست کلاینت‌ها

برای هر کلاینت VMess (علاوه بر فیلدهای مشترک `email` ، محدودیت‌ها، `enable` ، `tgId` ، `subId` ، `comment` ، `reset`):

فیلد کلاینت	مقدار پیش‌فرض	توضیح
`id`	–	UUID کلاینت

فیلد کلاینت	مقدار پیش‌فرض	توضیح
`security`	`auto`	روش رمزنگاری محموله VMess. مقادیر مجاز: `aes-128-gcm` ، `chacha20-poly1305` ، `auto` ، `none` ، `zero`

مقدار `security` :

- `auto` — Xray خودش رمز را بر اساس پلتفرم انتخاب می‌کند (توصیه‌شده)؛
- `aes-128-gcm` ، `chacha20-poly1305` — رمزهای AEAD ثابت؛
- `none` — بدون رمزنگاری محموله (فقط روی TLS معنا دارد)؛
- `zero` — بدون رمزنگاری و بدون احراز هویت محموله.

> سازگاری با نسخه‌های قدیمی: رکوردهای قدیمی ممکن است `security: ""` داشته باشند — هنگام خواندن، رشته خالی به `auto` تبدیل می‌شود. هنگام تولید پیکربندی سرور، فیلد `security` در کلاینت‌های VMess از `settings` **حذف** می‌شود، چون برای inbound لازم نیست.

چه زمانی VMess را انتخاب کنید: برای سازگاری با کلاینت‌های قدیمی یا پیکربندی‌های موجود. برای استقرارهای جدید، VLESS معمولاً ترجیح دارد.

5.5. Trojan

کاربرد: پروکسی که ترافیک HTTPS معمولی را تقلید می‌کند. احراز هویت با رمز عبور. مانند VLESS، از fallbackها و (با `network = tcp`) REALITY/TLS پشتیبانی می‌کند.

فیلدهای بلوک `settings` :

فیلد	مقدار پیش‌فرض	توضیح
`clients`	`[]`	فهرست کلاینت‌ها
`fallbacks`	`[]`	فهرست fallbackها (در دسترس با `network = tcp` و TLS/REALITY)

فیلد کلیدی هر کلاینت Trojan:

فیلد کلاینت	مقدار پیش‌فرض	توضیح
`password`	—	رمز عبور کلاینت (الزامی، حداقل 1 کاراکتر)
`email`	—	شناسه یکتای کلاینت

سایر فیلدهای کلاینت مشترک هستند (`limitIp` ، `totalGB` ، `expiryTime` ، `enable` ، `tgId` ، `subId` ، `comment` ، `reset`).

چه زمانی Trojan را انتخاب کنید: وقتی به پنهان‌سازی به عنوان HTTPS روی پورت 443 نیاز دارید، از جمله با fallbackها به وب‌سرور (Nginx) برای اتصال‌های ناخواسته.

مثال: بلوک `settings` Trojan با fallback به وب‌سرور محلی. اتصال‌های ناخواسته (بدون رمز معتبر) به Nginx که روی `127.0.0.1:8080` گوش می‌دهد، هدایت می‌شوند:

```
{
  "clients": [
    { "password": "S3cret-Pass-1", "email": "user1" }
  ],
  "fallbacks": [
    { "dest": "127.0.0.1:8080" }
  ]
}
```

برای fallback به `network = tcp` و Security = TLS یا REALITY نیاز است؛ در غیر این صورت فیلد fallbacks در دسترس نیست.

5.6. Shadowsocks

کاربرد: پروکسی سبک Shadowsocks. هم از رمزهای AEAD قدیمی و هم از روش‌های جدید SIP022 (`2022-blake3-*`) پشتیبانی می‌کند. می‌تواند در حالت تک‌کاربره یا چندکاربره کار کند.

فیلدهای بلوک `settings` :

فیلد	مقدار پیش‌فرض	توضیح
`method`	`2022-blake3-aes-256-gcm`	روش رمزنگاری inbound. برچسب در UI: «Метод шифрования» (انگلیسی «Encryption method»)
`password`	``	رمز عبور inbound (برای روش‌های 2022 به‌طور خودکار متناسب با روش انتخابی تولید می‌شود)
`network`	`tcp,udp`	انتقال. برچسب: «Сеть» (انگلیسی «Network»). گزینه‌ها: `tcp,udp` (TCP، UDP)، `tcp` ، `udp`
`clients`	`[]`	فهرست کلاینت‌ها
`ivCheck`	`false` (غیرفعال)	کلید «ivCheck» — محافظت در برابر استفاده مجدد از IV

روش‌های رمزنگاری (`method`)

مجموعه مجاز:

روش	دسته
`aes-256-gcm`	AEAD قدیمی
`chacha20-poly1305`	AEAD قدیمی
`chacha20-ietf-poly1305`	AEAD قدیمی
`xchacha20-ietf-poly1305`	AEAD قدیمی
`2022-blake3-aes-128-gcm`	SS 2022 (توصیه‌شده)
`2022-blake3-aes-256-gcm`	SS 2022 (پیش‌فرض)
`2022-blake3-chacha20-poly1305`	SS 2022، تک‌کاربره

منطق پنل در مورد روش‌ها:

- **روش‌های 2022** (`2022-blake3-*`) به عنوان «SS 2022» در نظر گرفته می‌شوند. روش `2022-blake3-chacha20-poly1305` — **تک‌کاربره** است (multi-user پشتیبانی نمی‌شود)؛ سایر روش‌های 2022 از چند کلاینت پشتیبانی می‌کنند. فیلد رمز عبور (با دکمه تولید که طول کلید را با روش تطبیق می‌دهد) در فرم فقط برای روش‌های 2022 نمایش داده می‌شود.
- **رمزهای قدیمی** (`aes-*` ، `chacha20-*`) به شکل کلاسیک «یک روش + یک رمز عبور» کار می‌کنند.

> نرمال‌سازی قبل از اجرای Xray: برای رمزهای قدیمی هر کلاینت باید `method` داشته باشد که با روش inbound مطابقت داشته باشد (در غیر این صورت Xray با «unsupported cipher method:» خطا می‌دهد). برای روش‌های 2022 برعکس — فیلد `method` کلاینت باید **خالی** باشد (در غیر این صورت Xray inbound را با «users must have empty method» رد می‌کند). پنل خودش داده‌ها را هنگام تغییر روش مرتب می‌کند.

> بازتولید کلیدهای کلاینت هنگام تغییر اندازه کلید: برای Shadowsocks-2022 هنگام تغییر روش رمزنگاری به روشی با اندازه کلید متفاوت (مثلاً بین `2022-blake3-aes-256-gcm` و `2022-blake3-aes-128-gcm`)، پنل به‌طور خودکار PSK کلاینت‌ها را هنگام ذخیره inbound برای طول جدید بازتولید می‌کند. در غیر این صورت کلیدهای قدیمی با طول قبلی می‌مانند و Xray آنها را رد می‌کرد. در نتیجه: کلاینت‌های متأثر باید اشتراک را دوباره دریافت کنند — لینک‌های قبلی دیگر وصل نمی‌شوند.

چه زمانی Shadowsocks را انتخاب کنید: برای استقرارهای ساده بدون پنهان‌سازی TLS؛ انتخاب مدرن — روش‌های `2022-blake3-*` هستند.

مثال: بلوک `settings` Shadowsocks برای روش blake3-2022 (حالت چندکاربره). inbound رمز عبور خودش را دارد (کلید base64 با طول مناسب)، هر کلاینت رمز عبور خودش را دارد، فیلد `method` کلاینت **خالی** است:

```
{
  "method": "2022-blake3-aes-256-gcm",
  "password": "d2hhdGV2ZXItMzItYnl0ZS1iYXNlNjQta2V5LWhlcmU=",
  "network": "tcp,udp",
  "clients": [
    {
      "email": "user1",
      "password": "Y2xpZW50LWtleS0zMi1ieXRlcy1iYXNlNjQtaGVyZQ==",
      "method": ""
    }
  ]
}
```

برای رمزهای legacy (مانند `aes-256-gcm`) — برعکس: یک رمز عبور برای inbound، و `method` کلاینت باید با روش inbound مطابقت داشته باشد.

5.7. Dokodemo-door / Tunnel (فوروارد شفاف)

کاربرد: فوروارد شفاف (در پنل — پروتکل `tunnel` ، که رفتار `dokodemo-door` را پیاده‌سازی می‌کند). ترافیک را می‌پذیرد و به آدرس/پورت مشخصی هدایت می‌کند، بدون احراز هویت و کلاینت.

فیلدهای بلوک `settings` :

فیلد	مقدار پیش‌فرض	توضیح
`rewriteAddress`	(نیست)	«بازنویسی آدرس» (انگلیسی «Rewrite address») — آدرس هدف که ترافیک به آن هدایت می‌شود
`rewritePort`	(نیست)	«بازنویسی پورت» (انگلیسی «Rewrite port») — پورت هدف (0–65535)
`allowedNetwork`	`tcp,udp`	«شبکه مجاز» (انگلیسی «Allowed network»). گزینه‌ها: `tcp,udp` ، `tcp` ، `udp`
`portMap`	`{}`	«نگاشت پورت» — نقشه پورت→پورت (<Record<string,string>)
`followRedirect`	`false` (غیرفعال)	«پیروی از redirect» (انگلیسی «Follow redirect») — استفاده از آدرس مقصد اصلی از اتصال رهگیری‌شده

> تب «Transport» برای Tunnel: در inbound از این نوع، تب **«Transport»** با محدودیت فقط تنظیم `sockopt` در دسترس است — این برای حالت **TProxy** (پروکسی شفاف/redirect از طریق `sockopt.tproxy`) کافی است. لیست کشویی انتخاب انتقال (`network`) و تب «Security» برای Tunnel پنهان هستند، چون TLS/REALITY توسط این نوع پشتیبانی نمی‌شود.

چه زمانی انتخاب کنید: برای پروکسی شفاف/هدایت پورت به سرویس‌های داخلی.

فیلد «بازنویسی پورت» (`rewritePort`) را می‌توان خالی گذاشت: هنگام پاک کردن، مقدار فقط از تنظیمات inbound حذف می‌شود و خطای ذخیره‌سازی ایجاد نمی‌کند. (قبلاً پاک کردن این فیلد باعث خطای اعتبارسنجی `settings.rewritePort` می‌شد و ذخیره‌سازی را مسدود می‌کرد، از جمله از طریق تب JSON)

5.8. SOCKS / HTTP (پروتکل `mixed`)

در این نسخه پروتکل جداگانه‌ای به نام `socks` وجود ندارد — SOCKS و HTTP-proxy در پروتکل **`mixed`** (SOCKS + HTTP ترکیبی) ادغام شده‌اند. علاوه بر این، یک `http` -proxy خالص جداگانه وجود دارد.

5.8.1. Mixed (SOCKS + HTTP)

فیلدهای بلوک `settings` :

فیلد	مقدار پیش‌فرض	توضیح
`auth`	`password`	«Auth» — حالت احراز هویت. گزینه‌ها: `password` (با نام کاربری/رمز عبور) یا `noauth` (بدون احراز هویت)
`accounts`	(اختیاری)	«حساب‌ها» — فهرست جفت‌های user/pass. با `auth = noauth` در تنظیمات نوشته نمی‌شود
`udp`	`false` (غیرفعال)	کلید «UDP» — پشتیبانی از UDP از طریق SOCKS
`ip`	`127.0.0.1`	«UDP IP» — آدرس محلی برای اتصال‌های UDP. این فیلد فقط هنگام فعال بودن `udp` نمایش داده می‌شود

حساب‌ها با دکمه «افزودن» اضافه می‌شوند؛ هنگام افزودن، نام کاربری تصادفی (8 کاراکتر) و رمز عبور تصادفی (12 کاراکتر) تولید می‌شود که قابل ویرایش هستند.

5.8.2. HTTP (پروکسی خالص)

کاربرد: forward-proxy HTTP کلاسیک. در سطح Xray کلاینت‌ها را به عنوان «مشتری صورت‌حساب» ردیابی نمی‌کند (بدون email/ محدودیت) — فقط فهرست حساب‌ها وجود دارد.

فیلدهای بلوک `settings` :

فیلد	مقدار پیش‌فرض	توضیح
`accounts`	`[]`	«حساب‌ها» — فهرست جفت‌های user/pass (هر دو فیلد الزامی هستند)
`allowTransparent`	`false` (غیرفعال)	«اجازه شفاف» (انگلیسی «Allow transparent») — ارسال درخواست‌ها با هدر اصلی Host

چه زمانی SOCKS/HTTP را انتخاب کنید: برای دسترسی پروکسی محلی یا سرویسی بدون پنهان‌سازی پیچیده. `mixed` مزیت دارد چون یک پورت هم کلاینت‌های SOCKS و هم HTTP را سرویس می‌دهد.

5.9. WireGuard (inbound)

کاربرد: inbound WireGuard — یک تونل VPN WireGuard، نه یک پروکسی با پنهان‌سازی انتقال و TLS/REALITY برای آن قابل اعمال نیستند.

از نسخه 3.4.2 WireGuard به مدل چندکلاینتی منتقل شده است (مانند VLESS/VMess/Trojan/Shadowsocks/Hysteria). خود inbound فقط بخش سروری را نگه می‌دارد؛ دستگاه‌های peer اکنون به‌عنوان **کلاینت‌های** معمولی افزوده می‌شوند (به بخش 8 مراجعه کنید) — با تخصیص خودکار آدرس در تونل، پشتیبانی از اشتراک‌ها، محدودیت‌های ترافیک/مدت و گروه‌ها. لیست درون‌خطی قبلی «Peerها / افزودن peer» از فرم inbound حذف شده است: inbound جدید بدون peer ایجاد می‌شود، و آدرس هر کلاینت به‌طور خودکار تخصیص می‌یابد.

فیلدهای inbound (بلوک `settings`):

فیلد	مقدار پیش‌فرض	توضیح
`secretKey`	—	کلید خصوصی سرور (الزامی). در کنار آن دکمه تولید قرار دارد؛ کلید عمومی (`Public Key`) به‌طور خودکار از آن استخراج می‌شود (فقط‌خواندنی؛ `pubKey` که جداگانه ذخیره می‌شد حذف شده است)
`mtu`	`1420`	MTU رابط
`dns`	`1.1.1.1, 1.0.0.1`	**جدید در 3.4.2.** DNS که در خط `DNS =` فایل `.conf` کلاینت نوشته می‌شود
`noKernelTun`	`false` (غیرفعال)	«TUN بدون userspace-TUN — «kernel به جای kernel
`domainStrategy`	(اختیاری)	«Domain Strategy»: `ForceIP` ، `ForceIPv4` ، `ForceIPv4v6` ، `ForceIPv6` ، `ForceIPv6v4`

پارامترهای WireGuard کلاینت — تب **«اعتبارنامه‌ها»** در پنجره افزودن/ویرایش کلاینت (وقتی inbound متصل WireGuard باشد نمایش داده می‌شوند):

فیلد	توضیح
«کلید خصوصی WireGuard»	کلید خصوصی کلاینت (قابل ویرایش، دارای دکمه «تولید مجدد»)؛ هنگام ورود، کلید عمومی به‌طور خودکار از آن استخراج می‌شود

فیلد	توضیح
«کلید عمومی WireGuard»	فقط‌خواندنی؛ از کلید خصوصی محاسبه می‌شود
«کلید مشترک WireGuard» (PSK)	کلید مشترک اضافی اختیاری
«IPهای مجاز WireGuard»	فقط‌خواندنی (در حالت ویرایش): آدرس تخصیص‌یافته به کلاینت در تونل، مثلاً `10.0.0.2/32`

آدرس در تونل به‌طور خودکار توسط سرور از زیرشبکه inbound تخصیص می‌یابد (به‌طور پیش‌فرض `10.0.0.0/24` : سرور `.1` را می‌گیرد، کلاینت‌ها — از `.2` به بعد)؛ اگر کلاینت‌های موجود از `/24` دیگری استفاده کنند، آدرس‌های جدید در همان زیرشبکه تخصیص می‌یابند.

Domain Strategy و تب Transport

علاوه بر موارد فهرست‌شده، inbound WireGuard دارای فیلد **Domain Strategy** (استراتژی تفسیر دامنه: `ForceIP` ، `ForceIPv4` ، `ForceIPv4v6` ، `ForceIPv6` ، `ForceIPv6v4`) است. این فیلد اختیاری است و فقط در صورت تنظیم در تنظیمات نوشته می‌شود.

> فیلد **Workers** (`workers` ، تعداد تردهای کاری) از فرم‌های WireGuard (هم inbound و هم outbound) حذف شده است: از xray-core v26.6.22 به بعد موتور دیگر از آن استفاده نمی‌کند و به مکانیزم داخلی wireguard-go متکی است. تنظیمات قبلاً ذخیره‌شده بدون تغییر کار می‌کنند — هنگام تجزیه، فیلد فقط نادیده گرفته می‌شود و مهاجرت لازم نیست.

برای WireGuard تب **«Transport»** نیز در دسترس است — اما در نسخه محدودشده: در آن فقط `sockopt` و obfuscation **Finalmask** پیکربندی می‌شوند. لیست کشویی انتقال (`network`) پنهان است، چون WireGuard همیشه روی UDP گوش می‌دهد. در رکوردهای نویز (noise)، Finalmask در یک فیلد جداگانه **Rand Range** (محدوده بایت 0–255، با اعتبارسنجی) دارد، و به عنوان روش obfuscation برای WireGuard و Hysteria، **Salamander** در دسترس است.

پیکربندی و اشتراک‌گذاری کلاینت WireGuard

برای کلاینت WireGuard در پنجره‌های «اطلاعات»/QR یک **کارت پیکربندی تاشو** در دسترس است: `.conf` کامل (`[Interface]` با `PrivateKey` / `Address` / `DNS` / `MTU` و `[Peer]` با `PublicKey` / `PresharedKey` =/ `AllowedIPs` `0.0.0.0/0, ::/0` / `Endpoint` / `PersistentKeepalive`) با دکمه‌های **کپی**، **دانلود** و **QR**. لینکی به شکل `wireguard://` / `wg://` نیز پشتیبانی می‌شود که اپلیکیشن کلاینت آن را به یک `.conf` آماده تجزیه می‌کند.

چه زمانی WireGuard را انتخاب کنید: وقتی به تونل VPN WireGuard واقعی نیاز دارید، نه یک پروکسی با پنهان‌سازی.

5.10. Hysteria (به‌طور پیش‌فرض نسخه 2)

کاربرد: inbound Hysteria روی QUIC. پنل به‌طور پیش‌فرض با نسخه 2 کار می‌کند. هر کلاینت با توکن `auth` به جای UUID/رمز عبور احراز هویت می‌شود. TLS برای Hysteria همیشه در دسترس است (جدول قابلیت‌ها در 5.2 را ببینید).

فیلدهای بلوک `settings` :

فیلد	مقدار پیش‌فرض	توضیح
`version`	`2`	نسخه پروتکل (حداقل 1؛ پیش‌فرض پنل 2)
`clients`	`[]`	فهرست کلاینت‌ها

فیلد کلیدی هر کلاینت `auth` (توکن، الزامی) به علاوه فیلدهای مشترک (`email` ، محدودیت‌ها، `enable` ، `tgId` ، `subId` ، `comment` ، `reset`) است.

پارامترهای اضافی در `streamSettings.hysteriaSettings` تنظیم می‌شوند:

فیلد	مقدار / گزینه‌ها	توضیح
`version`	ثابت شده 2 (فیلد قفل شده)	«Version»
`udpIdleTimeout`	(عدد صحیح ⩾ 1، ثانیه)	«UDP idle timeout (c)» — تایم‌اوت بیکاری UDP
`masquerade`	به‌طور پیش‌فرض غیرفعال	«Masquerade» — جعل هویت به عنوان یک وب‌سرور معمولی برای درخواست‌های «ناخواسته»

هنگام فعال‌سازی `masquerade` انتخاب نوع (`type`) ممکن است:

- `` — پیش‌فرض (صفحه 404)؛
- `proxy` — پروکسی معکوس (فیلدهای «Upstream URL»، «بازنویسی Host»، «نادیده گرفتن TLS verify»)؛
- `file` — سرویس‌دهی دایرکتوری (فیلد «دایرکتوری»، مثلاً `/var/www/html`)؛
- `string` — پاسخ ثابت (فیلدهای «کد وضعیت»، «Body»، «هدرها»).

چه زمانی Hysteria را انتخاب کنید: وقتی به انتقال QUIC و پایداری روی کانال‌های ناپایدار/موبایل نیاز دارید؛ پنهان‌سازی مخفی بودن نقطه ورود را افزایش می‌دهد.

5.11. MTProto (پروکسی برای تلگرام)

> در نسخه **3.3.0** اضافه شده است. مقدار پروتکل — `mtproto` .

MTProto پروتکل پروکسی اختصاصی تلگرام است. در 3X-UI، چنین inbound **نه توسط Xray، بلکه توسط پروسه جداگانه `mtg`** مدیریت می‌شود که خود پنل آن را کنترل می‌کند. پنل به‌طور دوره‌ای inboundهای MTProto فعال را با پروسه‌های `mtg` در حال اجرا مقایسه می‌کند: موارد کم را راه‌اندازی می‌کند، موارد اضافه را متوقف می‌کند و شمارنده‌های ترافیک را از متریک‌های `mtg` جمع‌آوری می‌کند. بنابراین **حسابداری ترافیک** برای چنین inbound مانند پروتکل‌های معمولی کار می‌کند.

راهنمای رسمی در فرم:

> «MTProto توسط پروسه جداگانه mtg مدیریت می‌شود، نه Xray. تنظیمات انتقال و کلاینت‌ها در اینجا اعمال نمی‌شوند — لینک زیر را در تلگرام به اشتراک بگذارید.»

پیامدها:

- تب‌های **«Transport» (Stream Settings) و «کلاینت‌ها» برای این inbound اعمال نمی‌شوند** — دسترسی با یک secret تنظیم می‌شود، نه فهرست کلاینت‌ها.
- inbound MTProto **فقط روی پنل اصلی** اجرا می‌شود؛ روی گره‌های فرزند (nodes) مستقر نمی‌شود (inboundهایی با `NodeID` مشخص نادیده گرفته می‌شوند).
- تب **«Sniffing»** برای MTProto پنهان است — این پروتکل توسط پروسه `mtg` سرویس می‌دهد، نه Xray، بنابراین sniffing برای آن قابل اعمال نیست.

فیلدها. در `settings` inbound ذخیره می‌شوند:

فیلد در UI	کلید	توضیح
Remark	`remark`	برچسب inbound.
Listen IP	`listen`	IP برای گوش دادن (خالی = همه رابطها).
Port	`port`	پورت پروکسی.
Secret	`settings.secret`	Secret دسترسی در فرمت **FakeTLS**.
دامنه پوشش (FakeTLS)	`settings.fakeTlsDomain`	دامنه‌ای که پروکسی ترافیک HTTPS آن را جعل می‌کند.

فرمت secret (FakeTLS). پنل به‌طور خودکار secret را به شکل صحیح تبدیل می‌کند: نتیجه = `ee` + 32 کاراکتر hex + کد hex دامنه پوشش، یعنی `ee<hex32><hex(fakeTlsDomain)>`. پیشوند `ee` حالت FakeTLS را فعال می‌کند، و دامنه (مثلاً یک سایت شناخته‌شده) برای پنهان‌سازی ترافیک به عنوان HTTPS معمولی استفاده می‌شود. کافی است دامنه را وارد کنید — پنل بقیه را خودش می‌سازد.

Domain-fronting و گزینه‌های پیشرفته mtg

inbound MTProto دارای پارامترهای اضافی پروسه `mtg` است. فیلدهای **Domain fronting IP**، **Domain fronting port** و **Domain fronting PROXY protocol** مشخص می‌کنند که `mtg` ترافیک غیر-تلگرام را به کجا ارسال کند (مثلاً به سایت جعلی NGINX): اگر IP خالی باشد، از دامنه FakeTLS از طریق DNS استفاده می‌شود، پورت پیش‌فرض — `443` است. علاوه بر این **Accept PROXY protocol** (برای listener)، **IP preference** (`prefer-ipv6` / `prefer-ipv4` / `only-ipv6` / `only-ipv4`) و **Debug logging** در دسترس هستند. هر مقدار فقط در صورت تنظیم در فایل `mtg-<id>.toml` نوشته می‌شود.

مسیریابی ترافیک تلگرام از طریق Xray

کلید **«Route through Xray»** (به‌طور پیش‌فرض غیرفعال) و فیلد اختیاری **Outbound** امکان می‌دهند egress تلگرام را به مسیریاب Xray واگذار کنید. هنگام فعال‌سازی، پنل یک SOCKS bridge محلی با تگ همان inbound را در تنظیمات Xray جاسازی می‌کند، و `mtg` ترافیک تلگرام را از طریق آن ارسال می‌کند. پس از این، ترافیک را می‌توان با قوانین روی تب «Routing» مطابقت داد یا از طریق فیلد **Outbound** به outbound یا load balancer انتخابی هدایت کرد (اگر فیلد خالی باشد، قوانین مسیریابی تصمیم می‌گیرند).

نحوه توزیع برای کاربر. برای inbound MTProto، پنل یک لینک دعوت تولید می‌کند:

مثال: secret FakeTLS و لینک آماده. اگر دامنه پوشش `www.cloudflare.com` باشد، secret به این شکل ساخته می‌شود: `ee` + 32 کاراکتر hex + کد hex دامنه، مثلاً:

```
secret = ee1a2b3c4d5e6f70819293a4b5c6d7e8f7777772e636c6f7564666c6172652e636f6d
```

لینک دعوت آماده (این لینک و QR code را در تلگرام برای کاربر ارسال می‌کنید):

```
tg://proxy?
server=203.0.113.10&port=443&secret=ee1a2b3c4d5e6f70819293a4b5c6d7e8f7777772e636c6f75646
66c6172652e636f6d
```

```
tg://proxy?server=<адрес>&port=<порт>&secret=<секрет>
```

(معادل — `https://t.me/proxy?server=…&port=…&secret=…`). این لینک و QR code را باید برای کاربر تلگرام ارسال کنید — با باز کردن، پروکسی مستقیماً به اپلیکیشن اضافه می‌شود. لینک همچنین از طریق سرور اشتراک ارائه می‌شود.

چه زمانی استفاده کنید. روش استاندارد دور زدن مسدودسازی تلگرام؛ پنهان‌سازی FakeTLS (دامنه پوشش) ترافیک را شبیه یک بازدید معمولی از سایت مشخص می‌کند.

5.12. راهنمای سریع انتخاب پروتکل

- **VLESS** — انتخاب پیش‌فرض؛ بهترین گزینه با REALITY یا TLS + XTLS-Vision، از احراز هویت پست‌کوانتومی پشتیبانی می‌کند.
- **Trojan** — پنهان‌سازی به عنوان HTTPS با fallback به وب‌سرور.
- **VMess** — سازگاری با کلاینت‌های قدیمی.
- **Shadowsocks** — پروکسی ساده بدون TLS؛ انتخاب مدرن — روش‌های `2022-blake3-*`.
- **Hysteria** — QUIC، پایداری روی کانال‌های بد.
- **mixed / http** — پروکسی‌های SOCKS/HTTP سرویسی.
- **WireGuard** — تونل VPN کامل.
- **tunnel** — هدایت شفاف پورت.
- **MTProto** — پروکسی برای دور زدن مسدودسازی تلگرام (FakeTLS)؛ پروسه جداگانه `mtg`.

6. ترانسپورت (Stream Settings)

ترانسپورت (در رابط پنل — فیلد **«ترانسپورت»**، انگلیسی *Transmission*) شیوه‌ای را تعیین می‌کند که Xray-core داده‌ها را درون inbound منتقل می‌کند: چه پروتکل شبکه‌ای روی TLS/Reality استفاده می‌شود و ترافیک دقیقاً چگونه قاب‌بندی می‌گردد. این پارامترها در شیء `streamSettings` پیکربندی Xray ذخیره می‌شوند و در تب ترانسپورت در ویرایشگر inbound تنظیم می‌گردند. رمزنگاری (TLS / Reality) در بخش جداگانه‌ای بررسی می‌شود — اینجا فقط انتخاب شبکه و پارامترهای آن شرح داده شده است.

6.1. انتخاب شبکه انتقال

شبکه در فهرست کشویی **«ترانسپورت»** (`streamSettings.network`) انتخاب می‌شود. مقدار پیش‌فرض — `tcp` است (در فهرست به‌صورت **RAW** نمایش داده می‌شود). گزینه‌های زیر در دسترس هستند:

مقدار در فهرست	فیلد `network`	ترانسپورت
RAW	`tcp`	TCP معمولی (در نسخه‌های جدید Xray به RAW تغییر نام یافته)، به‌صورت اختیاری با مبهم‌سازی HTTP
mKCP	`kcp`	ترانسپورت قابل‌اعتماد UDP به نام mKCP
WebSocket	`ws`	WebSocket روی HTTP(S)
gRPC	`grpc`	تونل gRPC (HTTP/2)
HTTPUpgrade	`httpupgrade`	HTTP Upgrade
XHTTP	`xhttp`	XHTTP / SplitHTTP — ترانسپورت مدرن قابل‌مالتی‌پلکس

هنگام تغییر مقدار، پنل بلوک تنظیمات شبکه قبلی را پاک می‌کند و بلوک شبکه جدید را با مقادیر پیش‌فرض از شِمای آن پر می‌کند، بنابراین هر فیلد زیرفرم همواره مقدار اولیه معناداری دارد.

> **مهم.** در این بیلد پنل **ترانسپورت HTTP/2 (h2) در فهرست وجود ندارد** — از مجموعه شبکه‌ها حذف شده است؛ برای تونل دوطرفه شبیه HTTP/2 از gRPC و برای ترانسپورت مدرن نقاب‌دار HTTP از XHTTP استفاده می‌شود. ترانسپورت **Hysteria** (`hysteria`) از طریق این فهرست انتخاب نمی‌شود: این ترانسپورت به‌طور سفت‌وسخت به پروتکل Hysteria گره خورده و به‌صورت خودکار زمانی پدیدار می‌شود که خودِ inbound با پروتکل Hysteria ساخته شود (به بند 6.8 مراجعه کنید).

در ادامه هر شبکه و هر فیلد آن جداگانه بررسی شده است.

6.2. RAW / TCP (`tcpSettings`)

ترانسپورت پایه TCP. به‌طور پیش‌فرض ترافیک «همان‌گونه که هست» منتقل می‌شود؛ به‌صورت اختیاری می‌توان آن را به شکل تبادل معمولی HTTP/1.1 نقاب زد.

فیلد	مقدار پیش‌فرض	توضیح
Proxy Protocol (`acceptProxyProtocol`)	`false` (خاموش)	پذیرش هدر PROXY protocol از بالانسر/پراکسی بالادست
مبهم‌سازی HTTP (`header.type`)	`none` (خاموش)	نقاب‌زدن ترافیک به شکل HTTP/1.1 را فعال می‌کند

Proxy Protocol

کلید **«Proxy Protocol»** (`acceptProxyProtocol`). وقتی فعال باشد، Xray در اتصال ورودی منتظر هدر PROXY protocol است و IP واقعی کلاینت را از آن استخراج می‌کند. فقط زمانی فعال کنید که جلوی پنل یک پراکسی معکوس/بالانسر (مثلاً HAProxy یا nginx با `send-proxy`) قرار دارد که این هدر را اضافه می‌کند. به‌طور پیش‌فرض خاموش است.

مبهم‌سازی HTTP (camouflage)

کلید **«HTTP مبهم‌سازی»**. فیلد `header` را مدیریت می‌کند:

- **خاموش** ← `header.type = "none"` (روی سیم، فیلد `header` صرفاً وجود ندارد). TCP خالص.
- **روشن** ← `header.type = "http"` . ترافیک به شکل درخواست و پاسخ HTTP/1.1 قالب‌بندی می‌شود. هنگام فعال‌سازی، پنل بلافاصله زیرشیء‌های `request` و `response` را با مقادیر پیش‌فرض پر می‌کند.

هنگام فعال‌بودن مبهم‌سازی HTTP، فیلدهای تنظیم درخواست و پاسخ شبیه‌سازی‌شده پدیدار می‌شوند.

هدر درخواست (`header.request`):

فیلد	کلید	مقدار پیش‌فرض	توضیح
نسخه درخواست	`request.version`	`1.1`	نسخه HTTP در خط آغازین درخواست
متد درخواست	`request.method`	`GET`	متد HTTP درخواست شبیه‌سازی‌شده
مسیر درخواست	`request.path`	`/`	مسیر(ها). به‌صورت فهرست مقادیر جداشده با ویرگول وارد می‌شود؛ روی سیم این یک آرایه از رشته‌هاست. اگر خالی بماند، `/` جایگزین می‌شود
هدرهای درخواست	`request.headers`	`{}` (خالی)	جدول «نام/مقدار» هدرهای HTTP. به‌صورت نگاشت `نام → [مقادیر]` ذخیره می‌شود (یک نام می‌تواند چند مقدار داشته باشد)

هدر پاسخ (`header.response`):

فیلد	کلید	مقدار پیش‌فرض	توضیح
نسخه پاسخ	`response.version`	`1.1`	نسخه HTTP در خط آغازین پاسخ
وضعیت پاسخ	`response.status`	`200`	کد وضعیت HTTP پاسخ شبیه‌سازی‌شده
دلیل پاسخ	`response.reason`	`OK`	توصیف متنی وضعیت (reason-phrase)
هدرهای پاسخ	`response.headers`	`{}` (خالی)	جدول «نام/مقدار» هدرهای پاسخ (نگاشت `نام → [مقادیر]`)

فیلدهای هدر به‌صورت خطبه‌خط ویرایش می‌شوند — هر خط نام هدر (`نام`) و مقدار آن (`مقدار`) را تعیین می‌کند. این پارامترها فقط برای نقاب‌زدن ظاهر ترافیک استفاده می‌شوند؛ بر رمزنگاری تأثیری ندارند. مقادیر پیش‌فرض (`GET / HTTP/1.1` ، پاسخ `200 OK`) برای بیشتر سناریوها مناسب‌اند — تغییر آن‌ها فقط زمانی منطقی است که نیاز به شبیه‌سازی یک سایت/سرویس مشخص داشته باشید.

نمونه `streamSettings` برای RAW با مبهم‌سازی HTTP:

```
{
  "network": "tcp",
  "tcpSettings": {
    "acceptProxyProtocol": false,
    "header": {
      "type": "http",
      "request": {
        "version": "1.1",
        "method": "GET",
        "path": ["/"],
        "headers": {
          "Host": ["www.example.com"]
        }
      },
      "response": {
        "version": "1.1",
        "status": "200",
        "reason": "OK"
      }
    }
  }
}
```

توجه کنید: `path` روی سیم یک آرایه از رشته‌هاست، و هر هدر یک آرایه از مقادیر است (`Host → ["www.example.com"]`).

6.3. mKCP (`kcpSettings`)

mKCP — ترانسپورت قابل‌اعتماد روی UDP. در کانال‌هایی با از دست رفتن بسته و تأخیر بالا مفید است، اما ترافیک سرویس بیشتری ایجاد می‌کند. تمام مقادیر پیش‌فرض با مقادیر توصیه‌شده در xray-core یکسان هستند.

فیلد	کلید	پیش‌فرض	مجاز	توضیح
MTU	`mtu`	`1350`	576–1460	حداکثر اندازه بسته (بایت). در صورت بروز مشکلات تکه‌تکه‌شدن کاهش می‌دهند
TTI (میلی‌ثانیه)	`tti`	`20`	10–100	فاصله انتقال (ms). کمتر — تأخیر پایین‌تر اما هزینه سربار بالاتر
Uplink (مگابایت/ثانیه)	`uplinkCapacity`	`5`	0 ⩽	پهنای‌باند محاسباتی برای ارسال (مگابایت/ثانیه)
Downlink (مگابایت/ثانیه)	`downlinkCapacity`	`20`	0 ⩽	پهنای‌باند محاسباتی برای دریافت (مگابایت/ثانیه)
ضریب CWND	`cwndMultiplier`	`1`	1 ⩽	ضریب پنجره ازدحام (congestion window)
حداکثر پنجره ارسال	`maxSendingWindow`	`2097152`	0 ⩽	حداکثر اندازه پنجره ارسال

نکات درباره فیلدها:

- **Uplink / Downlink capacity** تعیین می‌کنند mKCP چقدر تهاجمی کانال را اشغال می‌کند. آن‌ها را متناسب با پهنای واقعی کانال تنظیم می‌کنند: مقادیر بیش‌ازحد بالا به ترافیک اضافی و مقادیر بیش‌ازحد پایین به کم‌استفاده‌ماندن کانال می‌انجامند.
- **TTI** مستقیماً بر معاوضه «تأخیر ⁉ سربار» اثر می‌گذارد: مقادیر کمتر تأخیر را کاهش می‌دهند اما حجم بسته‌های سرویس را افزایش می‌دهند.
- **MTU** اندازه یک بسته mKCP را محدود می‌کند؛ کاهش آن در کانال‌هایی که بسته‌های بزرگ UDP بریده یا گم می‌شوند کمک می‌کند.

> در این بیلد فیلد «seed» (رمز مبهم‌سازی mKCP) و فهرست کشویی **نوع هدر/مبهم‌سازی** (`none` ، `srtp` ، `utp` ، `wechat-video` ، `dtls` ، `wireguard`) در زیرفرم mKCP **به‌عنوان فیلدهای جداگانه وجود ندارند** — مبهم‌سازی لایه ترانسپورت به مکانیزم عمومی «FinalMask» (شامل حالت `mkcp-legacy`) منتقل شده که در بخش مربوطه شرح داده شده است. پارامتر «congestion» نیز به‌صورت یک چک‌باکس جداگانه ارائه نشده است؛ مدیریت ازدحام از طریق `cwndMultiplier` و `maxSendingWindow` تنظیم می‌شود.

6.4. WebSocket (`wsSettings`)

ترانسپورت WebSocket روی HTTP(S). به‌خوبی از میان CDN و پراکسی‌های معکوس عبور می‌کند و به شکل ترافیک وب معمولی نقاب می‌خورد.

فیلد	کلید	پیش‌فرض	توضیح
Proxy Protocol	`acceptProxyProtocol`	`false`	پذیرش هدر PROXY protocol از پراکسی بالادست (به بند 6.2 مراجعه کنید)
Host (هاست)	`host`	`""` (خالی)	مقدار هدر HTTP به نام `Host` . هنگام کار از طریق CDN/فرانتینگ دامنه مشخص می‌شود
Path (مسیر)	`path`	`/`	مسیر در خط درخواست دست‌دهی WebSocket
دوره heartbeat	`heartbeatPeriod`	`0`	فاصله ارسال قاب‌های heartbeat (بر حسب ثانیه). `0` heartbeat را خاموش می‌کند
هدرها	`headers`	`{}` (خالی)	هدرهای HTTP اضافی دست‌دهی. نگاشت «نام ← مقدار» (فقط مقادیر رشته‌ای، بدون آرایه)

نکات:

- **مسیر** باید روی سرور (inbound) و کلاینت یکسان باشد. غالباً پشت این مسیر در سمت وب‌سرور نقطه ورود نقاب زده می‌شود.
- **هاست** زمانی معنا دارد که inbound پشت CDN قرار دارد یا از فرانتینگ دامنه استفاده می‌شود؛ در غیر این صورت می‌توان آن را خالی گذاشت.
- **دوره heartbeat** اتصال را هنگام عبور از پراکسی/CDNهایی که نشست‌های غیرفعال را تهاجمی قطع می‌کنند «زنده» نگه می‌دارد. به‌طور پیش‌فرض (`0`) heartbeat خاموش است.
- برخلاف RAW، جدول هدرهای WebSocket از فرمت «تخت» `نام → مقدار` استفاده می‌کند (یک خط مقدار به‌ازای هر هدر).

نمونه `streamSettings` برای WebSocket پشت CDN:

```
{
  "network": "ws",
  "wsSettings": {
    "acceptProxyProtocol": false,
    "host": "cdn.example.com",
    "path": "/ray",
    "heartbeatPeriod": 0,
    "headers": {
      "User-Agent": "Mozilla/5.0"
    }
  }
}
```

مقادیر `host` و `path` باید با کلاینت یکسان باشند؛ برخلاف RAW، مقدار هدر در اینجا یک رشته معمولی است نه یک آرایه.

6.5. gRPC (`grpcSettings`)

«سبک‌ترین» ترانسپورت از نظر تعداد پارامترها. ترافیک را درون فراخوانی‌های gRPC (روی HTTP/2) تونل می‌کند؛ با CDNهایی که از gRPC پشتیبانی می‌کنند به‌خوبی سازگار است. مبهم‌سازی هدر ندارد.

فیلد	کلید	پیش‌فرض	توضیح
نام سرویس (`Service Name`)	`serviceName`	`""` (خالی)	نام سرویس gRPC (در عمل — «مسیر» تونل). باید روی سرور و کلاینت یکسان باشد
Authority	`authority`	`""` (خالی)	مقدار شبه‌هدر `:authority` (معادل `Host` برای HTTP/2). هنگام کار از طریق CDN/دامنه مشخص می‌شود
Multi Mode	`multiMode`	`false` (خاموش)	مالتی‌پلکس چند جریان موازی gRPC درون یک اتصال را فعال می‌کند

نکات:

- **Service Name** — شناسه اصلی کانال gRPC است؛ باید در هر دو طرف یکسان باشد. مقدار خالی مجاز است، اما معمولاً یک رشته غیرواضح برای نقاب‌زدن تنظیم می‌کنند.
- **Authority** بر این تأثیر می‌گذارد که چه `:authority` در قاب‌های HTTP/2 ارسال شود؛ در درجه اول هنگام پراکسی‌کردن از طریق CDN لازم است.
- **Multi Mode** اجازه می‌دهد چند جریان منطقی از میان یک اتصال فیزیکی واحد عبور کنند؛ برای افزایش کارایی فعال می‌شود، زمانی که هم سرور و هم کلاینت از آن پشتیبانی می‌کنند.

نمونه `streamSettings` برای gRPC:

```
{
  "network": "grpc",
  "grpcSettings": {
    "serviceName": "GunService",
    "authority": "grpc.example.com",
    "multiMode": false
  }
}
```

`serviceName` (در اینجا `GunService`) نقش «مسیر» تونل را بازی می‌کند و باید روی سرور و کلاینت یکسان باشد.

6.6. HTTPUpgrade (`httpupgradeSettings`)

ترانسپورت مبتنی بر مکانیزم HTTP `Upgrade` (مانند WebSocket، اما بدون خودِ پروتکل WebSocket). همچنین به‌خوبی از میان پراکسی و CDN عبور می‌کند. مجموعه فیلدها همان WebSocket است، اما **بدون** دوره heartbeat.

فیلد	کلید	پیش‌فرض	توضیح
Proxy Protocol	`acceptProxyProtocol`	`false`	پذیرش هدر PROXY protocol از پراکسی بالادست
Host (هاست)	`host`	`""` (خالی)	مقدار هدر HTTP به نام `Host`
Path (مسیر)	`path`	`/`	مسیر درخواست HTTP با هدر `Upgrade`
هدرها	`headers`	`{}` (خالی)	هدرهای HTTP اضافی نگاشت «تخت» `نام → مقدار` (مانند WebSocket)

کاربرد فیلدهای **هاست**، **مسیر** و **هدرها** با WebSocket (بند 6.4) یکسان است. heartbeat برای HTTPUpgrade پیش‌بینی نشده — این ویژگی خاص WebSocket است.

6.7. XHTTP / SplitHTTP (`xhttpSettings`)

XHTTP (همان SplitHTTP) — ترانسپورت مدرن و قابل‌مالتی‌پلکس HTTP در xray-core. جریان‌های صعودی و نزولی را به درخواست‌های HTTP جداگانه تقسیم می‌کند، که به‌خوبی برای CDN و محیط‌هایی با محدودیت مدت‌زمان اتصال مناسب است. فیلدها همگی به‌یک‌باره در ویرایشگر قابل‌مشاهده نیستند: بخشی از آن‌ها بسته به حالت انتخاب‌شده (`mode`) و کلیدهای فعال‌شده پدیدار می‌شوند.

فیلدهای پایه (همواره قابل‌مشاهده)

فیلد	کلید	پیش‌فرض	توضیح
Host (هاست)	`host`	`""` (خالی)	مقدار هدر HTTP به نام `Host`
Path (مسیر)	`path`	`/`	مسیر پایه درخواست‌های HTTP
حالت (`Mode`)	`mode`	`auto`	حالت انتقال (به پایین مراجعه کنید)
Server Max Header Bytes	`serverMaxHeaderBytes`	`0`	محدودیت اندازه هدرهای درخواست در سرور (بایت). `0` — مقدار پیش‌فرض xray-core
Padding Bytes	`xPaddingBytes`	`100-1000`	بازه پدینگ «پرکننده» تصادفی (بر حسب بایت، قالب `min-max`) برای دشوارکردن تحلیل اندازه‌ها
هدرها	`headers`	`{}` (خالی)	هدرهای HTTP اضافی نگاشت «تخت» `نام → مقدار`
متد HTTP در Uplink	`uplinkHTTPMethod`	`""` (Default = POST)	متد HTTP درخواست‌های صعودی. گزینه‌ها: خالی (پیش‌فرض POST)، `POST`، `PUT`، `GET` (آخری فقط در حالت `packet-up` در دسترس است)

فیلد	کلید	پیش‌فرض	توضیح
Padding Obfs Mode	`xPaddingObfsMode`	`false` (خاموش)	مبهم‌سازی پیشرفته پدینگ را فعال و فیلدهای اضافی را باز می‌کند (به پایین مراجعه کنید)
No SSE Header	`noSSEHeader`	`false` (خاموش)	هدر `Content-Type: text/event-stream` (SSE) را ارسال نکن. اگر مانع عبور از گره‌های میانی شود فعال می‌کنند

فیلد «حالت» (`mode`)

فهرست کشویی با مقادیر:

مقدار	توضیح
`auto`	انتخاب خودکار حالت (پیش‌فرض)
`packet-up`	جریان صعودی به درخواست‌های HTTP جداگانه (یک بسته در هر درخواست) تقسیم می‌شود
`stream-up`	جریان صعودی با یک درخواست استریمیِ طولانی منتقل می‌شود
`stream-one`	یک درخواست استریمی دوطرفه مشترک

انتخاب حالت تعیین می‌کند کدام فیلدهای اضافی قابل‌مشاهده می‌شوند.

فیلدهای قابل‌مشاهده فقط هنگام `mode = packet-up` :

فیلد	کلید	پیش‌فرض	توضیح
حداکثر بارگذاری بافرشده	`scMaxBufferedPosts`	`30`	حداکثر تعداد درخواست‌های POST جریان صعودی که همزمان بافر می‌شوند
حداکثر اندازه بارگذاری (بایت)	`scMaxEachPostBytes`	`1000000`	حداکثر اندازه یک درخواست POST صعودی (بایت)
Uplink Data Placement	`uplinkDataPlacement`	`""` (Default = body)	محل قرارگیری داده‌های جریان صعودی: `body` ، `header` ، `cookie` ، `query`
Uplink Data Key	`uplinkDataKey`	`""`	نام کلید/هدر برای داده‌های uplink. تنها زمانی پدیدار می‌شود که `uplinkDataPlacement` تعیین شده و برابر `body` نباشد

فیلد قابل‌مشاهده فقط هنگام `mode = stream-up` :

فیلد	کلید	پیش‌فرض	توضیح
Stream-Up Server	`scStreamUpServerSecs`	`20-80`	بازه زمان نگه‌داشتن اتصال استریمی سمت سرور (بر حسب ثانیه، قالب `min-max`)

فیلدهای مبهم‌سازی پدینگ (هنگام `xPaddingObfsMode = روشن` قابل‌مشاهده‌اند)

فیلد	کلید	پیش‌فرض	توضیح
Padding Key	`xPaddingKey`	`""` (placeholder `x_padding`)	نام کلید برای پدینگ

فیلد	کلید	پیش‌فرض	توضیح
Padding Header	`xPaddingHeader`	`""` (placeholder X-Padding)	نام هدر HTTP که پدینگ در آن منتقل می‌شود
Padding Placement	`xPaddingPlacement`	`""` (Default = queryInHeader)	محل قرارگیری پدینگ: `queryInHeader`، `header`، `cookie`، `query`
Padding Method	`xPaddingMethod`	`""` (Default = repeat-x)	روش تولید پدینگ: `repeat-x` یا `tokenish`

قرارگیری نشست و توالی (همواره قابل‌مشاهده)

فیلد	کلید	پیش‌فرض	توضیح
Session ID Placement	`sessionIDPlacement`	`""` (Default = path)	محل انتقال شناسه نشست: `path`، `header`، `cookie`، `query`
Session ID Key	`sessionIDKey`	`""` (placeholder `x_session`)	نام کلید نشست. تنها زمانی پدیدار می‌شود که `sessionIDPlacement` تعیین شده و برابر `path` نباشد
Session ID Table	`sessionIDTable`	`""` (placeholder `Base62`)	مجموعه نویسه‌ها برای تولید شناسه‌های نشست. می‌توان یک مقدار از پیش‌تعریف‌شده را از فهرست کشویِ تکمیل‌خودکار انتخاب کرد (`ALPHABET`، `Alphabet`، `BASE36`، `Base62`، `HEX`، `alphabet`، `base36`، `hex`، `number`) یا یک رشته ASCII دلخواه وارد کرد. خالی — مقدار پیش‌فرض xray-core
Session ID Length	`sessionIDLength`	`""` (خالی)	طول یا بازه (مثلاً `8–16`) شناسه‌های تولیدشده. تنها زمانی نمایش داده می‌شود که `Session ID Table` تعیین شده باشد؛ حداقل باید بزرگ‌تر از 0 باشد
Sequence Placement	`seqPlacement`	`""` (Default = path)	محل انتقال شماره ترتیب بسته: `path`، `header`، `cookie`، `query`
Sequence Key	`seqKey`	`""` (placeholder `x_seq`)	نام کلید توالی. تنها زمانی پدیدار می‌شود که `seqPlacement` تعیین شده و برابر `path` نباشد

فیلدهای نشست برای xray-core نسخه v26.6.22 تغییر نام یافته‌اند: پیش‌تر **Session Key / Session Placement** (`sessionPlacement` / `sessionKey`) نامیده می‌شدند — اکنون **Session ID Key / Session ID Placement** (`sessionIDPlacement` / `sessionIDKey`) هستند؛ هسته دیگر نام قدیمی را نمی‌فهمد. inboundهایی که پیش از به‌روزرسانی ذخیره شده‌اند به‌صورت خودکار به کلیدهای جدید مهاجرت می‌کنند — لازم نیست آن‌ها را دوباره ذخیره کنید.

توصیه‌ها:

- برای بیشتر نصب‌ها کافی است **حالت = `auto`** بماند، **مسیر/هاست** تعیین شود و (هنگام کار از طریق CDN) با کلاینت هماهنگ گردد.
- فیلدهای قرارگیری (`*Placement` / `*Key`) و مبهم‌سازی پدینگ فقط برای تنظیم دقیق متناسب با یک سناریوی مشخص ضدِDPI/CDN لازم‌اند؛ هنگام مقادیر خالی، مقادیر پیش‌فرض xray-core که در پرانتز آمده‌اند استفاده می‌شوند.
- پارامترهای مربوط به سمت کلاینت/outbound (مثلاً فواصل POST مجدد، اندازه چانک‌ها) در فرم inbound ارائه نمی‌شوند — سرورِ شنونده آن‌ها را نادیده می‌گیرد. در مقابل، مالتی‌پلکس XMUX در فرم inbound در دسترس است (به پایین مراجعه کنید).
- **پیش‌فرض‌های سرویس درج نمی‌شوند.** پنل دیگر مقادیر پیش‌فرض سرویس `scMaxEachPostBytes` و `scMinPostsIntervalMs` را در پیکربندی‌های XHTTP ثبت نمی‌کند — مقادیر داخلی xray-core اعمال می‌شوند. این کار یک

امضای ثابت DPI (لیترال `scMinPostsIntervalMs=30`) را حذف می‌کند که پیش‌تر می‌توانست باعث مسدودشدن ترافیک شود. برای inboundهای از پیش ذخیره‌شده، مقادیری که با پیش‌فرض‌های xray-core یکسان‌اند در لینک‌ها و اشتراک نمایش داده نمی‌شوند (لازم نیست inbound را دوباره ذخیره کنید)؛ مقادیری که دستی تنظیم شده‌اند حفظ می‌گردند.

نمونه `streamSettings` برای XHTTP (حالت `auto`):

```
{
  "network": "xhttp",
  "xhttpSettings": {
    "host": "xhttp.example.com",
    "path": "/yourpath",
    "mode": "auto",
    "xPaddingBytes": "100-1000"
  }
}
```

برای بیشتر نصب‌ها همین چهار فیلد کافی است؛ فیلدهای قرارگیری نشست/توالی و مبهم‌سازی پدینگ را خالی می‌گذارند — در این صورت مقادیر پیش‌فرض xray-core استفاده می‌شوند.

XMUX (مالتی‌پلکس اتصالات)

کلید **XMUX** (`enableXmux`) لایه مالتی‌پلکسی را فعال می‌کند که درخواست‌های موازی را روی یک استخر کوچک از اتصالات فیزیکی توزیع می‌کند. هنگام فعال‌سازی، شش فیلد قابل‌تنظیم باز می‌شوند (در `xhttpSettings.xmux` ذخیره می‌شوند):

فیلد	کلید	پیش‌فرض	توضیح
Max Concurrency	`maxConcurrency`	`16-32`	حداکثر درخواست‌های هم‌زمان روی یک اتصال (بازه `min-max`)
Max Connections	`maxConnections`	`6`	حداکثر اتصالات فیزیکی (`0` — بدون محدودیت). از نسخه 3.4.2 inboundهای جدید با مقدار `6` ایجاد می‌شوند؛ موارد قبلی مقدار خود را حفظ می‌کنند.
Max Reuse Times	`cMaxReuseTimes`	`""` (خالی)	تعداد دفعات استفاده مجدد از یک اتصال
Max Request Times	`hMaxRequestTimes`	`600-900`	حداکثر درخواست‌ها روی یک اتصال (بازه)
Max Reusable Secs	`hMaxReusableSecs`	`1800-3000`	مدتی که در طول آن اتصال برای استفاده مجدد مناسب است (ثانیه، بازه)
Keep Alive Period	`hKeepAlivePeriod`	`""` (خالی)	دوره keep-alive برای نگه‌داشتن اتصال

> **مهم.** نمی‌توان هم‌زمان **Max Connections** و **Max Concurrency** را تعیین کرد — xray-core چنین پیکربندی را رد می‌کند. به‌طور پیش‌فرض هنگام فعال‌کردن XMUX پنل `Max Concurrency = 16-32` را درج می‌کند؛ اگر **Max Connections** را تعیین کنید (مقدار بزرگ‌تر از `0`)، پنل مقدار پیش‌فرض `Max Concurrency` را برمی‌دارد تا از بروز تعارض جلوگیری کند.

6.8. ترانسپورت Hysteria (`hysteriaSettings`)

ترانسپورت **Hysteria** در فهرست «ترانسپورت» انتخاب نمی‌شود: به‌صورت خودکار زمانی فعال می‌گردد که inbound با پروتکل Hysteria ساخته شود، و برای پروتکل‌های دیگر پنهان است (هنگام خروج از پروتکل Hysteria، شبکه به‌اجبار به `tcp` بازمی‌گردد). پارامترها:

فیلد	کلید	پیش‌فرض	توضیح
نسخه	`version`	`2` (تثبیت‌شده، فیلد قفل است)	نسخه Hysteria. فقط Hysteria 2 پشتیبانی می‌شود
UDP idle timeout (ثانیه)	`udpIdleTimeout`	`60`	تایم‌اوت بیکاری نشست UDP (ثانیه). بازه مجاز — 2–600؛ xray-core هنگام راه‌اندازی مقادیر خارج از این بازه را رد می‌کند
Masquerade	`masquerade`	خاموش (وجود ندارد)	نقاب‌زدن شنونده به شکل سرور HTTP/3 هنگام پروب‌زدن را فعال می‌کند

هنگام فعال‌بودن **Masquerade** انتخاب نوع (`type`) و فیلدهای وابسته به آن پدیدار می‌شوند:

- **`""` — default (404 page)**: صفحه استاندارد 404 ارائه می‌شود (فیلد اضافی لازم نیست).
- **`proxy` (reverse proxy)**: پراکسی معکوس به سایت خارجی.
  - `url` (**Upstream URL**، placeholder `https://www.example.com`) — آدرس مقصد؛
  - `rewriteHost` (**بازنویسی Host**، پیش‌فرض `false`) — جایگزین‌کردن هدر `Host` ؛
  - `insecure` (**رد TLS verify**، پیش‌فرض `false`) — گواهی TLS آپ‌استریم را بررسی نکن.
- **`file` (serve directory)**: ارائه فایل‌ها از یک پوشه.
  - `dir` (**دایرکتوری**، placeholder `/var/www/html`).
- **`string` (fixed body)**: پاسخ HTTP ثابت.
  - `statusCode` (**کد وضعیت**، پیش‌فرض `0` ، بازه 0–599)؛
  - `content` (**Body**) — بدنه پاسخ؛
  - `headers` (**هدرها**) — نگاشت `نام → مقدار` .

Masquerade به inbound مبتنی بر Hysteria اجازه می‌دهد در برابر پروب‌های فعال مانند یک سرور HTTP/3 معمولی به نظر برسد، که پنهان‌کاری را افزایش می‌دهد. به‌طور پیش‌فرض نقاب‌زدن خاموش است.

نمونه `hysteriaSettings` با پراکسی معکوس (`masquerade` → `proxy`):

```
{
  "version": 2,
  "udpIdleTimeout": 60,
  "masquerade": {
    "type": "proxy",
    "url": "https://www.example.com",
    "rewriteHost": true,
    "insecure": false
  }
}
```

در اینجا هنگام پروب‌زدن، شنونده پاسخ را از `https://www.example.com` ارائه می‌دهد و به شکل یک سایت HTTP/3 معمولی نقاب می‌خورد.

6.9. پارامترهای همراه

علاوه بر انتخاب شبکه، در همان تب دو بلوک عمومی در دسترس‌اند که به ترانسپورت مشخصی وابسته نیستند (به‌تفصیل — در بخش‌های مربوطه):

- **External Proxy** (`externalProxy`) — فهرستی از آدرس‌ها/پورت‌های خارجی که به‌جای آدرس خودِ پنل در لینک‌های اشتراک جایگزین می‌شوند.
- **Sockopt** (`sockopt`) — گزینه‌های سطح‌پایین سوکت (mark، TCP Fast Open، استراتژی دامنه، پراکسی شفاف و غیره).

Real client IP (تشخیص IP واقعی پشت CDN/رله)

وقتی inbound پشت یک واسطه قرار دارد (CDN مانند Cloudflare، تونل/رله L4 یا پنل دیگر)، Xray آدرس واسطه را می‌بیند نه بازدیدکننده واقعی را. این آدرس وارد فهرست کلاینت‌های آنلاین می‌شود و محدودیت IP به‌ازای کلاینت بر اساس آن شمارش می‌گردد، که باعث می‌شود هر دو پشت پراکسی بی‌فایده شوند. برای بازیابی IP واقعی، در بخش **Sockopt** فرم inbound یک انتخاب پیش‌تنظیم **Real client IP** وجود دارد که تنظیمات `acceptProxyProtocol` و `trustedXForwardedFor` را یکجا می‌کند:

پیش‌تنظیم	چه می‌کند	چه زمانی به کار می‌رود
Off / direct	هر دو فیلد را پاک می‌کند.	inbound مستقیماً برای کلاینت‌ها در دسترس است
Cloudflare CDN	`sockopt.trustedXForwardedFor = ["CF-Connecting-IP"]` را تنظیم می‌کند.	WebSocket / HTTPUpgrade / XHTTP / gRPC پشت CDN کلودفلر (ابر نارنجی)
L4 relay / Spectrum (PROXY)	`acceptProxyProtocol = true` را فعال می‌کند.	تونل/رله L4 جلوی inbound یا Cloudflare **Spectrum**

پیش‌تنظیم‌ها متقابلاً ناسازگارند: انتخاب یکی فیلد دیگری را پاک می‌کند، بنابراین `trustedXForwardedFor` منسوخ IP بازیابی‌شده از طریق پروتکل PROXY را بازنویسی نمی‌کند. زیر پیش‌تنظیم، کلید «خام» **Proxy Protocol** و فهرست **Trusted X-Forwarded-For** همچنان دیده می‌شوند — پیش‌تنظیم فقط آن‌ها را به‌جای شما پر می‌کند و در صورت نیاز می‌توان آن‌ها را دستی ویرایش کرد. اگر پیش‌تنظیم انتخاب‌شده توسط ترانسپورت فعلی پشتیبانی نشود (مثلاً پروتکل PROXY روی mKCP)، فرم هشداری نشان می‌دهد. این فیلدها فقط به سمت سرور مربوطاند و **هرگز در اشتراک‌ها به کلاینت‌ها ارسال نمی‌شوند**.

> **فقط یکی را به کار ببرید.** `acceptProxyProtocol` IP واقعی را از هدر L4 پروتکل PROXY می‌خواند، و `trustedXForwardedFor` — از هدر HTTP درخواست؛ آمیختن دستی آن‌ها فقط زمانی منطقی است که زنجیره واسطه‌های شما این را الزامی کند.

- **FinalMask** (`finalmask`) — مکانیزم عمومی مبهم‌سازی لایه ترانسپورت (شامل مبهم‌سازی legacy برای mKCP)، که جایگزین فیلدهای جداگانه «header type»/«seed» درون زیرفرم‌های شبکه شده است.

7. امنیت اتصال: XTLS، TLS و REALITY

هر inbound که از انتقال داده از طریق جریان transport پشتیبانی می‌کند (Hysteria، Shadowsocks، Trojan، VLESS، VMess)، در ویرایشگر دارای تب **«امنیت»** است. در این تب نحوه رمزگذاری و مبهم‌سازی کانال transport تنظیم می‌شود. سه حالت در دسترس هستند که با دکمه‌های رادیویی جابجا می‌شوند:

حالت	برچسب در UI	زمان در دسترس بودن
`none`	**هیچ**	همیشه (به جز Hysteria که TLS برای آن اجباری است)
`tls`	**TLS**	برای VMess/VLESS/Trojan/Shadowsocks در شبکه‌های `tcp`، `ws`، `http`، `grpc`، `httpupgrade`، `xhttp`؛ برای Hysteria — همیشه
`reality`	**Reality**	فقط برای VLESS/Trojan در شبکه‌های `tcp`، `http`، `grpc`، `xhttp`

دکمه **هیچ** نمایش داده نمی‌شود اگر پروتکل Hysteria باشد (TLS برای آن اجباری است). دکمه **Reality** فقط در صورت ترکیب مجاز پروتکل و شبکه ظاهر می‌شود (جدول بالا را ببینید).

با تغییر حالت، پانل بلوک `streamSettings` را کاملاً بازسازی می‌کند: `tlsSettings` و `realitySettings` حالت قبلی حذف شده و مقادیر پیش‌فرض حالت انتخابی جایگزین می‌شوند. به‌طور خاص، با انتخاب **Reality**، پانل فوراً به‌صورت خودکار: یک جفت تصادفی `target` + `serverNames` (SNI) از فهرست داخلی دامنه‌های محبوب جایگزین می‌کند، `shortIds` تصادفی تولید می‌کند، و همچنین یک درخواست به سرور برای دریافت جفت کلید تازه (privateKey/publicKey) X25519 ارسال می‌کند.

7.1. تفاوت: TLS در مقابل XTLS در مقابل REALITY

- **TLS** — رمزگذاری کلاسیک transport بر اساس پروتکل TLS 1.2/1.3. سرور باید دارای گواهی معتبر باشد (دامنه اختصاصی + زنجیره). ترافیک مانند HTTPS معمولی به نظر می‌رسد، اما برای یک سانسورگر فعال، handshake TLS قابل تشخیص به دامنه شما است؛ در صورت مسدودسازی بر اساس SNI یا نبودن گواهی معتبر، اتصال مسدود می‌شود یا خطا می‌دهد.

- **XTLS (Vision)** — این یک حالت جداگانه در فهرست «امنیت» نیست، بلکه مکانیزم *flow* بر فراز TLS **یا** Reality است. از طریق فیلد **Flow** = `xtls-rprx-vision` (یا `xtls-rprx-vision-udp443`) در سمت کلاینت inbound فعال می‌شود. Vision برای VLESS در شبکه `tcp` با `security = tls` یا `security = reality` در دسترس است، همچنین برای VLESS بر فراز transport `xhttp` با فعال بودن رمزگذاری (ML-KEM / vlessenc) VLESS — در این حالت فیلد **Flow** را نیز می‌توان روی `xtls-rprx-vision` تنظیم کرد و مقدار به درستی در لینک `vless://` قرار می‌گیرد (`flow=xtls-rprx-vision`). Vision «رمزگذاری دوگانه» (TLS-in-TLS) را کاهش می‌دهد و بار مفید را مستقیماً پس از دست‌دهی ارسال می‌کند که سرعت انتقال را افزایش داده و مبهم‌سازی را بهبود می‌بخشد. به همین دلیل ترکیب **VLESS + Reality + Flow** **`xtls-rprx-vision`** به عنوان پیکربندی مدرن توصیه‌شده در نظر گرفته می‌شود.

> **بازیابی خودکار flow Vision.** اگر رمزگذاری برای VLESS/XHTTP-inbound (ML-KEM، فیلدهای decryption/encryption) پس از افزودن کلاینت‌ها فعال شود، inbound واجد شرایط flow می‌شود. در این وضعیت، پانل به‌طور خودکار `flow = xtls-rprx-vision` را برای کلاینت‌هایی که به آن نیاز دارند اما فیلد **Flow** آن‌ها خالی است بازیابی می‌کند. پیش از این در چنین سناریویی، Vision بی‌صدا از کانفیگ‌ها، لینک‌های اشتراک‌گذاری و اشتراک‌ها حذف می‌شد (به‌ویژه در inbound های گره‌ای مشهود بود). هیچ اقدام

> دستی لازم نیست: اصلاح به‌طور خودکار هنگام ذخیره inbound و یک‌بار هنگام به‌روزرسانی پانل اعمال می‌شود. رفتار محافظه‌کارانه است — پانل flow را اختراع نمی‌کند و مقدار صریح تعیین‌شده توسط کلاینت را بازنویسی نمی‌کند.

- **REALITY** — مکانیزم مبهم‌سازی بدون گواهی اختصاصی. سرور handshake TLS یک سایت خارجی واقعی (`target` / `serverNames`) را «قرض می‌گیرد»، بنابراین برای ناظر، اتصال از دسترسی به آن سایت غیرقابل تمایز است و اصلاً به گواهی نیاز نیست. احراز هویت بر اساس جفت کلید X25519 و مجموعه `shortIds` ساخته می‌شود. REALITY در برابر کاوش‌های فعال (`active probing`) و مسدودسازی بر اساس SNI مقاوم است، زیرا SNI به یک دامنه خارجی واقعی اشاره می‌کند. قیمت آن الزامات سخت‌تر پیکربندی است (یک `target` صحیح با پورت، همگام‌سازی کلیدها با کلاینت).

قانون کوتاه انتخاب:

- دامنه اختصاصی و گواهی معتبر دارید و ظاهر ساده HTTPS کافی است → **TLS** (در صورت امکان با Vision)؛
- دامنه/گواهی ندارید یا به حداکثر پنهان‌کاری از DPI نیاز دارید → **REALITY** (با Vision برای VLESS/TCP).

7.2. حالت «هیچ» (`none`)

Transport بدون wrapper TLS منتقل می‌شود: بلوک‌های `tlsSettings` و `realitySettings` از `streamSettings` حذف می‌شوند. این حالت فیلد اضافی ندارد. مناسب است وقتی:

- inbound فقط روی `127.0.0.1` گوش می‌دهد و به عنوان هدف fallback عمل می‌کند (طبق قانون پانل، inbound فرزند برای fallback باید روی `127.0.0.1` با `security=none` گوش دهد)؛
- رمزگذاری/مبهم‌سازی توسط یک لایه خارجی تأمین می‌شود (مثلاً پروکسی معکوس Nginx در جلوی پانل)؛
- از پروتکلی با رمزگذاری داخلی خود (Shadowsocks) در شبکه داخلی استفاده می‌شود.

برای inbound هایی که از بیرون قابل دسترسی هستند، حالت «هیچ» توصیه نمی‌شود.

مثال: بلوک `streamSettings` برای TLS در شبکه `tcp` (VLESS/Trojan/VMess). نتیجه پس از انتخاب حالت **TLS** و پر کردن SNI و مسیرهای گواهی اینگونه به نظر می‌رسد:

```
"streamSettings": {
  "network": "tcp",
  "security": "tls",
  "tlsSettings": {
    "serverName": "vpn.example.com",
    "minVersion": "1.2",
    "maxVersion": "1.3",
    "alpn": ["h2", "http/1.1"],
    "settings": { "fingerprint": "chrome" },
    "certificates": [
      {
        "certificateFile": "/root/cert/vpn.example.com.crt",
        "keyFile": "/root/cert/vpn.example.com.key",
        "ocspStapling": 3600,
        "usage": "encipherment"
      }
    ]
  }
}
```

7.3. حالت TLS

فیلدهای بلوک `tlsSettings`. مقادیر پیش‌فرض از طرح پانل گرفته شده‌اند.

پارامترهای اصلی

فیلد (برچسب)	مقدار پیش‌فرض	توضیح
SNI (`serverName`)	`""` (خالی)	Server Name Indication — نام دامنه‌ای که در TLS handshake ارائه می‌شود. باید با دامنه گواهی مطابقت داشته باشد. متن راهنمای انگلیسی: «Server Name Indication».
Cipher Suites (`cipherSuites`)	`""` → **خودکار**	فهرست مجموعه‌های رمز مجاز. به‌طور پیش‌فرض خالی است — انتخاب به Xray/Go واگذار می‌شود (گزینه **خودکار**). فقط در صورت نیاز به محدود کردن صریح رمزها تغییر دهید.
حداقل/حداکثر نسخه (`minMaxVersion`)	min = `1.2`، max = `1.3`	حداقل و حداکثر نسخه‌های TLS. مقادیر در دسترس: `1.0`، `1.1`، `1.2`، `1.3`. توصیه می‌شود `1.2` – `1.3` نگه دارید؛ کاهش حداقل به 1.0/1.1 مطلوب نیست (نسخه‌های منسوخ و ناامن).
uTLS (`settings.fingerprint`)	`chrome` (در فرم — گزینه **None** = `""` در دسترس است)	اثر انگشت TLS تقلیدی برای client hello (uTLS fingerprint) تا handshake مانند مرورگر محبوب به نظر برسد. فهرست را در پایین ببینید. در TLS اولین گزینه فهرست **None** (`""`) است که تقلید را غیرفعال می‌کند.
ALPN (`alpn`)	`["h2", "http/1.1"]`	فهرست پروتکل‌های لایه کاربردی که در TLS مذاکره می‌شوند (انتخاب چندگانه). مقادیر مجاز: `h3`، `h2`، `http/1.1`. به‌طور پیش‌فرض `h2` و `http/1.1` پیشنهاد می‌شود.

مقادیر ممکن برای **uTLS fingerprint** (یکسان برای TLS و REALITY): `chrome`، `firefox`، `safari`، `ios`، `android`، `edge`، `360`، `qq`، `random`، `randomized`، `randomizednoalpn`، `unsafe`. در فرم TLS، گزینه خالی **None** نیز در دسترس است (تقلید اثر انگشت اعمال نمی‌شود).

مقادیر در دسترس برای **Cipher Suites** (TLS 1.3 و مجموعه‌های ECDHE): `TLS_AES_128_GCM_SHA256`، `TLS_AES_256_GCM_SHA384`، `TLS_CHACHA20_POLY1305_SHA256`، `TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA`، `TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA`، `TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA`، `TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA`، `TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256`، `TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384`، `TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256`، `TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384`، `TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256`، `TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256`.

کلیدهای جابجایی TLS

کلید جابجایی	پیش‌فرض	توضیح
رد کردن SNI ناشناخته (`rejectUnknownSni`)	خاموش (`false`)	اگر فعال باشد، سرور اتصال را قطع می‌کند وقتی SNI ارائه‌شده توسط کلاینت با نام موجود در گواهی مطابقت ندارد. پنهان‌کاری را افزایش می‌دهد (سرور به درخواست‌های «بیگانه» پاسخ نمی‌دهد)، اما به تطابق دقیق SNI در کلاینت نیاز دارد.
غیرفعال کردن System Root (`disableSystemRoot`)	خاموش (`false`)	استفاده از مخزن گواهی ریشه معتمد سیستم را غیرفعال می‌کند.

کلید جابجایی	پیش‌فرض	توضیح
از سرگیری نشست (`enableSessionResumption`)	خاموش (`false`)	از سرگیری نشست TLS (session resumption / session tickets) را فعال می‌کند.

پارامترهای اضافی TLS (vcn، منحنی‌ها، لاگ کلید، ECH Sockopt)

زیر تنظیمات اصلی TLS، فیلدهای اضافی در دسترس هستند.

فیلد (برچسب)	پیش‌فرض	توضیح
Verify Peer Cert By Name (`settings.verifyPeerCertByName`)	`""`	نام‌ها (با کاما جدا شده) که کلاینت با آن‌ها گواهی سرور را به جای SNI تأیید می‌کند. این جایگزین مدرن فیلد `allowInsecure` است که پس از 2026-06-01 از Xray حذف شد. مقدار فقط برای پانل است: در کانفیگ xray سرور نوشته نمی‌شود، اما در لینک‌های دعوت و اشتراک‌ها منتقل می‌شود (`vcn=…`) تا کلاینت آن را اعمال کند. متن راهنما: `example.com` .
Curve Preferences (`curvePreferences`)	`""`	محدودیت و ترتیب منحنی‌های تبادل کلید TLS، به ترتیب اولویت (مثلاً `X25519MLKEM768` ، `X25519`). خالی — مقادیر پیش‌فرض Xray-core استفاده می‌شود.
Master Key Log (`masterKeyLog`)	`""`	مسیر برای ثبت کلیدهای TLS master در قالب `SSLKEYLOGFILE` (برای رمزگشایی ترافیک در Wireshark هنگام اشکال‌زدایی). متن راهنما: `/path/to/sslkeylog.txt` . در محیط production خالی بگذارید — فایل اجازه رمزگشایی تمام ترافیک را می‌دهد.
ECH Sockopt (`echSockopt`)	خاموش	کلید جابجایی با پارامترهای سوکت برای اتصالی که Xray از طریق آن فهرست ECH config را درخواست می‌کند. هنگام فعال‌سازی در دسترس هستند: **Dialer Proxy** (`dialerProxy`) — درخواست را از طریق outbound مشخص‌شده با تگ هدایت کنید)، **Domain Strategy** (`domainStrategy`)، **TCP Fast Open** (`tcpFastOpen`)، **Multipath TCP** (`tcpMptcp`). در صورت عدم ضرورت خاموش بگذارید.

فیلدهای `verifyPeerCertByName` ، `curvePreferences` ، `masterKeyLog` و `echSockopt` در سطح بالای `tlsSettings` قرار دارند و از «تراشیدن» فیلدهای پانل هنگام ذخیره کانفیگ جان سالم به در می‌برند.

گواهی‌ها

بخش **SSL-گواهی** (در UI با عنوان «SSL-گواهی») به عنوان فهرست تنظیم می‌شود: با دکمه **+** یک ورودی گواهی جدید اضافه می‌شود، با دکمه **– حذف** — حذف می‌شود (دکمه حذف فقط وقتی بیشتر از یک ورودی وجود داشته باشد در دسترس است). به‌طور پیش‌فرض هنگام فعال‌سازی TLS یک ورودی خالی ایجاد می‌شود.

برای هر ورودی، کلید جابجایی حالت ورودی (`useFile`):

- **مسیر گواهی** (مقدار `useFile = true` ، پیش‌فرض) — مسیرهای فایل‌ها روی سرور مشخص می‌شوند:
  - **کلید عمومی** (`certificateFile`) — مسیر فایل گواهی (`.crt` / `.pem`)؛
  - **کلید خصوصی** (`keyFile`) — مسیر فایل کلید خصوصی (`.key`).
- **محتوای گواهی** (مقدار `useFile = false`) — محتوا مستقیماً در فیلدها (ناحیه‌های متن چندخطی) وارد می‌شود:
  - **کلید عمومی** (`certificate`) — محتوای PEM گواهی؛
  - **کلید خصوصی** (`key`) — محتوای PEM کلید.

زیر فیلدهای حالت «مسیر گواهی» دو دکمه در دسترس است:

- **نصب گواهی پانل** — مسیرهای SSL-گواهی خود پانل را در فیلدها جایگزین می‌کند. برای inbound روی پانل مرکزی، گواهی آن استفاده می‌شود (`webKeyFile` / `webCertFile` → `POST /panel/setting/all`)؛ برای inbound اختصاص‌یافته به گره — گواهی خود گره (`GET /panel/api/nodes/webCert/{nodeId}`)، زیرا مسیرهای پانل مرکزی روی گره وجود ندارند. اگر گواهی تنظیم نشده باشد، هشدار نمایش داده می‌شود: «برای پانل هیچ گواهی‌ای تنظیم نشده است. ابتدا آن را در تنظیمات نصب کنید.» (گواهی خود پانل در بخش «تنظیمات → امنیت» تنظیم می‌شود).
- **پاک کردن** — هر دو مسیر را پاک می‌کند.

فیلدهای اضافی هر ورودی گواهی:

فیلد	پیش‌فرض	توضیح
OCSP Stapling (`ocspStapling`)	`0` (خاموش)	فاصله به‌روزرسانی OCSP stapling بر حسب ثانیه (حداقل `0`). برای inbound های جدید به‌طور پیش‌فرض خاموش است (`0`): این خطاهای xray را برای گواهی‌هایی بدون OCSP responder (مثلاً Let's Encrypt که OCSP را کنار گذاشته) حذف می‌کند. فقط برای گواهی‌هایی که از stapling پشتیبانی می‌کنند فعال کنید.
بارگذاری یک‌باره (`oneTimeLoading`)	خاموش (`false`)	اگر فعال باشد، گواهی یک‌بار در هنگام راه‌اندازی از دیسک خوانده می‌شود و با تغییر فایل مجدداً خوانده نمی‌شود.
گزینه استفاده (`usage`)	`encipherment`	کاربرد گواهی مجاز است: `encipherment` (رمزگذاری — گواهی معمول سرور)، `verify` (تأیید)، `issue` (صدور — سرور خودش گواهی‌ها را امضا/صادر می‌کند).
Build Chain (`buildChain`)	خاموش (`false`)	**فقط** در `usage = issue` نمایش داده می‌شود. ساخت زنجیره گواهی.

> دکمه جداگانه‌ای برای گواهی خودامضا در ویرایشگر inbound وجود ندارد: پانل در لحظه گواهی خودامضا برای inbound تولید نمی‌کند. گواهی یا از طریق مسیر/محتوا مشخص می‌شود، یا با دکمه «نصب گواهی پانل» از تنظیمات پانل کشیده می‌شود. صدور/دریافت SSL-گواهی خود پانل (از جمله آپلود فایل‌ها و اتصال به دامنه) در بخش **تنظیمات** → **امنیت** انجام می‌شود؛ اندپوینت‌های ACME/Let's Encrypt برای inbound اینجا وجود ندارند.

ECH و pinning گواهی (فیلدهای پیشرفته TLS)

فیلد	پیش‌فرض	توضیح
ECH key (`echServerKeys`)	`""`	کلیدهای سرور برای Encrypted Client Hello.
ECH config (`settings.echConfigList`)	`""`	فهرست ECH config (بخش کلاینت، در لینک قرار می‌گیرد).
SHA-256 گواهی همتا (`settings.pinnedPeerCertSha256`)	`[]`	هش‌های SHA-256 گواهی همتا (رشته‌های hex، با کاما جدا شده). متن راهنمای دقیق: «هش‌های *SHA-256* گواهی همتا به صورت رشته شانزده‌هشتی (مثلاً *e8e2d3...*)، با کاما جدا شده. فقط برای پانل — در کانفیگ *xray* سرور نوشته نمی‌شود، اما در لینک‌های دعوت قرار می‌گیرد تا کلاینت‌ها بتوانند گواهی را *pin* کنند.»

دکمه‌ها: کنار فیلد **SHA-256 گواهی همتا** دو دکمه پر کردن خودکار وجود دارد:

- **Fill from this inbound's certificate** (آیکون سپر) — هش SHA-256 گواهی خود این inbound را جایگزین می‌کند (از طریق اندپوینت `getCertHash` به‌صورت محلی گرفته می‌شود).

- **Fetch the hash by pinging the SNI (xray tls ping)** (آیکون بارگذاری) — هش گواهی زنده سرور را با اتصال TLS به SNI مشخص‌شده دریافت می‌کند (روی سرور `getRemoteCertHash` فراخوانی می‌شود). فیلد **SNI** (`serverName`) باید پر شده باشد — در غیر این صورت راهنمایی نمایش داده می‌شود «*Set the SNI (serverName) first to ping the remote certificate*».

هش‌های دریافت‌شده به فیلد اضافه می‌شوند (با کاما) و در لینک‌های دعوت قرار می‌گیرند تا کلاینت بتواند گواهی را pin کند.

- **دریافت گواهی ECH جدید** — یک جفت ECH جدید برای SNI فعلی از سرور درخواست می‌کند (`POST /panel/api/server/getNewEchCert`، روی سرور `xray tls ech --serverName <SNI>` اجرا می‌شود)؛ فیلدهای **ECH key** و **ECH config** را پر می‌کند.
- **پاک کردن** — هر دو فیلد ECH را خالی می‌کند.

7.4. حالت REALITY

فیلدهای بلوک `realitySettings` . REALITY از SSL-گواهی استفاده نمی‌کند: به جای آن — handshake TLS قرضی از دامنه خارجی و جفت کلید X25519.

پارامترهای مبهم‌سازی

فیلد (برچسب)	مقدار پیش‌فرض	توضیح
نمایش (`show`)	خاموش (`false`)	خروجی اشکال‌زدایی REALITY در لاگ‌های Xray. معمولاً خاموش می‌ماند.
Xver (`xver`)	`0`	نسخه پروتکل PROXY که به backend منتقل می‌شود (`0` — خاموش). حداقل `0` .
uTLS (`settings.fingerprint`)	`chrome`	اثر انگشت TLS تقلیدی (همان فهرست TLS، اما بدون گزینه خالی None).
هدف (`target`)	`""` (از 3.4.2 هنگام فعال‌سازی REALITY خالی می‌ماند)	**فیلد اجباری.** دامنه واقعی که REALITY handshake TLS آن را قرض می‌گیرد. متن راهنمای دقیق: «اجباری. باید حاوی پورت باشد (مثلاً *example.com:443*). بدون پورت *Xray-core* اجرا نمی‌شود.» اعتبارسنجی پانل وجود و صحت پورت را بررسی می‌کند؛ در غیر این صورت خطاهای «هدف REALITY اجباری است» / «هدف REALITY باید حاوی پورت باشد...» / «هدف REALITY دارای پورت نامعتبر است» نمایش داده می‌شود. کنار این فیلد — دکمه‌های **«اسکن»** (بررسی «زنده» هدف فعلی) و **«یافتن هدف‌ها»** (باز کردن اسکنر هدف REALITY)؛ به ادامه مراجعه کنید.
SNI (`serverNames`)	`[]` (همراه با هدف جایگزین می‌شود)	فهرست SNI های مجاز (ورودی چندگانه با تگ). باید با دامنه موجود در **هدف** مطابقت داشته باشد. هنگام اسکن موفق هدف، SNI بر اساس گواهی آن پر می‌شود.
حداکثر اختلاف زمانی (ms) (`maxTimediff`)	`0`	حداکثر اختلاف ساعت مجاز بین کلاینت و سرور بر حسب میلی‌ثانیه (`0` — بدون محدودیت). حداقل `0` .
حداقل نسخه کلاینت (`minClientVer`)	`""`	حداقل نسخه Xray-client (متن راهنما `25.9.11`). خالی — بدون محدودیت.
حداکثر نسخه کلاینت (`maxClientVer`)	`""`	حداکثر نسخه Xray-client. خالی — بدون محدودیت.
Short IDs (`shortIds`)	`[]` (هنگام فعال‌سازی تولید می‌شوند)	فهرست شناسه‌های کوتاه (hex)، که کلاینت‌ها را از هم متمایز می‌کند. ورودی چندگانه با تگ؛ دکمه به‌روزرسانی یک مجموعه تصادفی تولید می‌کند.

فیلد (برچسب)	مقدار پیش‌فرض	توضیح
SpiderX (`settings.spiderX`)	`/`	مسیر «عنکبوت» (بخش کلاینت REALITY)، که هنگام شبیه‌سازی دسترسی به سایت خارجی استفاده می‌شود. در لینک دعوت قرار می‌گیرد.

از نسخه 3.4.2 **هدف** (`target`) و **SNI** (`serverNames`) هنگام فعال‌سازی REALITY دیگر به‌طور خودکار پر **نمی‌شوند** — هر دو فیلد خالی می‌مانند، و هدف توسط اسکنر زنده انتخاب می‌شود (به ادامه مراجعه کنید). یک سایت HTTPS معتبر، پایدار و خارجی با پشتیبانی از TLS 1.3 و HTTP/2 انتخاب کنید که پشت سرور خودتان نباشد.

یافتن هدف REALITY (اسکنر زنده)

از نسخه 3.4.2 دکمه قبلی «هدف تصادفی» (با فهرست داخلی ثابت) با دو اقدام کنار فیلد **هدف** جایگزین شده است:

- **«اسکن»** — هدف فعلی را «زنده» بررسی می‌کند: یک اتصال TLS برقرار می‌کند و اگر هدف مناسب باشد، SNI را از گواهی آن جایگزین می‌کند. اعلان‌ها: «هدف مناسب است — هدف و SNI پر شدند.» / «هدف در دسترس است، اما برای REALITY مناسب نیست.» / «اسکن هدف REALITY ناموفق بود.».
- **«یافتن هدف‌ها»** — پنجره **«اسکنر هدف REALITY»** را باز می‌کند: می‌توانید یک دامنه، **IP یا بازه CIDR** را مشخص کنید (در این صورت پنل هدف‌ها را بر اساس گواهی‌های هاست‌های زنده می‌یابد) یا فیلد را خالی بگذارید (نامزدهای داخلی بررسی می‌شوند). نتایج رتبه‌بندی می‌شوند؛ ستون‌ها — «وضعیت»/«مناسب»، «تبادل کلید»، «گواهی»، `TLS`، `ALPN`، «تأخیر». با دکمه **«استفاده»** ردیف انتخاب‌شده در فیلدهای inbound جایگزین می‌شود.

یک هدف زمانی **مناسب** شمرده می‌شود که سرور **TLS 1.3**، **HTTP/2 (ALPN h2)**، تبادل کلید **X25519/X25519MLKEM768** را توافق کند و یک گواهی **معتبر** (نه wildcard) ارائه دهد. اسکن در سمت سرور پنل انجام می‌شود (`POST /panel/api/server/scanRealityTarget` و `scanRealityTargets/…`).

مثال: بلوک `streamSettings` برای REALITY در شبکه `tcp` (VLESS). به گواهی نیاز نیست — به جای آن دامنه قرضی و جفت کلید X25519:

```
"streamSettings": {
  "network": "tcp",
  "security": "reality",
  "realitySettings": {
    "show": false,
    "xver": 0,
    "dest": "www.nvidia.com:443",
    "serverNames": ["www.nvidia.com"],
    "privateKey": "YOUR_X25519_PRIVATE_KEY",
    "shortIds": ["", "6ba85179e30d4fc2"],
    "settings": {
      "publicKey": "YOUR_X25519_PUBLIC_KEY",
      "fingerprint": "chrome",
      "spiderX": "/"
    }
  }
}
```

اینجا فیلد **هدف** (`target`) پانل با `dest` در کانفیگ آماده Xray مطابقت دارد. اگر REALITY-inbound با destination در کلید `dest` ایجاد شده باشد (توسط نسخه‌های قدیمی پانل، از طریق API یا ابزارهای خارجی)، پانل هنگام پردازش `dest` → `target` را نرمال‌سازی می‌کند وقتی `target` خالی باشد — بنابراین چنین inbound به درستی بارگذاری می‌شود، فیلد **هدف** خالی نمی‌ماند، و ذخیره مجدد REALITY کارکرد را خراب نمی‌کند.

کلیدهای REALITY (X25519)

فیلد	پیش‌فرض	توضیح
کلید عمومی (`settings.publicKey`)	`""`	کلید عمومی X25519 (کلاینت آن را در کانفیگ/لینک خود قرار می‌دهد).
کلید خصوصی (`privateKey`)	`""`	کلید خصوصی X25519 (فقط روی سرور ذخیره می‌شود).

دکمه‌های زیر کلیدها:

- **دریافت گواهی جدید** — از سرور یک جفت کلید جدید درخواست می‌کند (`GET /panel/api/server/getNewX25519Cert` ؛ روی سرور `xray x25519` اجرا می‌شود)، **کلید خصوصی** و **کلید عمومی** را پر می‌کند. همین جفت به‌طور خودکار هنگام اولین فعال‌سازی حالت REALITY تولید می‌شود.

مثال: دریافت جفت کلید X25519 از طریق API (خارج از فرم، مثلاً برای اسکریپت). درخواست کلید خصوصی و عمومی را برمی‌گرداند:

```
curl -s -b cookie.txt https://your-panel:2053/panel/api/server/getNewX25519Cert
# پاسخ:
# {"success":true,"obj":{"privateKey":"...","publicKey":"..."}}
```

`cookie.txt` — فایل cookie نشست، که پس از ورود از طریق `POST /login` دریافت می‌شود.

- **پاک کردن** — هر دو کلید را خالی می‌کند.

امضای پسا-کوانتومی ML-DSA-65 (mldsa65)

لایه اضافی (اختیاری) احراز هویت پسا-کوانتومی REALITY:

فیلد	پیش‌فرض	توضیح
mldsa65 Seed (`mldsa65Seed`)	`""`	seed کلید ML-DSA-65 سرور.
mldsa65 Verify (`settings.mldsa65Verify`)	`""`	مقدار تأیید (بخش کلاینت، در لینک قرار می‌گیرد).

دکمه‌ها:

- **دریافت Seed جدید** — یک جفت جدید درخواست می‌کند (`GET /panel/api/server/getNewmldsa65` ؛ روی سرور `xray` `mldsa65` اجرا می‌شود)، **mldsa65 Seed** و **mldsa65 Verify** را پر می‌کند.
- **پاک کردن** — هر دو فیلد را خالی می‌کند.

محدودیت سرعت fallback و لاگ کلید REALITY

در تنظیمات REALITY، محدودیت سرعت ترافیک fallback در دسترس است — این مانع از استفاده کاوشگران فعال از سرور به عنوان کانال رایگان به دامنه قرضی می‌شود. تنظیم به‌طور جداگانه برای دو جهت تعیین می‌شود — **Limit Fallback Upload** و **Limit Fallback Download** (`limitFallbackUpload` / `limitFallbackDownload`)، هر کدام با مجموعه فیلدهای یکسانی:

فیلد (برچسب)	پیش‌فرض	توضیح
After Bytes (`afterBytes`)	`0`	چند بایت با سرعت کامل قبل از شروع محدودیت عبور کند. `0` — از اولین بایت محدود کن.

فیلد (برچسب)	پیش‌فرض	توضیح
Bytes Per Sec (`bytesPerSec`)	`0`	سقف سرعت ترافیک fallback بر حسب بایت در ثانیه پس از آستانه. `0` — بدون محدودیت (این جهت را غیرفعال می‌کند).
Burst Bytes Per Sec (`burstBytesPerSec`)	`0`	ذخیره برای افزایش‌های کوتاه‌مدت فراتر از سرعت ثابت (اندازه token-bucket). اگر کمتر از **Bytes Per Sec** باشد، به مقدار آن افزایش می‌یابد.

همانجا فیلد **Master Key Log** (`masterKeyLog`) نیز اضافه شده — مسیر برای ثبت کلیدهای TLS master در قالب `SSLKEYLOGFILE` برای اشکال‌زدایی در Wireshark؛ در محیط production خالی بگذارید.

7.5. توصیه‌های عملی برای پیکربندی

1. **VLESS + Reality (توصیه می‌شود):** یک VLESS-inbound روی شبکه `tcp` ایجاد کنید، در تب «امنیت» **Reality** را انتخاب کنید — پانل خودش `target` /SNI تصادفی، `shortIds` جایگزین می‌کند و کلیدهای X25519 تولید می‌کند. در صورت نیاز برای جفت کلید خود «دریافت گواهی جدید» را بزنید. برای کلاینت‌های VLESS **Flow** = `xtls-rprx-vision` (XTLS Vision) را فعال کنید — این حداکثر عملکرد و پنهان‌کاری را فراهم می‌کند.

مثال: لینک نهایی کلاینت VLESS + Reality + Vision. اینگونه لینک دعوتی که پانل برای چنین inbound صادر می‌کند به نظر می‌رسد (مقادیر کلید/ID نمایشی هستند):

```
vless://uuid-клиента@1.2.3.4:443?
type=tcp&security=reality&pbk=ПУБЛИЧНЫЙ_КЛЮЧ&fp=chrome&sni=www.nvidia.com&sid=6ba85179e3
0d4fc2&spx=%2F&flow=xtls-rprx-vision#my-reality
```

اینجا `pbk` — کلید عمومی X25519، `sni` — دامنه قرضی از **هدف**، `sid` — یکی از **Short IDs**، `flow=xtls-rprx-vision` — XTLS Vision فعال‌شده. 2. **TLS با دامنه اختصاصی: TLS** را انتخاب کنید، **SNI** را با نام دامنه پر کنید، گواهی اضافه کنید (از طریق مسیر فایل‌ها یا محتوا)، یا «نصب گواهی پانل» را بزنید اگر دامنه و گواهی قبلاً در «تنظیمات → امنیت» تنظیم شده باشند. **حداقل/حداکثر نسخه** = `1.2` – `1.3` و **uTLS** = `chrome` را برای شبیه‌سازی مرورگر معمول نگه دارید. 3. حالت **هیچ** را برای inbound هایی که به بیرون باز هستند نگذارید — فقط برای اهداف fallback محلی (`127.0.0.1`) یا زمانی که TLS توسط پروکسی خارجی تأمین می‌شود استفاده کنید. 4. توصیه از رابط کاربری: برای اکثر فیلدهای پیشرفته راهنمای «توصیه می‌شود تنظیمات پیش‌فرض را نگه دارید» اعمال می‌شود — فقط با درک عواقب آن‌ها را تغییر دهید.

8. کلاینت‌ها

کلاینت یک حساب کاربری VPN است: مجموعه‌ای از اطلاعات احراز هویت (UUID یا رمز عبور) که به یک یا چند inbound متصل می‌شود و دارای سهمیه ترافیکی مستقل، تاریخ انقضا و محدودیت اتصال همزمان است. در این فورک، کلاینت یک موجودیت مستقل است (جدول `clients`): یک کلاینت می‌تواند به چندین inbound به‌طور همزمان متصل باشد و UUID/رمز عبور مشترک و شمارنده ترافیک مشترک داشته باشد. بخش **کلاینت‌ها** تمام حساب‌های پنل را صرف‌نظر از inbound نشان می‌دهد، همراه با جستجو، فیلتر، مرتب‌سازی و عملیات دسته‌جمعی.

8.1. فیلدهای کلاینت

در ادامه هر فیلد ویرایشگر **افزودن کلاینت / ویرایش کلاینت** توضیح داده شده است.

فرم کلاینت به دو برگه تقسیم می‌شود: **اصلی** (email، اتصال به inbound، محدودیت‌ها، مدت، گروه، توضیحات، تگ بازگشتی) و **اطلاعات احراز هویت** (UUID/رمز عبور/VMess Security، Flow، auth). در برچسب فیلدها سهمیه به‌صورت **محدودیت ترافیک (GB)** و مدت‌ها به‌صورت **مدت (روز)** و **تمدید خودکار (روز)** نمایش داده می‌شوند؛ فیلدهای **محدودیت ترافیک (GB)** و **محدودیت IP** راهنمایی دارند که توضیح می‌دهد مقدار `0` به معنای «بدون محدودیت» است. هنگام ویرایش کلاینت موجود، دکمه تولید email تصادفی پنهان می‌شود و دکمه گزارش IP مستقیماً در کنار فیلد **محدودیت IP** قرار می‌گیرد و تعداد آدرس‌های ثبت‌شده را نشان می‌دهد.

فیلد	کلید JSON	پیش‌فرض	توضیح
Email	`email`	— (اجباری)	شناسه یکتای کلاینت
UUID	`id`	تولید می‌شود	شناسه برای VMess/VLESS
رمز عبور	`password`	تولید می‌شود	رمز عبور برای Trojan/Shadowsocks
احراز هویت	`auth`	تولید می‌شود	رمز عبور برای Hysteria
Flow	`flow`	خالی	کنترل جریان (XTLS)، فقط VLESS
VMess Security	`security`	`auto`	روش رمزنگاری VMess
محدودیت IP	`limitIp`	`0` (بدون محدودیت)	حداکثر IP همزمان
مجموع ارسال/دریافت (GB)	`totalGB`	`0` (بدون محدودیت)	سهمیه ترافیک
تاریخ انقضا	`expiryTime`	`0` (بی‌پایان)	تاریخ انقضا
تمدید خودکار	`reset`	`0` (غیرفعال)	دوره بازنشانی ترافیک، روز
شناسه کاربری Telegram	`tgId`	`0` (ندارد)	شناسه عددی Telegram
شناسه اشتراک	`subId`	تولید می‌شود	شناسه اشتراک
گروه	`group`	خالی	برچسب منطقی گروه‌بندی
توضیحات	`comment`	خالی	یادداشت دلخواه
فعال	`enable`	`true`	آیا حساب فعال است

Email (شناسه)

فیلد **Email** شناسه اصلی و اجباری کلاینت است. با وجود نام، لزوماً یک آدرس ایمیل نیست: هر برچسب متنی (نام کاربری، شماره) مناسب است. مقدار باید در سطح پنل **یکتا** باشد؛ تلاش برای ایجاد کلاینت دوم با email تکراری رد می‌شود (`email already in use`)، مگر اینکه `subId` نیز یکسان باشد (در این صورت به‌عنوان اتصال همان کلاینت تفسیر می‌شود).

Email **نمی‌تواند خالی باشد** (`client email is required`) و **نمی‌تواند شامل فاصله، کاراکترهای `/` ، `\` و کاراکترهای کنترلی باشد** («Email نمی‌تواند شامل فاصله، '/'، " یا کاراکترهای کنترلی باشد»). Email در حسابداری ترافیک، گزارش IP، فهرست آنلاین و نام عملیات استفاده می‌شود — تغییر آن به‌صورت بازگشتی توصیه نمی‌شود.

UUID / رمز عبور / احراز هویت (اطلاعات احراز هویت)

فیلد مشخص اطلاعات احراز هویت به پروتکل inbound متصل‌شده بستگی دارد. اگر فیلد خالی بماند، مقادیر به‌صورت خودکار جایگزین می‌شوند:

- **UUID** (فیلد `id`) — برای پروتکل‌های **VMess** و **VLESS**. اگر تعریف نشود، یک UUID v4 تصادفی تولید می‌شود.
- **رمز عبور** (فیلد `password`) — برای **Trojan** و **Shadowsocks**. برای Trojan به‌طور پیش‌فرض UUID بدون خط تیره تولید می‌شود. برای Shadowsocks یک کلید با طول مناسب در Base64 بسته به روش رمزنگاری inbound تولید می‌شود: 16 بایت برای `2022-blake3-aes-128-gcm`، 32 بایت برای `2022-blake3-aes-256-gcm` و `2022-blake3-chacha20-poly1305` ؛ برای سایر روش‌ها — UUID بدون خط تیره. اگر کلید واردشده دستی با روش blake3-2022 سازگار نباشد، با کلید تولیدشده جایگزین می‌شود.
- **احراز هویت** (فیلد `auth`) — رمز عبور برای **Hysteria**. به‌طور پیش‌فرض UUID بدون خط تیره.

از آنجا که یک کلاینت می‌تواند به inboundهای با پروتکل‌های مختلف متصل باشد، رکورد کلاینت می‌تواند به‌طور همزمان UUID، رمز عبور و auth داشته باشد — برای هر پروتکل فیلد مخصوص خودش استفاده می‌شود.

مثال: اطلاعات احراز هویت کلاینت در `settings` inboundهای مختلف چگونه به نظر می‌رسد. همان کلاینت در یک VLESS-inbound با `id` شناسایی می‌شود، در Trojan با `password` ، در Shadowsocks با `password` (کلید Base64):

```
// VLESS-inbound در settings.clients فرگمنت
{ "id": "b831381d-6324-4d53-ad4f-8cda48b30811", "email": "user-a", "flow": "xtls-rprx-vision" }

// Trojan-inbound در کلاینت همان
{ "password": "b831381d63244d53ad4f8cda48b30811", "email": "user-a" }

// Shadowsocks-inbound (روش 2022-blake3-aes-256-gcm) در کلاینت همان
{ "password": "GPyOaA3f7CO0az53eaQ8eqMfRDjmBlOh7v1u3+Z+pHk=", "email": "user-a" }
```

Flow

Flow (فیلد `flow`) — کنترل جریان XTLS. **فقط برای VLESS** و تنها زمانی که inbound برای XTLS Vision پیکربندی شده قابل اعمال است: VLESS روی انتقال **TCP** با security `tls` یا `reality` . مقدار مجاز — `xtls-rprx-vision` (و نیز `xtls-rprx-vision-udp443` به‌صورت تاریخی)؛ مقدار خالی به معنای نبود flow است.

اگر inbound از XTLS-flow پشتیبانی نکند، flow تعریف‌شده هنگام ذخیره کلاینت **بی‌سروصدا صفر می‌شود**: برای یک کلاینت متصل به چندین inbound، flow فقط در جایی که مجاز است اعمال می‌شود. تغییر آن فقط در صورتی توصیه می‌شود که به‌طور هدفمند از VLESS-Vision استفاده می‌کنید.

VMess Security

VMess Security (فیلد `security`) — روش رمزنگاری محتوا برای VMess. مقدار پیش‌فرض `auto` است (Xray خودش رمز را انتخاب می‌کند). مقادیر مجاز — استانداردهای VMess: `auto`، `aes-128-gcm`، `chacha20-poly1305`، `none`، `zero`. برای سایر پروتکل‌ها این فیلد استفاده نمی‌شود.

محدودیت IP (اتصالات همزمان)

محدودیت IP (فیلد `limitIp`) — حداکثر تعداد **آدرس‌های IP متفاوت** که کلاینت می‌تواند به‌طور همزمان متصل باشد. مقدار پیش‌فرض `0` به معنای **عدم محدودیت** است. با مقدار مثبت، پنل IPهای فعال کلاینت را ردیابی می‌کند و در صورت تجاوز از محدودیت، حساب را از طریق یک کار پس‌زمینه غیرفعال می‌کند. (از نسخه **3.3.1** شمارش IP از طریق API آمارهای آنلاین هسته Xray انجام می‌شود و به گزارش دسترسی **نیازی نیست**؛ در نسخه‌های قدیمی‌تر هسته، پنل به خواندن گزارش دسترسی برمی‌گردد که در آن صورت باید فعال باشد.) برای جلوگیری از اشتراک‌گذاری یک اشتراک در دستگاه‌های زیاد استفاده کنید: مثلاً `2` — دو دستگاه مجاز است.

محدودیت IP با استفاده از **Fail2ban** اعمال می‌شود، بنابراین فیلد **محدودیت IP** فقط زمانی فعال است که Fail2ban نصب و در حال اجرا باشد (پنل وضعیت آن را از طریق `GET /panel/api/server/fail2banStatus` بررسی می‌کند). اگر Fail2ban نصب نشده باشد، فیلد ویرایشگر کلاینت (و فرم افزودن دسته‌جمعی) غیرفعال می‌شود و با نگه داشتن ماوس روی آن، راهنمایی با پیشنهاد نصب Fail2ban از منوی bash مربوط به `x-ui` نمایش می‌یابد («Fail2ban is not installed, so the IP limit cannot be enforced. Install Fail2ban from the x-ui bash menu to enable this option.»)؛ در ویندوز راهنما اعلام می‌کند که Fail2ban در آنجا در دسترس نیست («Fail2ban is not available on Windows, so the IP limit cannot be enforced.»)، و اگر این قابلیت در سرور غیرفعال باشد — «The IP limit feature is disabled on this server.». هنگام به‌روزرسانی پنل، محدودیت IP ذخیره‌شده کلاینت‌ها در سرورهای بدون Fail2ban توسط یک مهاجرت یکباره به صفر تنظیم می‌شود، چون در آنجا اعمال نمی‌شد.

نمونه مقادیر. `limitIp: 0` — بدون محدودیت؛ `limitIp: 1` — دقیقاً یک دستگاه به‌طور همزمان؛ `limitIp: 3` — حداکثر سه IP متفاوت. با چهارمین IP فعال، کار پس‌زمینه کلاینت را غیرفعال می‌کند (`enable = false`) تا **بازنشانی محدودیت IP** را اجرا کنید.

عملیات مرتبط: **گزارش IP** فهرست IPهای ثبت‌شده کلاینت را نشان می‌دهد؛ هر رکورد علاوه بر خود IP، زمان آخرین درخواست و برچسب نود (`@ نام_نود`) که از طریق آن اتصال ثبت شده را دارد — در پیکربندی چند پنلی مشخص است که کلاینت از کدام گره متصل شده. **بازنشانی محدودیت IP** گزارش IP انباشته‌شده را پاک می‌کند تا کلاینت بدون انتظار برای انقضای طبیعی رکوردها دوباره بتواند متصل شود.

مجموع ارسال/دریافت (GB) — سهمیه ترافیک

مجموع ارسال/دریافت (GB) (فیلد `totalGB`) — سهمیه کل ترافیک (ارسال + دریافت). مقدار پیش‌فرض `0` به معنای **بدون محدودیت** است. با رسیدن به سهمیه (`up + down >= total`)، کلاینت **تمام‌شده** (depleted) محسوب می‌شود و غیرفعال می‌شود. در رابط کاربری معمولاً بر حسب گیگابایت وارد می‌شود؛ در پایگاه داده به بایت ذخیره می‌شود.

در فهرست کلاینت‌ها ستون **ترافیک** یک نوار رنگی از میزان مصرف نشان می‌دهد: حجم ترافیک مصرف‌شده، برچسب محدودیت (یا نماد ∞ در صورت بی‌محدودیت) و راهنمای نگه‌داشتن ماوس با تفکیک ارسال/دریافت و باقیمانده. همان نشانگر فشرده در کارت‌های کلاینت‌ها در گوشی نیز نمایش داده می‌شود.

تاریخ انقضا (Expiry)

تاریخ انقضا (فیلد `expiryTime`) لحظه انقضای حساب را تعریف می‌کند. این فیلد سه حالت دارد:

- **بی‌پایان** — `0`. کلاینت هرگز از نظر زمانی منقضی نمی‌شود.
- **تاریخ مشخص** — Unix-timestamp مثبت (به میلی‌ثانیه). با رسیدن (`expiryTime <= اکنون`) کلاینت منقضی‌شده (expired) محسوب می‌شود و غیرفعال می‌شود. در رابط کاربری معمولاً با انتخاب تاریخ یا مدت به روز (**مدت**، واحد — **روز**) تعریف می‌شود.

- **شروع پس از اولین استفاده** — مقدار **منفی** که مدت را کُد می‌کند. تا زمانی که کلاینت حتی یک بایت ارسال نکرده، مدت منفی می‌ماند («شروع به‌تأخیرافتاده»). در اولین تیک حسابداری ترافیک، پنل آن را به یک تاریخ مطلق تبدیل می‌کند: `| اکنون + |مدت`. این امکان می‌دهد، مثلاً، «30 روز از لحظه اولین اتصال» را بدون دانستن زمان فعال‌سازی کلاینت بفروشید. تبدیل یک بار بر اساس email انجام می‌شود تا همه inboundهای متصل تاریخ انقضای یکسانی داشته باشند.

نمونه کُدگذاری انقضا. تاریخ ثابت 1 مارس 2026، 00:00 UTC ← `expiryTime: 1772323200000` (timestamp مثبت به میلی‌ثانیه). «30 روز از اولین اتصال» ← `expiryTime: -2592000000` (مقدار منفی، `30 × 24 × 60 × 60 × 1000`)؛ با اولین بایت ترافیک پنل آن را با `اکنون + 2592000000` جایگزین می‌کند. بی‌پایان ← `expiryTime: 0`.

تمدید خودکار (دوره بازنشانی ترافیک کلاینت)

فیلد **تمدید خودکار** (فیلد `reset`) — دوره تمدید/بازنشانی خودکار به روز است. راهنما: «تمدید خودکار پس از پایان (0 = غیرفعال) (واحد: روز)».

- `0` — تمدید خودکار **غیرفعال** است (مقدار پیش‌فرض). پس از انقضا، کلاینت به‌سادگی تمام‌شده می‌شود.
- `> 0` — کار پس‌زمینه هنگام انقضا **شمارنده‌های ترافیک را به صفر بازنشانی می‌کند** (`up = down = 0`)، **تاریخ انقضا را** به میزان `reset` روز به جلو **می‌برد** (در صورت نیاز چند دوره، تا زمانی که تاریخ جدید در آینده باشد) و در صورت لزوم کلاینت را دوباره **فعال** می‌کند. این یک اشتراک دوره‌ای (مثلاً ماهانه) را پیاده‌سازی می‌کند. تمدید خودکار **برای inboundهای روی نودهای گره (`node_id IS NOT NULL`) اعمال نمی‌شود.**

پیامد مهم: کلاینت‌های دارای `reset > 0` از مفهوم «تمام‌شده» در عملیات حذف دسته‌جمعی **مستثنی می‌شوند** — ترافیک/انقضای آن‌ها به‌طور مورد انتظار توسط تمدید خودکار صفر می‌شود، نه اینکه حساب را کاندیدای حذف کند.

شناسه کاربری Telegram

شناسه کاربری Telegram (فیلد `tgId`) — شناسه عددی Telegram کاربر برای اتصال به ربات Telegram داخلی پنل (اعلان‌ها، مشاهده مستقل آمار). راهنما: «شناسه عددی کاربر Telegram (0 = ندارد)». مقدار پیش‌فرض `0` — هیچ اتصالی وجود ندارد. این فیلد قابل فیلتر کردن است (**دارد / ندارد**).

شناسه اشتراک (subId)

شناسه اشتراک (فیلد `subId`) — شناسه‌ای که کلاینت زیر آن در **اشتراک** (subscription) گنجانده می‌شود. همه کلاینت‌های با یک `subId` یکسان از طریق یک لینک اشتراک ارائه می‌شوند. اگر هنگام ایجاد فیلد خالی باشد، پنل به‌طور خودکار یک `subId` تصادفی (UUID) **تولید می‌کند**. مقدار باید بین کلاینت‌های با email متفاوت **یکتا** باشد (`subId already in use`) و همان محدودیت‌های کاراکتری که برای email وجود دارد را دارد («شناسه اشتراک نمی‌تواند شامل فاصله، '/'، " یا کاراکترهای کنترلی باشد»).

بدون `subId` لینک اشتراک برای کلاینت در دسترس نیست («این کلاینت subId ندارد، لینک اشتراک عمومی در دسترس نیست.»).

برگه Links (لینک‌های خارجی و اشتراک‌ها)

علاوه بر برگه‌های **اصلی** و **اطلاعات احراز هویت**، در ویرایشگر کلاینت یک برگه سوم **Links** وجود دارد (راهنما: «Add third-party share links and remote subscription URLs to include in this client's subscription»). در آن با دکمه **Add External Link** لینک‌های اشتراک‌گذاری شخص ثالث (`vless://`، `vmess://`، `trojan://`، `ss://`، `hysteria2://`، `wireguard://`) اضافه می‌شوند، و با دکمه **Add External Subscription** — URL اشتراک‌های راه دور (مثلاً `https://provider.example/sub/…`).

همه موارد ذکرشده به خروجی اشتراک این کلاینت (فرمت‌های JSON، raw و Clash) افزوده می‌شوند: لینک‌ها همان‌طور که هستند اضافه می‌شوند، و اشتراک‌های راه دور را پنل به‌صورت دوره‌ای دریافت می‌کند (با کَش و تایم‌اوت کوتاه) و پیکربندی‌های آن‌ها را با پیکربندی‌های خودش ادغام می‌کند. به این ترتیب می‌توان در یک لینک اشتراک کلاینت، کنار سرورهای خودی، پیکربندی‌های خارجی هم ارائه داد.

گروه

گروه (فیلد `group`) — برچسب منطقی برای گروه‌بندی کلاینت‌های مرتبط. راهنما: «برچسب منطقی برای گروه‌بندی کلاینت‌های مرتبط (مثلاً تیم، مشتری، منطقه). از نوار ابزار قابل فیلتر است.»، placeholder — «مثلاً customer-a». این فیلد اختیاری است (پیش‌فرض خالی). می‌توان از طریق گروه فهرست را فیلتر کرد و عملیات دسته‌جمعی انجام داد؛ برای پاک کردن برچسب از یک کلاینت از عملیات **حذف از گروه** استفاده می‌شود.

می‌توان گروه را مستقیماً در ویرایشگر یک کلاینت هم حذف کرد: اگر فیلد **گروه** را خالی کرده و ذخیره کنید، برچسب به‌درستی حذف می‌شود و کلاینت دیگر زیر گروه قبلی نمایش نمی‌یابد.

توضیحات

توضیحات (فیلد `comment`) — یادداشت متنی دلخواه برای مدیر (پیش‌فرض خالی). محتوا در جستجو گنجانده می‌شود و قابل فیلتر است (**دارد** / **ندارد** توضیحات).

فعال

فعال (فیلد `enable`) — پرچم فعال بودن حساب. به‌طور پیش‌فرض **فعال** (`true`) است؛ هنگام ایجاد، حتی اگر پرچم ارسال نشود، پنل به‌اجبار `true` قرار می‌دهد. کلاینت غیرفعال (`enable = false`) نمی‌تواند متصل شود و در خلاصه وضعیت در دسته **غیرفعال** (deactive) قرار می‌گیرد. پنل به‌طور خودکار کلاینت‌هایی را که سهمیه‌شان تمام شده، منقضی شده‌اند یا از محدودیت IP تجاوز کرده‌اند غیرفعال می‌کند.

فیلدهای فقط خواندنی

در کارت کلاینت فیلدهای خدماتی هم نمایش داده می‌شوند: **ایجادشده** (`created_at`) و **به‌روزشده** (`updated_at`) — مهرهای زمانی ایجاد و آخرین تغییر که به‌طور خودکار پر می‌شوند و قابل ویرایش نیستند. فیلد **تگ بازگشتی** (`reverse`) — تگ Reverse اختیاری برای پروکسی معکوس ساده VLESS («تگ Reverse اختیاری»).

8.2. اتصال به inbound

هر کلاینت باید حداقل به یک inbound متصل باشد — هنگام ایجاد حداقل یکی الزامی است (`at least one inbound is required`). در ویرایشگر این فیلد **inboundهای متصل** با راهنمای **یک یا چند inbound انتخاب کنید** است.

- **اتصال** — اضافه کردن کلاینت به inboundهای انتخاب‌شده (همان UUID/رمز عبور و ترافیک مشترک). اتصالات موجود حفظ می‌شوند.
- **قطع اتصال** — حذف کلاینت از inboundهای انتخاب‌شده. رکورد کلاینت حفظ می‌شود (برای حذف کامل از **حذف** استفاده کنید). جفت‌هایی که کلاینت به آن‌ها متصل نبوده، بی‌سروصدا رد می‌شوند.

هنگام ذخیره کلاینتی که به چندین inbound متصل است، فیلدهایی که با پروتکل/انتقال خاص ناسازگار هستند (مثلاً Flow خارج از VLESS-Vision)، به‌طور خودکار برای هر inbound به مقادیر مجاز تبدیل می‌شوند.

بالای فهرست انتخاب inbound (در فرم کلاینت، هنگام افزودن دسته‌جمعی کلاینت‌ها و در پنجره‌های اتصال/قطع اتصال دسته‌جمعی) دکمه‌های **انتخاب همه** و **پاک‌کردن** وجود دارد. در این فهرست‌ها هر inbound با یادداشتش (remark) برچسب می‌خورد، اگر تعریف شده باشد، در غیر این صورت با تگ inbound.

8.3. عملیات روی کلاینت

برای یک کلاینت خاص (از طریق کارت **اطلاعات کلاینت** یا منوی زمینه **اقدامات**) موارد زیر در دسترس است:

مشاهده اطلاعات، QR-کد و لینک

- **اطلاعات کلاینت** — کارت با تمام فیلدها، ترافیک مصرف‌شده/باقیمانده (**باقیمانده**)، تاریخ انقضا و inboundهای متصل.

درخواست کلاینت از طریق API (`GET /panel/api/clients/get/:email`) در کنار فیلدهای `client` و `inboundIds` همچنین `usedTraffic` را برمی‌گرداند — ترافیک واقعاً مصرف‌شده (ارسال + دریافت، با احتساب داده‌های نودها)، که مقایسه مصرف با سهمیه `totalGB` را ساده‌تر می‌کند.

- **QR-کد** و **لینک** — لینک پیکربندی کلاینت برای وارد کردن در برنامه کلاینت. بر اساس تمام inboundهای متصل با پروتکل پشتیبانی‌شده تشکیل می‌شود (`GET /links/:email`). اگر لینک مناسبی وجود نداشته باشد: «هیچ لینک اشتراک‌گذاری وجود ندارد — ابتدا کلاینت را به یک inbound با پروتکل پشتیبانی‌شده متصل کنید.».
- **لینک اشتراک** — URL اشتراک بر اساس `subId` (`GET /subLinks/:subId`). فقط در صورتی در دسترس است که کلاینت `subId` داشته باشد و سرویس اشتراک در **تنظیمات پنل** → **اشتراک** فعال باشد (در غیر این صورت «سرویس اشتراک غیرفعال است.»). علاوه بر این **URL اشتراک JSON** ارائه می‌شود.

بازنشانی ترافیک

بازنشانی ترافیک (`POST /resetTraffic/:email`) شمارنده‌های ارسال/دریافت (`up` ، `down`) کلاینت خاص را صفر می‌کند. سهمیه (`totalGB`) و تاریخ انقضا **تغییر نمی‌کنند** — فقط حجم مصرف‌شده صفر می‌شود. اعلان: «ترافیک بازنشانی شد». اگر کلاینت به هیچ inbound متصل نباشد: «ابتدا این کلاینت را به یک inbound متصل کنید.».

دکمه **بازنشانی ترافیک** در فرم ویرایش کلاینت هم در دسترس است — در پنل پایین، کنار **لغو** / **ذخیره** (پیش از بازنشانی تأیید درخواست می‌شود). اگر کلاینت به دلیل تمام شدن ترافیک غیرفعال شده باشد، بازنشانی (هم تک‌تایی هم دسته‌جمعی) به‌طور خودکار آن را دوباره **فعال** می‌کند (`enable = true`) و بلافاصله این تغییر را به نودها ارسال می‌کند — نیازی به فعال کردن دستی کلاینت بر روی مستر و نودها نیست.

بازنشانی محدودیت IP

گزارش IP انباشته‌شده کلاینت را پاک می‌کند (`POST /clearIps/:email`) تا مسدودیت موقت به دلیل تجاوز از محدودیت اتصالات همزمان برداشته شود. اعلان: «گزارش پاک شد».

حذف

حذف (`POST /del/:email`) — حذف کامل کلاینت. دیالوگ تأیید: عنوان «حذف کلاینت {email}؟»، متن «کلاینت از تمام inboundهای متصل حذف می‌شود و رکورد ترافیک آن نابود می‌شود. این عملیات قابل برگشت نیست.». حذف کلاینت را از **تمام** inboundها جدا کرده و رکورد ترافیکش را نابود می‌کند. اعلان: «کلاینت حذف شد».

8.4. عملیات دسته‌جمعی

در فهرست کلاینت‌ها می‌توان چندین رکورد را علامت زد (**انتخاب همه**، **پاک‌کردن همه**)؛ شمارنده — «{count} انتخاب‌شده». روی موارد انتخاب‌شده موارد زیر در دسترس است:

- **حذف ({count})** (`POST /bulkDel`) — حذف گروهی. تأیید: «حذف {count} کلاینت؟»، «هر کلاینت انتخاب‌شده از تمام inboundهای متصل حذف می‌شود و رکورد ترافیکش نابود می‌شود. این عملیات قابل برگشت نیست.». اعلان: «کلاینت‌های حذف‌شده: {count}»، در صورت شکست جزئی — «حذف‌شده: {ok}، ناموفق: {failed}».
- **ویرایش ({count})** / **تنظیم** (`POST /bulkAdjust`) — تغییر دسته‌جمعی انقضا و/یا سهمیه. دیالوگ «ویرایش {count} کلاینت» با راهنمای «مقادیر مثبت اضافه می‌کنند، مقادیر منفی کم می‌کنند. کلاینت‌های با انقضا یا ترافیک نامحدود برای فیلد مربوطه رد می‌شوند.». فیلدها: **افزودن روز**، **افزودن ترافیک (GB)** و **Set flow**. منطق:
  - **انقضا:** کلاینت‌های با انقضای بی‌پایان (`expiryTime == 0`) رد می‌شوند («unlimited expiry»)؛ برای کلاینت‌های با تاریخ انقضا به تعداد روزهای مشخص‌شده تغییر می‌کند؛ برای کلاینت‌های در حالت «پس از اولین استفاده» (انقضای منفی) مدت انتظار تنظیم می‌شود. کاهشی که از باقیمانده تجاوز کند، رد می‌شود («reduction exceeds remaining time/delay window»).
  - **ترافیک:** کلاینت‌های با بی‌محدودیت (`totalGB == 0`) رد می‌شوند («unlimited traffic»)؛ در غیر این صورت سهمیه به میزان مشخص‌شده تغییر می‌کند، نه کمتر از صفر.
  - **Flow:** لیست کشویی **Set flow** امکان تنظیم یا بازنشانی XTLS flow را برای همه کلاینت‌های انتخاب‌شده به یکباره فراهم می‌کند. به‌طور پیش‌فرض **No change** (بدون تغییر) انتخاب است. گزینه **Disable (clear flow)** flow را بازنشانی می‌کند، و مقادیر `xtls-rprx-vision` و `xtls-rprx-vision-udp443` vision-flow مربوطه را تعیین می‌کنند. تنظیم vision-flow فقط بر inboundهایی که از flow پشتیبانی می‌کنند اعمال می‌شود؛ inboundهای نامناسب بدون تغییر می‌مانند و به‌عنوان ردشده علامت‌گذاری می‌شوند، در حالی که بازنشانی flow همیشه مجاز است.
  - **فعال‌سازی خودکار (از 3.4.2):** کلاینتی که **فقط به‌دلیل اتمام منابع** غیرفعال شده (مدتش منقضی شده یا از سهمیه عبور کرده)، هنگام تمدید به‌طور خودکار دوباره فعال می‌شود — به‌صورت محلی و روی نود او — اگر تنظیم او را به محدوده مجاز بازگرداند. کلاینت‌های دستی غیرفعال‌شده یا همچنان تمام‌شده، غیرفعال باقی می‌مانند (فیلد `enable` اکنون به‌صراحت نوشته می‌شود، بنابراین وضعیت حتی برای کلاینت‌های بدون inbound نیز حفظ می‌شود).
  - اگر نه روز، نه ترافیک و نه flow مشخص شده باشد: «قبل از اعمال، روز، ترافیک یا flow را مشخص کنید.». اعلان: «ویرایش‌شده: {count}» / «ویرایش‌شده: {ok}، ردشده: {skipped}».

مثال: تمدید کلاینت‌های انتخاب‌شده به مدت 30 روز و افزودن 50 GB. در دیالوگ **ویرایش** مقدار **افزودن روز** = `30` ، **افزودن ترافیک (GB)** = `50` را وارد کنید. برای کم کردن یک هفته و کاهش سهمیه به 10 GB، مقادیر منفی وارد کنید: **افزودن روز** = `-7` ، **افزودن ترافیک (GB)** = `-10` (کلاینت‌های با انقضای بی‌پایان یا بی‌محدودیت برای فیلد مربوطه رد می‌شوند).

- **اتصال ({count})** / **قطع اتصال ({count})** (`POST /bulkAttach` / `bulkDetach`) — اتصال/قطع اتصال دسته‌جمعی کلاینت‌های انتخاب‌شده به inboundهای انتخاب‌شده. اهداف — فقط inboundهای چندکاربره. نتیجه قطع اتصال: «قطع‌شده {detached}، ردشده {skipped}.».

- **لینک‌های اشتراک ({count})** — جدول خلاصه URL اشتراک‌ها و اشتراک‌های JSON کلاینت‌های انتخاب‌شده با دکمه **کپی همه**. اگر هیچ‌کدام subId نداشته باشند: «هیچ‌یک از کلاینت‌های انتخاب‌شده شناسه اشتراک ندارند.».

- **افزودن به گروه** و **حذف از گروه** — تعیین و حذف برچسب گروه.

- **فعال کردن ({count})** / **غیرفعال کردن ({count})** (`POST /bulkEnable` / `bulkDisable`) — فعال و غیرفعال کردن دسته‌جمعی کلاینت‌های انتخاب‌شده. **فعال کردن** هر کلاینت انتخاب‌شده را روی تمام inboundهای متصل فعال می‌کند؛ کلاینت‌های با سهمیه ترافیکی تمام‌شده یا انقضای گذشته به‌طور خودکار دوباره غیرفعال می‌شوند. **غیرفعال کردن** بلافاصله دسترسی کلاینت‌ها را قطع می‌کند، اما رکوردها و ترافیک انباشته حفظ می‌شوند. قبل از اجرا پنل تأیید درخواست می‌کند، و پس از عملیات اعلانی با تعداد کلاینت‌های پردازش‌شده و در صورت وجود، تعداد آن‌هایی که عملیات برایشان ناموفق بوده نشان می‌دهد.

بازنشانی ترافیک و حذف بر اساس وضعیت

- **بازنشانی ترافیک همه کلاینت‌ها** (`POST /resetAllTraffics`) — شمارنده‌های `down` / `up` **همه** کلاینت‌های پنل را صفر می‌کند. تأیید: «بازنشانی ترافیک همه کلاینت‌ها؟» و «شمارنده‌های ارسال/دریافت همه کلاینت‌ها به صفر بازنشانی می‌شوند. سهمیه‌ها و تاریخ انقضا تغییر نمی‌کنند. این عملیات قابل برگشت نیست.». اعلان: «ترافیک همه کلاینت‌ها بازنشانی شد».
- **حذف تمام‌شده‌ها** (`POST /delDepleted`) — همه کلاینت‌هایی را که **سهمیه‌شان تمام شده** (`total > 0 and up + down >=` `total`) **یا انقضایشان گذشته** (`expiry_time > 0 and expiry_time <= اکنون`) و `reset = 0` هستند (کلاینت‌های با تمدید خودکار دست‌نخورده می‌مانند) حذف می‌کند. تأیید: «حذف کلاینت‌های تمام‌شده؟»، «تمام کلاینت‌هایی که سهمیه ترافیکی‌شان تمام شده یا انقضایشان گذشته حذف می‌شوند. این عملیات قابل برگشت نیست.». اعلان: «کلاینت‌های تمام‌شده حذف‌شده: {count}».

صادر کردن، وارد کردن و حذف کلاینت‌های بدون اتصال

زمانی که هیچ موردی انتخاب نشده، در منوی **بیشتر** در صفحه **کلاینت‌ها** سه عملیات در دسترس است.

صادر کردن کلاینت‌ها (`GET /clients/export`) یک نمایشگر با فهرست JSON همه کلاینت‌ها در فرمت `{client, inboundIds}` با دکمه‌های کپی و دانلود (فایل `clients-export.json`) باز می‌کند. **وارد کردن کلاینت‌ها** (`POST /clients/import`) یک ویرایشگر باز می‌کند که در آن چنین JSON را وارد می‌کنند و **Import** را می‌زنند: کلاینت‌های دارای `inboundIds` ایجاد شده و به inboundها متصل می‌شوند، کلاینت‌های بدون اتصال به‌عنوان رکوردهای مجرد «خام» بازیابی می‌شوند، و emailهای از قبل موجود **هرگز بازنویسی نمی‌شوند** — آن‌ها به فهرست ردشده‌ها می‌روند. اعلان‌ها: «{ok} imported, {failed} {count}»، «clients imported, skipped».

حذف کلاینت‌های بدون اتصال (`POST /clients/delOrphans`) — عملیات خطرناک: همه کلاینت‌هایی را که به هیچ inbound متصل نیستند، به همراه رکورد ترافیک، گزارش IP و لینک‌های خارجی‌شان حذف می‌کند. تأیید: «Delete clients without an inbound?»، «Removes every client that is not attached to any inbound, along with its traffic record. This cannot be undone.». اعلان: «{count} unattached clients deleted». این عملیات غیرقابل بازگشت است.

8.5. جستجو، فیلتر و مرتب‌سازی

بالای فهرست یک نوار جستجو وجود دارد («جستجوی email، توضیحات، UUID، sub ID، رمز عبور، auth...») — که در email، توضیحات، UUID، subId، رمز عبور و auth جستجو می‌کند. شمارنده نتایج: «نمایش {shown} از {total}».

فهرست کلاینت‌ها به‌طور خودکار به‌روزرسانی می‌شود: پنل هر چند ثانیه یک بار صفحه جاری را دریافت می‌کند، بنابراین کلاینت‌های تازه متصل‌شده و ترتیب مرتب‌سازی تغییریافته بدون به‌روزرسانی دستی ظاهر می‌شوند (نشانگر بارگذاری در هنگام نظرسنجی پس‌زمینه چشمک نمی‌زند).

پنل **فیلتر کلاینت‌ها** امکان انتخاب بر اساس وضعیت (دسته)، پروتکل، inbound متصل، بازه تاریخ انقضا، بازه ترافیک مصرف‌شده، وجود تمدید خودکار (**دارد/ندارد**)، وجود Telegram ID و توضیحات، و همچنین گروه را فراهم می‌کند. در پنل‌های دارای نود، یک انتخاب چندگانه **نودها** ظاهر می‌شود: می‌توان فهرست را به کلاینت‌های نودهای انتخاب‌شده محدود کرد؛ گزینه جداگانه **پنل محلی** کلاینت‌های inboundهای بدون اتصال به نود را انتخاب می‌کند (فیلتر فقط در صورت وجود نود نمایش داده می‌شود). مرتب‌سازی: **قدیمی‌تر/جدیدتر**، **به‌روزشده اخیراً**، **اخیراً آنلاین**، **Email الف→ی / ی→الف**، **ترافیک بیشتر**، **باقیمانده بیشتر**، **انقضای نزدیک‌تر**.

8.6. نشانه‌ها و وضعیت‌ها

اولویت وضعیت‌ها: تمام‌شده/منقضی → غیرفعال → به‌زودی منقضی می‌شود → فعال.

- **آنلاین / آفلاین** — کلاینتی با اتصال فعال (در فهرست آنلاین جاری است) و **فعال** است. فهرست آنلاین با درخواست‌های جداگانه به‌روزرسانی می‌شود (`/onlines` ، `/onlinesByGuid`).

- **تمام‌شده** (depleted) — سهمیه مصرف شده (`up + down >= totalGB`) **یا** انقضا گذشته (`expiryTime <= اکنون`). چنین کلاینتی به‌طور خودکار غیرفعال می‌شود و مشمول **حذف تمام‌شده‌ها** می‌شود.
- **به‌زودی منقضی می‌شود / تمام می‌شود** (expiring) — کلاینت فعالی که تا انقضای انقضا کمتر از آستانه تعریف‌شده مانده **یا** تا تمام شدن سهمیه کمتر از حجم آستانه باقی مانده (آستانه‌ها در تنظیمات پنل تعریف می‌شوند). اگر کلاینت از قبل تمام‌شده/غیرفعال باشد محاسبه نمی‌شود.
- **غیرفعال** (deactive) — کلاینت با `enable = false` (به‌صورت دستی یا توسط کار پس‌زمینه غیرفعال شده).
- **فعال** (active) — فعال است، تمام نشده، انقضا نگذشته و تا آستانه‌ها هنوز فاصله دارد.

---

9. گروه‌های کلاینت

> این یک قابلیت همین فورک 3X-UI است. در پروژه اصلی 3x-ui (MHSanaei) مفهوم «گروه کلاینت» وجود ندارد — اینجا یک جدول مجزای گروه‌ها، صفحه **Группы** در منوی پنل و متدهای API متناظر اضافه شده‌اند. اگر پیکربندی را به 3x-ui اصلی منتقل کنید، برچسب گروه به سادگی در هیچ جایی در نظر گرفته نمی‌شود.

9.1. گروه کلاینت چیست و چرا لازم است

گروه یک برچسب منطقی نام‌گذاری‌شده (label) است که می‌توان آن را روی یک یا چند کلاینت قرار داد. این برچسب نه روش جدیدی برای اتصال ایجاد می‌کند و نه inbound یا نود است — صرفاً یک برچسب سازمانی است که با آن می‌توان کلاینت‌ها را به راحتی فیلتر و به‌صورت گروهی پردازش کرد.

ایده کلیدی مدل کلاینت‌ها در این فورک: **کلاینت یک موجودیت سطح بالاست که با email شناسایی می‌شود** (فیلد `email` در جدول `clients` دارای ایندکس یکتاست). همان کلاینت (یک email با همان اطلاعات حساب) می‌تواند همزمان در چند inbound و حتی روی چند نود قرار داشته باشد، از جمله با پروتکل‌های مختلف. برچسب گروه **یک بار به‌ازای هر کلاینت** ذخیره می‌شود، بنابراین به‌طور خودکار روی همه پیوندهای او به inbound یک‌جا اعمال می‌شود.

برچسب گروه یک برچسب منطقی برای گروه‌بندی است:

لایه	محل ذخیره	فیلد
رکورد کلاینت (پایگاه‌داده)	جدول `clients`	`group_name` (به‌صورت پیش‌فرض رشته خالی `''`)
فهرست گروه‌ها (پایگاه‌داده)	جدول `client_groups`	`name` (ایندکس یکتا، غیرخالی)
تنظیمات inbound (Xray)	JSON `settings.clients[].group`	در هر شیء کلاینت در هر inbound که کلاینت در آن عضو است تکرار می‌شود

چرا این در عمل لازم است:

- **یک کلاینت روی چند inbound/نود.** اگر کلاینت به‌عنوان دسترسی همزمان به چند inbound «فروخته» شود (مثلاً پروتکل‌های مختلف یا نودهای مختلف)، گروه امکان می‌دهد آن را به‌عنوان یک کل واحد مدیریت کنید: ریست ترافیک، حذف، تغییر نام برچسب — با یک عملیات روی همه inboundهای او.
- **عملیات گروهی و فیلترکردن.** در صفحه **Клиенты** می‌توان فهرست را بر اساس گروه فیلتر کرد؛ در صفحه **Группы** اقدامات گروهی روی همه اعضای گروه در دسترس است.
- **سازمان‌دهی مجموعه بزرگی از کلاینت‌ها.** برچسب‌هایی مانند `vip` ، `trial` ، `team-A` کمک می‌کنند هزاران کلاینت را در سبدهای منطقی دسته‌بندی کنید، بدون آنکه inboundهای جداگانه زیادی بسازید.

9.2. ارتباط گروه با کلاینت‌ها، inboundها، نودها و پروتکل‌ها

این مهم‌ترین زیربخش برای درک موضوع است، چون همگام‌سازی برچسب امری بدیهی نیست.

گروه کلاینت را توصیف می‌کند، نه inbound را. برچسب در رکورد کلاینت زندگی می‌کند (`clients.group_name`). وقتی کلاینت به چند inbound متصل باشد، در هر تغییر گروه، پنل از **همه** inboundهایی که این کلاینت در آن‌ها عضو است عبور می‌کند و فیلد `group` را در تنظیمات Xray آن‌ها (`settings.clients[]`) قرار می‌دهد یا حذف می‌کند. از نظر فنی این کار به این صورت انجام می‌شود: بر اساس

email کلاینت همه inboundهایی که او در آن‌ها عضو است یافته می‌شوند، سپس در تنظیمات JSON هر یک از این inboundها شیء کلاینت با همان email ویرایش می‌شود. به همین دلیل:

- گروه **به پروتکل وابسته نیست.** یک email می‌تواند در یک inbound کلاینت VLESS و در inbound دیگری کلاینت Hysteria باشد — برچسب گروه او باز هم یکی است و روی هر دو اعمال می‌شود (اطلاعات حساب در این حال برای هر پروتکل جداست و به‌صورت مجزا ذخیره می‌شود).
- گروه **نودها را در بر می‌گیرد.** inboundهایی که متعلق به نودها هستند با فیلد `nodeId` از inboundهای پنل اصلی متمایز می‌شوند (در inboundهای پنل اصلی این فیلد `null` / `0` است). برچسب گروه روی شیءهای کلاینت در inbound اعمال می‌شود، صرف‌نظر از اینکه آن inbound اصلی است یا نودی — کافی است کلاینتی با همان email آنجا حضور داشته باشد.

برچسب گروه در برابر همگام‌سازی با نودها و بازسازی تنظیمات مقاوم است. این رفتار عمداً تعبیه شده است:

- وقتی نود اسنپ‌شات ترافیک می‌فرستد، داده‌های آن مقادیر محلی `group_name` و `comment` کلاینت را در پایگاه‌داده پنل **بازنویسی نمی‌کنند.** گروه و کامنت فیلدهای «محلی پنل» محسوب می‌شوند — نود آن‌ها را مدیریت نمی‌کند.
- در بازسازی تنظیمات inbound، مقدار خالی `group` در داده‌های ورودی برچسب از پیش ذخیره‌شده را **ریست نمی‌کند.** گروه دقیقاً از طریق صفحه **Группы** مدیریت می‌شود (نه از طریق ویرایش تنظیمات Xrayی inbound)، بنابراین «گروه خالی» در بازسازی معمولی تنظیمات به‌معنای «دست نزن» تفسیر می‌شود، نه «پاک کن».

نتیجه عملی: تنها عملیاتی که **عمداً** برچسب را پاک می‌کنند عبارت‌اند از حذف گروه و حذف صریح کلاینت از گروه (در ادامه ببینید). ویرایش معمولی inbound یا همگام‌سازی پس‌زمینه با نود گروه را از دست نمی‌دهند.

9.3. فهرست گروه‌ها و گروه‌های «خالی»

فهرست گروه‌ها در صفحه از ادغام دو منبع تشکیل می‌شود:

1. **گروه‌های مشتق‌شده (derived)** — همه مقادیر غیرخالی `group_name` که واقعاً نزد کلاینت‌ها وجود دارند، همراه با شمارش تعداد کلاینت‌ها.
2. **گروه‌های ذخیره‌شده (stored)** — رکوردهای جدول `client_groups` .

این ادغام اثر مهمی به همراه دارد: گروه می‌تواند **بدون حتی یک کلاینت** وجود داشته باشد. چنین گروهی با دکمه صریح «Добавить группу» ساخته می‌شود (رکوردی در `client_groups`) و در فهرست با شمارنده `0` نمایش داده می‌شود. همین رکوردها **گروه‌های خالی** محسوب می‌شوند. فهرست همیشه بر اساس نام و بدون توجه به بزرگی و کوچکی حروف مرتب می‌شود.

شمارنده‌های خلاصه در صفحه:

فیلد (RU)	چه چیزی نشان می‌دهد
Всего групп	تعداد کل گروه‌ها (ذخیره‌شده و مشتق‌شده با هم).
Клиенты с группой	چند کلاینت برچسب گروه غیرخالی دارند.
Пустые группы	چند گروه بدون کلاینت وجود دارد (شمارنده `0`).
Клиентов в группе	تعداد کلاینت‌ها در یک گروه مشخص (ستون جدول).

9.4. فیلدها و ستون‌های گروه

رکورد گروه در جدول `client_groups` شامل موارد زیر است:

فیلد	نوع	پیش‌فرض	توضیح
`Id`	int	افزایش خودکار	کلید اصلی رکورد گروه.
`Name`	string	— (الزامی)	نام گروه. ایندکس یکتا، نمی‌تواند خالی باشد. در UI — ستون **Имя**.
`CreatedAt`	int64 (میلی‌ثانیه)	زمان ایجاد	لحظه ایجاد رکورد گروه.
`UpdatedAt`	int64 (میلی‌ثانیه)	زمان تغییر	لحظه آخرین تغییر.

در جدول صفحه دست‌کم ستون‌های **Имя** و **Клиентов в группе** و نیز دکمه‌های اقدام نمایش داده می‌شوند (در ادامه ببینید).

9.5. ایجاد گروه

دکمه **Добавить группу**.

مراحل:

1. روی **Добавить группу** کلیک کنید.
2. نام گروه را وارد کنید.
3. تأیید کنید.

رفتار بک‌اند (`POST /panel/api/clients/groups/create`، بدنه `{"name": "..."}`):

- نام از فاصله‌های ابتدا و انتها پیراسته می‌شود. نام خالی با خطای «group name is required» رد می‌شود.
- اگر گروهی با همین نام از قبل وجود داشته باشد — خطای «group already exists».
- در صورت موفقیت، رکوردی در `client_groups` ساخته می‌شود (در ابتدا بدون کلاینت — یعنی گروه خالی).

پیام موفقیت: **«Группа «{name}» создана.»**

مثال: ساخت گروه خالی از طریق API. مجموعه‌ای از برچسب‌ها را از قبل و پیش از پرکردن با کلاینت‌ها آماده کنید:

```
curl -s 'https://panel.example.com:2053/panel/api/clients/groups/create' \
  -H 'Content-Type: application/json' \
  -b cookie.txt \
  -d '{"name": "vip"}'
```

پاسخ در صورت موفقیت:

```
{ "success": true, "msg": "Группа «vip» создана.", "obj": null }
```

فراخوانی مجدد با همان نام `"success": false` و پیام `group already exists` را برمی‌گرداند.

> ساخت گروه خالی از پیش زمانی مفید است که می‌خواهید مجموعه‌ای از برچسب‌ها را آماده کنید و سپس آن‌ها را از طریق «Добавить клиентов...» با کلاینت‌ها پر کنید.

9.6. تغییر نام گروه

دکمه **Переименовать**، عنوان دیالوگ — **«Переименовать {name}»**.

رفتار (`POST /panel/api/clients/groups/rename` ، بدنه `{"oldName": "...", "newName": "..."}`):

- هر دو نام از فاصله‌ها پیراسته می‌شوند. نام قدیمی خالی — خطای «old group name is required»، نام جدید خالی — «new group name is required».
- اگر نام جدید با نام قدیمی یکی باشد — هیچ کاری انجام نمی‌شود (تعداد کلاینت‌های متأثر `0`).
- در غیر این صورت تغییر نام به‌صورت اتمیک انجام می‌شود:
  - رکورد در `client_groups` تغییر نام می‌یابد؛
  - در همه کلاینت‌هایی که `group_name = oldName` دارند، فیلد به `newName` به‌روزرسانی می‌شود؛
  - در **همه inboundهایی** که کلاینت‌های متأثر در آن‌ها عضو هستند (از جمله نودی)، در تنظیمات Xray مقدار `group` از قدیمی به جدید ویرایش می‌شود.
- پس از تغییر نام، پنل Xray را به‌عنوان نیازمند راه‌اندازی مجدد علامت‌گذاری می‌کند و اعلانی درباره تغییر کلاینت‌ها ارسال می‌کند.

پیام‌ها:

- موفقیت: **«Группа переименована для {count} клиент(ов).»**
- تداخل نام در UI: **«Группа с именем «{name}» уже существует.»**

9.7. افزودن کلاینت‌ها به گروه

دکمه **Добавить клиентов...**، عنوان — **«Добавить клиентов в группу «{name}»».**

راهنمای کلمه‌به‌کلمه در دیالوگ:

> «Выберите клиентов для добавления в эту группу. Существующие привязки к входящим сохраняются; меняется только метка группы. Клиенты, уже входящие в эту группу, не показываются.»

اگر کسی برای افزودن نباشد، **«Нет других клиентов для добавления.»** نمایش داده می‌شود.

رفتار (`POST /panel/api/clients/groups/bulkAdd` ، بدنه `{"emails": [...], "group": "..."}`):

- نام گروه الزامی است (در غیر این صورت خطای «group name is required»)؛ فهرست خالی email — عملیات هیچ کاری انجام نمی‌دهد.
- اگر چنین گروهی هنوز نه در `client_groups` و نه در میان مشتق‌شده‌ها وجود نداشته باشد — به‌صورت خودکار ساخته می‌شود.
- برای emailهای انتخاب‌شده در کلاینت‌ها `group_name = group` قرار می‌گیرد؛ **پیوندهای کلاینت‌ها به inbound تغییر نمی‌کنند** — فقط برچسب متأثر می‌شود. سپس در همه inboundهای این کلاینت‌ها فیلد `group` قرار می‌گیرد.
- تعداد رکوردهای کلاینت متأثر برگردانده می‌شود؛ Xray برای راه‌اندازی مجدد علامت‌گذاری می‌شود.

پیام موفقیت: **«Добавлено {count} клиент(ов) в {name}.»**

مثال: برچسب‌زدن چند کلاینت با یک گروه در یک درخواست. کلاینت‌ها با email مشخص می‌شوند و پیوندهای به inbound در این حال تغییر نمی‌کنند:

```
curl -s 'https://panel.example.com:2053/panel/api/clients/groups/bulkAdd' \
  -H 'Content-Type: application/json' \
  -b cookie.txt \
  -d '{"emails": ["alice@example.com", "bob@example.com"], "group": "vip"}'
```

اگر گروه `vip` هنوز وجود نداشته باشد، به‌صورت خودکار ساخته می‌شود. پس از درخواست، در رکورد این کلاینت‌ها `group_name = "vip"` قرار می‌گیرد و در تنظیمات Xray هر یک از inboundهای آن‌ها، شیء کلاینت فیلد `"group": "vip"` را دریافت می‌کند:

```
{ "id": "6f1b...", "email": "alice@example.com", "group": "vip", "enable": true }
```

9.8. حذف کلاینت‌ها از گروه (بدون حذف خود کلاینت‌ها)

دکمه **Удалить клиентов...**، عنوان — **«Удалить клиентов из группы «{name}»».**

راهنمای کلمه‌به‌کلمه:

> Выберите участников для удаления из этой группы. Сами клиенты сохраняются (используйте «Удалить клиентов группы» для полного удаления).»

رفتار (`POST /panel/api/clients/groups/bulkRemove` ، بدنه `{"emails": [...]}`): از نظر فنی این همان «افزودن به گروه» با گروه خالی است. در کلاینت‌های انتخاب‌شده `group_name` پاک می‌شود و در inboundهای آن‌ها فیلد `group` از تنظیمات Xray حذف می‌شود. خود کلاینت‌ها و پیوندهای آن‌ها به inbound باقی می‌مانند.

پیام موفقیت: **«Удалено {count} клиент(ов) из {name}».**

9.9. ریست ترافیک گروه

دکمه **Сбросить трафик**.

دیالوگ تأیید:

- عنوان: **«Сбросить трафик группы {name}?»**
- متن: **«Сбрасывается только счётчик трафика группы. Счётчики отдельных клиентов не затрагиваются».**

رفتار: **فقط شمارنده نمایشی خود گروه** صفر می‌شود — پنل مجموع فعلی `up/down` گروه را به‌عنوان مقدار پایه به‌خاطر می‌سپارد و سپس در فهرست گروه‌ها `max(0, مجموع − مقدار پایه)` را نشان می‌دهد. شمارنده‌های `up` / `down` ، سهمیه‌ها و فیلد `enable` کلاینت‌های جداگانه **تغییر نمی‌کنند**، و راه‌اندازی مجدد Xray لازم نیست. این تغییری در رفتار نسبت به نسخه‌های قبلی است، که در آن‌ها بازنشانی ترافیک همه کلاینت‌های گروه را صفر می‌کرد.

درخواست: `POST /panel/api/clients/groups/resetTraffic` با بدنه `{"name": "<نام گروه>"}` .

پیام موفقیت: **«Трафик группы {name} сброшен».**

9.10. حذف گروه و حذف کلاینت‌های گروه

در صفحه **دو عملیات حذف کاملاً متفاوت** وجود دارد — آن‌ها به‌راحتی با هم اشتباه گرفته می‌شوند، بنابراین تفاوتشان حیاتی است.

9.10.1. حذف گروه (حفظ کلاینت‌ها)

دکمه **«Удалить группу (сохранить клиентов)».**

دیالوگ:

- عنوان: **«Удалить группу {name}?»**
- متن: **«Это удаляет группу и очищает её метку у {count} клиент(ов). Сами клиенты не удаляются.»**

رفتار (`POST /panel/api/clients/groups/delete` ، بدنه `{"name": "..."}`): رکورد گروه از `client_groups` حذف می‌شود، در همه کلاینت‌های آن `group_name` پاک می‌شود و از inboundهای آن‌ها فیلد `group` حذف می‌شود. **کلاینت‌ها، اتصالات و ترافیک آن‌ها حفظ می‌شوند.** Xray برای راه‌اندازی مجدد علامت‌گذاری می‌شود.

پیام موفقیت: **«Группа очищена у {count} клиент(ов).»**

9.10.2. حذف کلاینت‌های گروه (حذف کامل)

دکمه **«Удалить клиентов группы»**.

دیالوگ:

- عنوان: **«Удалить всех клиентов в {name}?»**
- متن: **«Это безвозвратно удаляет {count} клиент(ов) вместе с их записями трафика. Метка группы также очищается. Это нельзя отменить.»**

این یک عملیات مخرب است: خود کلاینت‌ها را حذف می‌کند (از طریق حذف گروهی بر اساس email، اندپوینت `POST /panel/api/clients/bulkDel`)، از جمله رکوردهای ترافیک آن‌ها، و بدین ترتیب آن‌ها را از همه inboundها حذف می‌کند.

پیام‌ها:

- موفقیت: **«Удалено {count} клиент(ов).»**
- نتیجه جزئی: **«удалено, {failed} пропущено {ok}»**

> اگر گروه خالی باشد، اقدامات روی اعضای آن در دسترس نیست — **«В этой группе пока нет клиентов.»** نمایش داده می‌شود.

9.11. ارتباط با صفحه «Клиенты»

برچسب گروه بیرون از صفحه **Группы** نیز دیده و استفاده می‌شود:

- در رکورد فشرده کلاینت فیلد `group` وجود دارد، بنابراین در فهرست کلاینت‌ها تعلق به گروه نمایش داده می‌شود.
- فهرست کلاینت‌ها (`/panel/api/clients/list/paged`) پارامتر فیلتر `group` را می‌پذیرد: می‌توان یک نام یا چند نام را با کاما جدا کرده ارسال کرد. تطبیق بر اساس اصل «یا» درون فیلد و بدون توجه به بزرگی و کوچکی حروف انجام می‌شود. حالت خاص: عنصر خالی در فهرست گروه‌های فیلتر به‌معنای «کلاینت‌های بدون گروه» است (آن‌هایی که `group` خالی دارند).
- در پاسخ صفحه کلاینت‌ها آرایه `groups` بازگردانده می‌شود — فهرست کامل نام‌های گروه‌های موجود، تا UI بتواند فهرست کشویی فیلتر را بسازد.

مثال: فیلترکردن کلاینت‌ها بر اساس گروه‌ها. درخواست فقط کلاینت‌هایی با برچسب `vip` یا `trial` را برمی‌گرداند (چند نام — با کاما، «یا»):

```
GET /panel/api/clients/list/paged?group=vip,trial
```

برای دریافت کلاینت‌های **بدون** گروه، عنصر خالی را در فهرست ارسال کنید — مثلاً مقدار فیلتر `group=` (رشته خالی) یا `group=vip,` (برچسب `vip` به‌علاوه کلاینت‌های بدون گروه).

9.12. خلاصه اندپوینت‌های API

همه مسیرهای گروه‌ها زیر `/panel/api/clients` نصب شده‌اند:

متد و مسیر	کاربرد	بدنه درخواست
`GET /panel/api/clients/groups`	فهرست گروه‌ها همراه با شمارنده کلاینت‌ها	—
`GET /panel/api/clients/groups/:name/emails`	email همه اعضای گروه (مرتب‌سازی بر اساس email)	—
`POST /panel/api/clients/groups/create`	ساخت گروه خالی	`{"name"}`
`POST /panel/api/clients/groups/rename`	تغییر نام گروه	`{"oldName","newName"}`
`POST /panel/api/clients/groups/delete`	حذف گروه با حفظ کلاینت‌ها (پاک‌کردن برچسب)	`{"name"}`
`POST /panel/api/clients/groups/bulkAdd`	افزودن کلاینت‌ها به گروه (بر اساس email)	`{"emails":[...],"group"}`
`POST /panel/api/clients/groups/bulkRemove`	حذف کلاینت‌ها از گروه (پاک‌کردن برچسب)	`{"emails":[...]}`
`POST /panel/api/clients/bulkDel`	حذف کامل کلاینت‌ها (در «Удалить клиентов группы» استفاده می‌شود)	`{"emails":[...],"keepTraffic"}`

مثال: سناریوی متعارف چرخه عمر گروه از طریق API.

```
# 1. Создать метку trial
curl -s .../panel/api/clients/groups/create   -d '{"name":"trial"}'

# 2. Навесить её на двух клиентов
curl -s .../panel/api/clients/groups/bulkAdd  -d '{"emails":
["u1@example.com","u2@example.com"],"group":"trial"}'

# 3. Обнулить трафик всех участников (по email из /groups/trial/emails)
curl -s .../panel/api/clients/groups/bulkRemove -d '{"emails":["u2@example.com"]}'

# 4. Удалить группу, но сохранить клиентов (только очистка метки)
curl -s .../panel/api/clients/groups/delete   -d '{"name":"trial"}'
```

گام 4 رکورد گروه را حذف می‌کند و `group_name` کلاینت‌های آن را پاک می‌کند، اما خود کلاینت‌ها، اتصالات و ترافیک آن‌ها باقی می‌مانند. برای حذف برگشت‌ناپذیر خود کلاینت‌ها به‌جای آن از `bulkDel` استفاده می‌شود.

عملیاتی که برچسب کلاینت‌ها را تغییر می‌دهند (`rename`، `delete`، `bulkAdd`، `bulkRemove`)، Xray را به‌عنوان نیازمند راه‌اندازی مجدد علامت‌گذاری می‌کنند و اعلانی درباره تغییر کلاینت‌ها ارسال می‌کنند.

9.13. ترافیک به تفکیک گروه

تازگی نسخه 3.3.0: در بخش **Группы** (صفحه «Клиенты»، تب مدیریت گروه‌ها) جدول گروه‌ها اکنون نه‌تنها تعداد کلاینت‌ها در هر گروه، بلکه مجموع ترافیک مصرف‌شده گروه را نیز نشان می‌دهد. ستون با عنوان **«Использованный трафик»** برچسب‌گذاری شده است.

این ستون چه چیزی نشان می‌دهد

برای هر ردیف-گروه، مجموع ترافیک همه کلاینت‌های عضو این گروه نمایش داده می‌شود — یعنی `up + down` (ترافیک ارسالی + دریافتی) همه اعضای آن جمع‌بسته‌شده. این پاسخی سریع به پرسش «در مجموع کل گروه چقدر دانلود/آپلود کرده» می‌دهد، بدون نیاز به بازکردن کلاینت‌ها یکی‌یکی و جمع‌کردن دستی.

در کنار آن در جدول گروه‌ها نمایش داده می‌شوند:

ستون	معنا
Имя	نام گروه
Клиенты	چند کلاینت با این گروه برچسب خورده‌اند (پیش‌تر ستون «Клиентов в группе» نام داشت)
Отправлено	مجموع `up` (ترافیک ارسالی) همه کلاینت‌های گروه
Получено	مجموع `down` (ترافیک دریافتی) همه کلاینت‌های گروه
Использованный трафик	مجموع `up + down` همه کلاینت‌های گروه

ترافیک ارسالی و دریافتی در ستون‌های جداگانه **Отправлено** و **Получено** نمایش داده می‌شوند، و ستون **Использованный трафик** مجموع آن‌ها را نشان می‌دهد. ستون تعداد کلاینت‌ها به‌سادگی **Клиенты** نامیده می‌شود.

خلاصه بالای جدول علاوه بر آن تجمیع‌ها روی همه گروه‌ها را نشان می‌دهد — **«Всего групп»** و **«Клиенты с группой»**، و مجموع ترافیک به دو کارت تقسیم می‌شود: **«Всего отправлено / получено»** (با فلش‌های بالا/پایین — ترافیک ارسالی و دریافتی همه گروه‌ها به‌صورت جداگانه) و **«Общий трафик»** (با آیکون نمودار — جمع کل آن‌ها).

چگونه محاسبه می‌شود

محاسبه با یک کوئری SQL به جدول کلاینت‌ها همراه با اتصال (`LEFT JOIN`) جدول ثبت ترافیک انجام می‌شود:

- بر اساس فیلد-برچسب گروه (`group_name`) کلاینت‌ها گروه‌بندی می‌شوند و تعدادشان شمرده می‌شود — همین «Клиентов в группе» است؛
- ترافیک به‌صورت مجموع `up + down` از جدول متصل‌شده `client_traffics` گرفته می‌شود. یعنی هم بایت‌های ارسالی (`up`) و هم دریافتی (`down`) هر کلاینت جمع‌بسته می‌شوند؛
- از آنجا که email هم در جدول کلاینت‌ها و هم در جدول ترافیک یکتاست، اتصال ترافیک یک کلاینت را دوبرابر نمی‌کند.

ویژگی‌های مقادیر:

- **کلاینت‌های بدون رکورد ترافیک** در شمارنده اعضا لحاظ می‌شوند، اما به مجموع 0 می‌افزایند، بنابراین گروه تازه‌ساخته‌شده ترافیک `0` نشان می‌دهد.
- **گروه‌های خالی** (ساخته‌شده اما بدون کلاینت) نیز در فهرست با شمارنده صفر و ترافیک صفر حضور دارند: علاوه بر گروه‌هایی که از برچسب‌های کلاینت‌ها «استخراج» شده‌اند، گروه‌های صراحتاً ذخیره‌شده نیز به نتیجه افزوده می‌شوند و سپس فهرست بر اساس نام و بدون توجه به بزرگی و کوچکی حروف مرتب می‌شود.
- کلاینت‌های بدون برچسب گروه (`group_name` خالی) در محاسبه وارد نمی‌شوند.

اقدامات مرتبط

از جدول گروه‌ها همچنان اقدامات روی کل گروه در دسترس است، از جمله **«Сбросить трафик» — فقط شمارنده خود گروه** (مقدار پایه) را صفر می‌کند، بدون دست زدن به ترافیک کلاینت‌های جداگانه (به 9.9 مراجعه کنید). پس از چنین ریستی، ستون «Использованный трафик» برای این گروه `0` نشان می‌دهد.

---

10. اشتراک‌ها (Subscription)

اشتراک (subscription) مکانیزمی است که به کاربر یک لینک دائمی (URL) می‌دهد تا کلاینت VPN به‌طور خودکار مجموعه کاملی از تنظیمات را از آن دانلود کرده و به‌صورت دوره‌ای به‌روز کند. به‌جای اینکه لینک هر inbound را جداگانه برای کاربر ارسال کنید، یک آدرس یکتا به شکل `https://دامنه:پورت/sub/<subId>` در اختیار او قرار می‌گیرد. از این آدرس، پنل به‌صورت لحظه‌ای تمام تنظیمات مرتبط با آن کاربر را جمع‌آوری کرده و در قالب مورد نظر کلاینت تحویل می‌دهد. در صورت تغییر تنظیمات سرور (آدرس جدید، چرخش کلیدهای Reality، اضافه شدن inbound)، کاربر در به‌روزرسانی خودکار بعدی تنظیمات جدید را دریافت می‌کند، بدون اینکه نیاز به اقدامی از سوی او باشد.

اشتراک توسط یک سرور HTTP/HTTPS مستقل درون پنل ارائه می‌شود که جدا از پنل وب اجرا می‌شود و روی پورت خودش گوش می‌دهد. این به دلایل امنیتی انجام شده است: پورت اشتراک را می‌توان به بیرون باز کرد، بدون اینکه پورت خود پنل در معرض دسترسی قرار گیرد.

10.1. subId چیست و لینک چگونه ساخته می‌شود

هر کاربر در یک inbound دارای فیلد `subId` (در رابط کاربری — «ID اشتراک») است. این مقدار کلید اشتراک است: پنل در تمام inboundها به دنبال کاربرانی می‌گردد که `subId` آن‌ها با مقدار درخواست‌شده مطابقت دارد و تنظیمات آن‌ها را در یک پاسخ ادغام می‌کند.

- اگر چندین کاربر (در یک یا چند inbound مختلف) `subId` یکسانی داشته باشند، تنظیمات همه آن‌ها در یک اشتراک قرار می‌گیرد. این روش استانداردی است برای اینکه به یک کاربر چندین سرور/پروتکل از طریق یک لینک ارائه شود.

مثال: یک کاربر — دو سرور از یک لینک. فرض کنید دو inbound وجود دارد (VLESS روی سرور A و Trojan روی سرور B). برای اینکه هر دو تنظیمات را با یک لینک به کاربر بدهید، `subId` یکسانی برای هر دو کاربر او تنظیم کنید:

```
Inbound 1 (VLESS):  email = ivan@vpn,  subId = ivan2025
Inbound 2 (Trojan): email = ivan@vpn,  subId = ivan2025
```

در این صورت از آدرس `https://sub.example.com:2096/sub/ivan2025` پنل هر دو تنظیمات را یک‌جا تحویل می‌دهد. اگر بعداً یک inbound سوم با همان `subId` اضافه کنید، کاربر در به‌روزرسانی خودکار بعدی آن را دریافت می‌کند، بدون نیاز به ارسال لینک جدید.

- اگر فیلد `subId` کاربر خالی باشد، امکان اشتراک‌گذاری لینک عمومی وجود ندارد. در رابط کاربری این موضوع با راهنمای زیر نشان داده می‌شود: «این کاربر subId ندارد، لینک اشتراک عمومی در دسترس نیست.»

لینک‌های خارجی و اشتراک‌های کاربر (تب «Links»)

در فرم کاربر، تب **«Links»** وجود دارد که در آن می‌توانید برای یک کاربر خاص، منابع اضافی تنظیمات را پیوست کنید — این منابع فقط در اشتراک همان کاربر گنجانده می‌شوند (قالب‌های JSON، RAW و Clash):

- **Add External Link** — یک لینک اشتراک‌گذاری خارجی (`vless://`، `trojan://`، `ss://` و غیره). عیناً به خروجی اضافه می‌شود، و برای JSON/Clash علاوه بر آن به تنظیمات تبدیل می‌شود.
- **Add External Subscription** — آدرس یک اشتراک خارجی. پنل آن را دانلود می‌کند (با کش‌گذاری و مهلت زمانی کوتاه) و تنظیمات دریافت‌شده را در لیست کلی کاربر ادغام می‌کند.

این قابلیت برای ارائه سرورهای اضافی به کاربر در کنار inboundهای شما از طریق همان لینک واحد مناسب است. اگر پاسخ اشتراک خارجی بیش از حد بزرگ باشد، دیگر بی‌سروصدا کوتاه نمی‌شود: پنل یک خطا برمی‌گرداند و آخرین مقدار موفق کش‌شده را ادامه می‌دهد.

- مقدار `subId` را نمی‌توان دلبخواهی تنظیم کرد: هنگام ذخیره بررسی می‌شود که فاصله، کاراکترهای `/`، `\` و کاراکترهای کنترلی در آن نباشد. راهنمای اعتبارسنجی مربوطه: «ID اشتراک نمی‌تواند شامل فاصله، '/'، " یا کاراکترهای کنترلی باشد».

لینک نهایی به شکل `<پایه>/<subPath>/<subId>` ساخته می‌شود (به بخش تنظیمات سرور اشتراک و فیلد «URI پروکسی معکوس» مراجعه کنید). اگر هیچ کاربری با `subId` مورد نظر پیدا نشود (کاربر حذف شده یا `subId` وجود ندارد)، سرور HTTP 404 بدون بدنه برمی‌گرداند. در صورت خطای داخلی — HTTP 500. کلاینت‌های VPN فقط به کد پاسخ توجه می‌کنند، بنابراین بدنه خطا عمداً خالی است.

ترتیب لینک‌های inbound در اشتراک

هر inbound دارای فیلد **«ترتیب در اشتراک»** (`subSortIndex`) است — عددی از 1 که موقعیت لینک‌های آن inbound در خروجی اشتراک را مشخص می‌کند. مقادیر کمتر اول می‌آیند؛ در صورت مساوی بودن مقادیر، ترتیب اصلی ایجاد (بر اساس id) حفظ می‌شود. ترتیب برای تمام قالب‌های خروجی اعمال می‌شود — متن خام، صفحه اشتراک، JSON و Clash. این فیلد روی ترتیب inboundها در خود پنل تأثیر ندارد.

این فیلد در فرم inbound در کنار تنظیمات آدرس در لینک (share address) قابل ویرایش است و طبق قوانین معمول با نودها همگام‌سازی می‌شود. اگر حداقل یک inbound ترتیبی غیر از 1 داشته باشد، یک ستون فشرده **«ترتیب»** در لیست Inboundها نمایش داده می‌شود.

10.2. تنظیمات سرور اشتراک

تمام پارامترهای اشتراک در بخش تنظیمات پنل، در تب **«اشتراک»** قرار دارند. در ادامه هر پارامتر شرح داده شده است؛ کلید داخلی تنظیمات و مقدار پیش‌فرض در پرانتز آمده.

خود این بخش به تب‌های زیر تقسیم شده است: **«تنظیمات پنل»**، **«اطلاعات»**، **«پروفایل»**، **«گواهینامه‌ها»**، **«Happ»** و **«Clash / Mihomo»**. فیلدهای عنوان اشتراک، URL پشتیبانی، صفحه پروفایل، اطلاعیه و پوشه تم در تب «پروفایل» قرار دارند؛ قوانین مسیریابی Happ و Clash/Mihomo در تب‌های مربوطه؛ و فاصله به‌روزرسانی اشتراک در تب «اطلاعات».

پارامترهای اصلی

فیلد (UI)	کلید	مقدار پیش‌فرض	توضیح
فعال‌سازی اشتراک	`subEnable`	`true` (فعال)	یک سرور اشتراک جداگانه راه‌اندازی می‌کند. راهنما: «قابلیت اشتراک با تنظیمات مستقل». اگر غیرفعال باشد — سرور اشتراک شروع نمی‌شود و هیچ لینکی کار نمی‌کند.
آدرس IP گوش‌دادن	`subListen`	خالی	آدرس IP که سرور اشتراک روی آن اتصالات را می‌پذیرد. راهنما: «برای پیش‌فرض خالی بگذارید تا همه آدرس‌های IP را رصد کند».
پورت اشتراک	`subPort`	`2096`	پورت TCP سرور اشتراک. راهنما: «شماره پورت برای سرویس اشتراک نباید در سرور استفاده شده باشد» — پورت باید آزاد باشد و با پنل یا Xray تداخل نداشته باشد.
مسیر URI	`subPath`	`/sub/`	مسیری که اشتراک‌های معمولی از آن ارائه می‌شوند. راهنما: «باید با '/' شروع شود و با '/' تمام شود».
دامنه گوش‌دادن	`subDomain`	خالی	دامنه‌ای که دسترسی به اشتراک از آن مجاز است (اعتبارسنجی Host). راهنما: «برای پیش‌فرض خالی بگذارید تا همه دامنه‌ها و آدرس‌های IP را بپذیرد». اگر تنظیم شود — درخواست‌هایی با Host متفاوت رد می‌شوند.

نکته امنیتی مهم: مسیر پیش‌فرض `/sub/` (و `/json/` برای JSON) به‌خوبی شناخته‌شده و به‌راحتی قابل حدس زدن است. پنل هشدار می‌دهد: «مسیر پیش‌فرض اشتراک "/sub/" به‌طور گسترده‌ای شناخته شده است — آن را تغییر دهید.» و پیام مشابهی برای JSON. توصیه می‌شود یک مسیر سفارشی غیرقابل حدس تنظیم کنید.

TLS / گواهینامه

فیلد (UI)	کلید	پیش‌فرض	توضیح
مسیر فایل کلید عمومی گواهینامه اشتراک	`subCertFile`	خالی	مسیر کامل فایل گواهینامه (`fullchain / .crt`). راهنما: «مسیر کامل شروع‌شده با '/' را وارد کنید».
مسیر فایل کلید خصوصی گواهینامه اشتراک	`subKeyFile`	خالی	مسیر کامل فایل کلید خصوصی. راهنما: «مسیر کامل شروع‌شده با '/' را وارد کنید».

اگر هر دو مسیر تنظیم شده باشند و گواهینامه با موفقیت بارگذاری شود، سرور اشتراک روی **HTTPS** کار می‌کند. اگر فیلدها خالی باشند یا گواهینامه قابل خواندن نباشد — سرور به **HTTP** بازمی‌گردد (خطا در لاگ نوشته می‌شود). داشتن TLS معتبر همچنین بر ساخت URL پایه تأثیر می‌گذارد: برای پورت 443 با TLS و پورت 80 بدون TLS، شماره پورت در لینک حذف می‌شود.

فاصله به‌روزرسانی

فیلد (UI)	کلید	پیش‌فرض	توضیح
فاصله به‌روزرسانی اشتراک	`subUpdates`	`12`	هر چند ساعت (بر حسب ساعت) برنامه کلاینت باید اشتراک را مجدداً درخواست کند. راهنما: «فاصله بین به‌روزرسانی‌ها در برنامه کلاینت (بر حسب ساعت)».

این مقدار در هدر HTTP پاسخ `Profile-Update-Interval` به کلاینت منتقل می‌شود؛ کلاینت‌های مدرن آن را به عنوان دوره به‌روزرسانی خودکار تنظیمات استفاده می‌کنند.

قالب و اطلاعات در پاسخ

فیلد (UI)	کلید	پیش‌فرض	توضیح
رمزنگاری	`subEncrypt`	`true`	راهنما: «رمزنگاری تنظیمات برگشتی در اشتراک». از نظر فنی این رمزنگاری نیست بلکه **کدگذاری Base64** از کل بدنه اشتراک معمولی است (قالبی که اکثر کلاینت‌ها انتظار دارند). اگر غیرفعال شود، لینک‌ها به‌صورت متن ساده، یکی در هر خط ارائه می‌شوند.
نمایش اطلاعات استفاده	`subShowInfo`	`true`	راهنما: «نمایش ترافیک باقی‌مانده و تاریخ انقضا بعد از نام تنظیمات». وقتی فعال است، به نام (remark) هر تنظیمات نشانگرهای ترافیک باقی‌مانده ( ) و مدت اعتبار (مثلاً `5D,3H`) اضافه می‌شود؛ برای کاربران منقضی/غیرقابل دسترس `N/A` نشان داده می‌شود.
افزودن Email به نام	`subEmailInRemark`	`true`	راهنما: «افزودن ایمیل کاربر به نام پروفایل اشتراک». ایمیل کاربر را به remark پروفایل اضافه می‌کند.

قالب نام‌نمایشی (Remark Template)

نام نمایشی (remark) هر تنظیمات در اشتراک بر اساس **قالب نام‌نمایشی** ساخته می‌شود — فیلد **«قالب یادداشت»** (`remarkTemplate`) در تب **«اطلاعات»** از تنظیمات اشتراک. سازنده مدل یادداشت قبلی (انتخاب جداگانه بخش‌های inbound/email/external proxy و کاراکتر جداکننده) از رابط کاربری حذف شده است؛ اکنون شما یک قالب نام دلخواه می‌نویسید و متغیرها را در آن قرار می‌دهید. مقدار پیش‌فرض — `{{INBOUND}}-{{EMAIL}}| {{TRAFFIC_LEFT}}| {{DAYS_LEFT}}D` (یعنی به‌طور پیش‌فرض نام پروفایل شامل ایمیل کاربر است). اگر فیلد خالی بماند، مدل یادداشت قدیمی (غیرقابل پیکربندی از طریق رابط) استفاده می‌شود.

متغیرها در بخش‌های **Client**، **Traffic** و **Time & status** گروه‌بندی شده‌اند و در کنار فیلد به‌صورت تراشه‌های قابل کلیک `{{VAR}}` با راهنمای ماوس نشان داده می‌شوند؛ با کلیک، توکن در قالب درج می‌شود، پیش‌نمایش زنده نیز در دسترس است. هر متغیر به‌صورت جداگانه برای

کاربر خاص در لحظه تولید اشتراک جایگزین می‌شود. نوشتار ساده‌تر در پرانتزهای تکی ({DATA_LEFT} ، {EXPIRE_DATE} ، {PROTOCOL} ، {TRANSPORT} و غیره) نیز پذیرفته می‌شود — پنل آن را به قالب داخلی {{...}} تبدیل می‌کند.

متغیرهای موجود:

- **شناسه کاربر:** {{EMAIL}} (مترادف — {{USERNAME}})، {{INBOUND}} (remark خود inbound)، {{HOST}} (remark هاست)، {{ID}} (UUID)، {{SHORT_ID}} (8 کاراکتر اول UUID)، {{SUB_ID}} ، {{COMMENT}} ، {{TELEGRAM_ID}} ، {{PROTOCOL}} ، {{TRANSPORT}} .
- **ترافیک:** {{TRAFFIC_USED}} ، {{TRAFFIC_LEFT}} ، {{TRAFFIC_TOTAL}} (و نسخه‌های *_BYTES آن‌ها در بایت دقیق)، {{UP}} ، {{DOWN}} ، {{USAGE_PERCENTAGE}} .
- **مدت اعتبار و وضعیت:** {{DAYS_LEFT}} ، {{TIME_LEFT}} ، {{EXPIRE_DATE}} (YYYY-MM-DD)، {{JALALI_EXPIRE_DATE}} (تاریخ به تقویم جلالی)، {{EXPIRE_UNIX}} ، {{CREATED_UNIX}} ، {{RESET_DAYS}} ، {{STATUS}} (active / expired / disabled / depleted)، {{STATUS_EMOJI}} .
- **اتصال (Connection):** {{PROTOCOL}} — پروتکل (Trojan، VMess، VLESS و غیره)، {{TRANSPORT}} — شبکه حمل‌ونقل (grpc، ws، tcp و غیره)، {{SECURITY}} — امنیت حمل‌ونقل (NONE، REALITY، TLS؛ با حروف بزرگ نمایش داده می‌شود).

مانند متغیرهای مصرف و مدت اعتبار، این سه متغیر فقط در بدنه اشتراک عمل می‌کنند و به‌طور خودکار از remark لینک‌های نمایشی در پنل (QR/«اطلاعات») و صفحه اطلاعات اشتراک حذف می‌شوند.

قالب را می‌توان با خط عمودی | به بخش‌هایی تقسیم کرد. بخشی که متغیر در آن مقدار «بدون محدودیت» (∞) تولید کند — مثلاً {{TRAFFIC_LEFT}} یا {{DAYS_LEFT}} برای کاربر بدون محدودیت — به‌طور خودکار پنهان می‌شود. علاوه بر این، بلوک مصرف ترافیک و مدت اعتبار فقط یک‌بار، در اولین لینک کاربر نمایش داده می‌شود تا در هر تنظیمات تکرار نشود.

مثال. قالب {{EMAIL}} | {{TRAFFIC_LEFT}} | {{DAYS_LEFT}}D برای کاربری با 42 گیگابایت باقی‌مانده و 7 روز، نامی مانند ivan@vpn 42.00GB 7D و برای کاربر بدون محدودیت فقط ivan@vpn (بخش‌های با ∞ حذف شده‌اند) تولید می‌کند.

از نسخه 3.4.2 متغیر {{EMAIL}} (و مترادف {{USERNAME}}) فقط روی **اولین** لینک کلاینت در بدنه اشتراک نمایش داده می‌شود — مانند بلوک باقی‌مانده ترافیک/مدت؛ روی بقیه لینک‌های همان کلاینت حذف می‌شود.

در لینک‌های نمایشی پنل (QR-کد و پنجره‌های «اطلاعات» در صفحه «کلاینت‌ها») و صفحه اطلاعات اشتراک، ایمیل کاربر در نام پروفایل وجود دارد: به شکل «inbound-host-email» اگر هاست تنظیم شده باشد، یا «inbound-email» بدون هاست. اطلاعات ترافیک و مدت اعتبار (و همچنین متغیرهای گروه «اتصال») در این نام‌های نمایشی جایگزین نمی‌شوند — آن‌ها فقط در بدنه اشتراکی که کلاینت VPN دریافت می‌کند عمل می‌کنند.

اگر ردیف آماری ترافیک کاربر پس از حذف و بازسازی inbound «یتیم» شده باشد، متغیر {{TRAFFIC_USED}} (و سایر شاخص‌های مصرف) دیگر 0.00B نشان نمی‌دهد: پنل آمار را با ایمیل کاربر جستجو می‌کند و ترافیک مصرف‌شده صحیح را جایگزین می‌کند. | قالب نام‌نمایشی | remarkTemplate | {{DAYS_LEFT}}D | {{TRAFFIC_LEFT}} | {{INBOUND}} | قالب آزاد نام نمایشی (remark) هر تنظیمات با جایگزینی متغیرهای {{VAR}} . برای هر کاربر به‌صورت جداگانه در زمان تولید اشتراک جایگزین می‌شود. سازنده «مدل یادداشت» قبلی (انتخاب inbound/email/external proxy و جداکننده) از رابط کاربری حذف شده و فقط به‌عنوان گزینه پشتیبان در صورت خالی گذاشتن فیلد استفاده می‌شود. برای اطلاعات بیشتر — به «قالب نام‌نمایشی (Remark Template)» مراجعه کنید. |

متادیتای پروفایل (هدرهای پاسخ)

این رشته‌ها در هدرهای HTTP پاسخ به کلاینت منتقل می‌شوند و در کلاینت VPN به‌عنوان متادیتای پروفایل نمایش داده می‌شوند. همه آن‌ها به‌طور پیش‌فرض خالی هستند.

فیلد (UI)	کلید	هدر	توضیح
عنوان اشتراک	subTitle	Profile-Title (به Base64)	«نام اشتراکی که کاربر در کلاینت VPN می‌بیند». برای Clash همچنین به‌عنوان نام پروفایل واردشده از طریق Content-Disposition استفاده می‌شود.

فیلد (UI)	کلید	هدر	توضیح
URL پشتیبانی	`subSupportUrl`	`Support-Url`	«لینک پشتیبانی فنی که در کلاینت VPN نمایش داده می‌شود».
URL پروفایل	`subProfileUrl`	`Profile-Web-Page-Url`	«لینک وب‌سایت شما که در کلاینت VPN نمایش داده می‌شود». اگر تنظیم نشود، URL واقعی درخواست اشتراک جایگزین می‌شود.
اطلاعیه	`subAnnounce`	`Announce` (به Base64)	«متن اطلاعیه که در کلاینت VPN نمایش داده می‌شود».

علاوه بر این، در هر پاسخ هدر `Subscription-Userinfo` با داده‌های ترافیک کلی کاربر ارسال می‌شود: `upload` ، `download` ، `total` و `expire` (لحظه انقضا به ثانیه). از طریق این هدر کلاینت ترافیک باقی‌مانده و تاریخ انقضا را نمایش می‌دهد.

مسیریابی (فقط برای کلاینت Happ)

فیلد (UI)	کلید	پیش‌فرض	توضیح
فعال‌سازی مسیریابی	`subEnableRouting`	`false`	«تنظیم سراسری برای فعال‌سازی مسیریابی در کلاینت VPN. (فقط برای Happ)». در هدر `Routing-Enable` منتقل می‌شود.
قوانین مسیریابی	`subRoutingRules`	خالی	«قوانین مسیریابی سراسری برای کلاینت VPN. (فقط برای Happ)». در هدر `Routing` منتقل می‌شود.

| پنهان کردن تنظیمات سرور | `subHideSettings` | `false` | «پنهان کردن تنظیمات سرور در اشتراک (فقط برای Happ)». وقتی فعال است، در کلاینت Happ امکان مشاهده و تغییر پارامترهای سرور پنهان می‌شود. این گزینه فقط برای کلاینت Happ عمل می‌کند. |

مسیریابی Incy (فقط برای کلاینت Incy)

برای کلاینت VPN **Incy**، در تنظیمات اشتراک یک تب جداگانه **«Incy»** با دو فیلد وجود دارد: کلید **«فعال‌سازی مسیریابی»** (`subIncyEnableRouting` ، به‌طور پیش‌فرض غیرفعال) و فیلد متنی **«قوانین مسیریابی»** (`subIncyRoutingRules`) به قالب `incy://routing/onadd/<base64>` . وقتی مسیریابی فعال و فیلد پر شده باشد، این رشته به‌عنوان یک خط جداگانه به بدنه اشتراک (قالب raw) اضافه می‌شود — بدین ترتیب پروفایل مسیریابی به کلاینت Incy تحویل داده می‌شود، بدون اینکه با هدر `Routing` کلاینت Happ تداخل داشته باشد. این تنظیمات فقط برای کلاینت Incy عمل می‌کنند.

URI پروکسی معکوس

فیلد (UI)	کلید	پیش‌فرض	توضیح
URI پروکسی معکوس	`subURI`	خالی	«تغییر URI پایه آدرس اشتراک برای استفاده پشت پروکسی‌سرورها».

اگر فیلد خالی باشد، پنل آدرس پایه لینک را خودش از دامنه و پورت اشتراک (با توجه به TLS) می‌سازد. اگر اشتراک از طریق یک ریورس‌پروکسی/CDN خارجی روی دامنه یا مسیر دیگری ارائه شود، آدرس URI نهایی پایه در این فیلد تنظیم می‌شود و همه لینک‌ها از آن ساخته می‌شوند. فیلدهای مستقل مشابه برای JSON (`subJsonURI`) و Clash (`subClashURI`) نیز وجود دارند.

اگر فقط `subURI` عمومی تنظیم شده باشد و فیلدهای جداگانه برای JSON و Clash خالی مانده باشند، لینک‌های این قالب‌ها در صفحه اشتراک طرح و هاست را از `subURI` به ارث می‌برند (نه پورت sub-server و `http`) — بدین ترتیب با آدرس ریورس‌پروکسی مطابقت دارند.

مثال: اشتراک پشت ریورس‌پروکسی. خود اشتراک روی `2096` گوش می‌دهد، اما از بیرون از طریق nginx/CDN روی `https://cfg.example.com/u/` قابل دسترسی است. برای اینکه لینک‌های پاسخ از آدرس خارجی ساخته شوند نه از آدرس داخلی `دامنه:2096`، در فیلد «Reverse proxy URI» «URI» پایه نهایی تنظیم می‌شود:

```
Reverse proxy URI: https://cfg.example.com/u
```

در این صورت لینک نهایی به شکل `https://cfg.example.com/u/ivan2025` خواهد بود. برای قالب‌های JSON و Clash در صورت نیاز، فیلدهای جداگانه `subJsonURI` و `subClashURI` به همین روش پر می‌شوند.

10.3. قالب‌های خروجی

اشتراک می‌تواند در سه قالب مستقل ارائه شود، هر کدام با اندپوینت خود که می‌توان آن را جداگانه فعال/غیرفعال کرد.

آدرس سرور و نودها در خروجی

آدرس سرور در لینک‌های اشتراک بر اساس همان استراتژی آدرس در لینک جایگزین می‌شود که برای لینک‌ها و QR-کدهای معمولی پنل استفاده می‌شود: «listen» — آدرس bind قابل مسیریابی، «custom» — آدرس سفارشی تنظیم‌شده توسط کاربر (`shareAddr`)، «node» (پیش‌فرض) — آدرس نود. برای inboundهایی که استراتژی صریحی ندارند، خروجی اشتراک تغییر نمی‌کند. این امکان می‌دهد یک inbound متعلق به نود که به یک IP عمومی خاص متصل است، آدرس قابل دسترسی را به کلاینت‌ها تحویل دهد. استراتژی برای قالب‌های JSON، raw و Clash اعمال می‌شود.

نام نود (Node) به نام (remark) پروفایل در اشتراک اضافه نمی‌شود: در برنامه کلاینت فقط remark inbound تنظیم‌شده توسط مدیر نمایش داده می‌شود، بدون پسوند داخلی به شکل `نام-نود@`. برای تمایز رکوردهای هم‌نام در اشتراک چندنودی، remarkهای متفاوتی به آن‌ها به‌صورت دستی تنظیم کنید یا از هاست‌های مدیریت‌شده (Hosts) با Remark خود استفاده کنید.

اگر به دلیل عدم همگام‌سازی بین نودها، یک کاربر دو بار در inbound JSON خدماتی قرار گرفته باشد، خروجی اشتراک به‌طور خودکار چنین تکراری‌هایی را بر اساس email در هر سه قالب حذف می‌کند، بنابراین پروفایل‌های تکراری در خروجی ظاهر نمی‌شوند.

هاست‌های مدیریت‌شده (Hosts)

بخش **Hosts** (آیتم منوی کناری؛ صفحه خلاصه با تعداد Total/Enabled/Disabled و لیست) جایگزینی آدرس برای لینک‌های اشتراک را تعریف می‌کند. برای هر inbound می‌توان یک یا چند **هاست** اضافه کرد — اندپوینت‌هایی که **به‌جای آدرس، پورت و پارامترهای TLS خود inbound** در لینک‌های اشتراک تحویل‌داده‌شده به کلاینت جایگزین می‌شوند. این برای توزیع ترافیک از طریق CDN یا رله، بدون تغییر خود inbound، مناسب است.

برای هر هاست موارد زیر تنظیم می‌شود:

- **Remark** و توضیح (Description)، اتصال به **Inbound** خاص، کلید **Enable** و انتساب به نودها (**Nodes**).
- **Address** (خالی — آدرس inbound به ارث می‌رسد) و **Port** (`0` — پورت inbound به ارث می‌رسد)؛ **Tags** (فقط در اشتراک RAW در نظر گرفته می‌شوند).
- تب **Security** — `same` / `tls` / `none` / `reality` با SNI، اثرانگشت (fingerprint)، ALPN، گواهینامه (pinned) pinned-cert)، `allowInsecure` و ECH.
- تب **Advanced** — هدر Host، Path، مسیر Final Mask، Sockopt، Mux، VLESS و حذف هاست از قالب‌های اشتراک خاص (raw / json / clash).
- تب **Clash (mihomo)** — نسخه Mihomo X25519، IP، درهم‌زدن هاست‌ها (Shuffle host).

هاست‌ها در محدوده inbound خود مرتب‌سازی می‌شوند و از فعال‌سازی، غیرفعال‌سازی و حذف دسته‌جمعی پشتیبانی می‌کنند. هاست‌های مدیریت‌شده جایگزین آرایه External Proxy قبلی می‌شوند.

مسیر VLESS (VLESS route). از نسخه 3.4.2 این یک عدد واحد `0–65535` است (نه فهرستی از پورت‌ها؛ راهنما — «یک مقدار واحد VLESS route (0-65535) که در UUID دوخته می‌شود، مثلاً 443؛ خالی — بدون آن»، متن راهنما `443`). مقدار تعیین‌شده واقعاً در UUID هر اشتراک تولیدشده «دوخته» می‌شود (raw / JSON / Clash): Xray بایت‌های 6-7 از UUID را می‌خواند و قبل از احراز هویت آن‌ها را ماسک می‌کند، بنابراین کلاینت همچنان مطابقت دارد. مقدار خالی یا نامعتبر، UUID را تغییر نمی‌دهد.

لینک‌های معمولی (SUB) — Base64 / متن ساده

قالب پایه، اندپوینت `subPath` (به‌طور پیش‌فرض `/sub/`). همیشه فعال است (وقتی اشتراک به‌طور کلی فعال باشد). لیستی از لینک‌های Xray (`vless://`، `vmess://`، `trojan://`، `ss://` و غیره) را برمی‌گرداند — یکی در هر خط. اگر گزینه «رمزنگاری» (`subEncrypt`) فعال باشد، کل لیست به Base64 کدگذاری می‌شود؛ اگر غیرفعال باشد — به‌صورت متن ساده ارائه می‌شود. این قالب را تقریباً همه کلاینت‌ها می‌فهمند (Happ، Shadowrocket، Streisand، NekoBox، Sing-box، V2RayTun، v2rayNG و غیره).

مثال: بدنه پاسخ با «رمزنگاری» غیرفعال. با `subEncrypt = false`، اندپوینت `/sub/` متن ساده ارائه می‌دهد — یک لینک در هر خط:

```
vless://3c8f...@a.example.com:443?security=reality&...#srvA-ivan
trojan://p4ss@b.example.com:443?security=tls&...#srvB-ivan
```

با `subEncrypt = true` (پیش‌فرض) همان لیست به‌طور کامل به Base64 کدگذاری می‌شود و به‌صورت یک رشته ارائه می‌شود — این همان چیزی است که اکثر کلاینت‌ها انتظار دارند.

اشتراک JSON (sing-box و سازگار)

اندپوینت `subJsonPath` (به‌طور پیش‌فرض `/json/`)، با یک تیک جداگانه فعال می‌شود.

فیلد (UI)	کلید	پیش‌فرض	توضیح
اشتراک JSON	`subJsonEnable`	`false`	«فعال/غیرفعال کردن اندپوینت JSON اشتراک به‌صورت مستقل.».

یک تنظیمات JSON کامل برمی‌گرداند (قالب قابل فهم برای sing-box و کلاینت‌های مشتق — Karing، sing-box، OpenWRT، Podkop، NekoBox). برای این قالب پارامترهای اضافی در دسترس هستند (تب `subFormats`):

- **Mux** (`subJsonMux`، به‌طور پیش‌فرض خالی) — تنظیمات JSON مالتی‌پلکسینگ (Mux) که در outbound هر جریان اشتراک JSON تزریق می‌شوند. «انتقال چندین جریان داده مستقل در یک اتصال.».
- **Final Mask** (`subJsonFinalMask`، به‌طور پیش‌فرض خالی) — «ماسک‌های finalmask xray (TCP/UDP) و تنظیمات QUIC که به هر جریان اشتراک JSON اضافه می‌شوند. نسخه جدید xray روی کلاینت لازم است.». از طریق زیرفیلدها تنظیم می‌شود: «بسته‌ها» (`packets`)، «طول» (`length`)، «فاصله» (`interval`)، «حداکثر تقسیم» (`maxSplit`)، «نویزها» (`noises`: «نوع»/`type`، «بسته»/`packet`، «تأخیر (ms)»/`delayMs`، «اعمال روی»/`applyTo`، دکمه «+ نویز»)، و همچنین «همزمانی» (`concurrency`)، «همزمانی xudp» (`xudpConcurrency`) و «xudp UDP 443» (`xudpUdp443`).
- **قوانین مسیریابی** (`subJsonRules`، به‌طور پیش‌فرض خالی) — قوانین سراسری که به تنظیمات JSON اضافه می‌شوند.

اشتراک Clash / Mihomo (YAML)

اندپوینت `subClashPath` (به‌طور پیش‌فرض `/clash/`)، با یک تیک جداگانه فعال می‌شود.

فیلد (UI)	کلید	پیش‌فرض	توضیح
اشتراک Clash / Mihomo	`subClashEnable`	`false`	تولید تنظیمات YAML برای کلاینت‌های Clash و Mihomo را فعال می‌کند.
فعال‌سازی مسیریابی	`subClashEnableRouting`	`false`	«افزودن قوانین مسیریابی سراسری Clash/Mihomo به اشتراک‌های YAML تولیدشده.».
قوانین مسیریابی سراسری	`subClashRules`	خالی	«قوانین Clash/Mihomo که در ابتدای هر اشتراک YAML قبل از MATCH,PROXY اضافه می‌شوند.».

پاسخ با نوع `application/yaml; charset=utf-8` ارائه می‌شود. اگر «عنوان اشتراک» (`subTitle`) تنظیم شده باشد، در هدر `Content-Disposition` نیز منتقل می‌شود (`attachment; filename*=UTF-8''<title>`) تا کلاینت Clash پروفایل واردشده را با این نام نام‌گذاری کند.

قالب لینک‌ها و YAML تولیدشده برای کلاینت‌های مدرن به‌روز نگه داشته می‌شود: Shadowsocks-2022 (SS2022) دیگر userinfo را به Base64 کدگذاری نمی‌کند؛ لینک‌های Shadowsocks با obfuscation http در قالب SIP002 با پلاگین `obfs-local` ارائه می‌شوند؛ برای اشتراک‌های Clash/Mihomo مجموعه کامل فیلدهای XHTTP پیاده‌سازی شده است. تنظیمات جداگانه‌ای لازم نیست — لینک‌ها فقط توسط کلاینت‌ها بهتر شناخته می‌شوند.

> توجه: در این نسخه دقیقاً سه قالب پشتیبانی می‌شوند — لینک‌های معمولی (Base64/متن)، JSON (سازگار با sing-box) و Clash/Mihomo (YAML). قالب جداگانه Outline در سرور اشتراک وجود ندارد.

10.4. صفحه اطلاعات اشتراک و QR-کدها

اگر لینک اشتراک را در مرورگر باز کنید (یا پارامتر `?html=1` یا `?view=html` را به URL اضافه کنید، یا هدر `Accept: text/html` ارسال کنید)، سرور به‌جای پاسخ «خام»، یک **صفحه بصری اطلاعات اشتراک** («اطلاعات اشتراک») ارائه می‌دهد. کلاینت‌های VPN همچنان پاسخ ماشینی دریافت می‌کنند، زیرا آن‌ها HTML درخواست نمی‌کنند.

صفحه (یک برنامه تک‌صفحه‌ای ساخته‌شده با Vite) موارد زیر را نشان می‌دهد:

- **اطلاعات اشتراک** (بلوک Descriptions):
  - «ID اشتراک» — مقدار `subId` ؛
  - «وضعیت» — «فعال»، «غیرفعال» یا «بدون محدودیت». وضعیت «غیرفعال» زمانی تنظیم می‌شود که کاربر غیرفعال شده، محدودیت ترافیک را تمام کرده یا تاریخ انقضا گذشته باشد؛
  - «دانلود» و «آپلود» — حجم ترافیک؛
  - «محدودیت کل» — محدودیت ترافیک یا `∞` اگر محدود نباشد؛
  - «تاریخ انقضا» — تاریخ پایان یا «بدون محدودیت»؛
  - ترافیک باقی‌مانده و زمان آخرین آنلاین.
  - تاریخ‌ها بر اساس تقویم میلادی یا جلالی نمایش داده می‌شوند، بسته به تنظیم «نوع تقویم» پنل (`datepicker` ، پیش‌فرض `gregorian`).
- **لینک‌های اشتراک**: برای هر قالب فعال — یک ردیف جداگانه با تگ رنگی (سبز **SUB**، بنفش **JSON**، طلایی **CLASH**)، دکمه کپی و دکمه **QR-کد** (پنجره پاپ‌آپ، اندازه 240 px). ردیف‌های JSON و CLASH فقط اگر قالب مربوطه در تنظیمات فعال باشد نمایش داده می‌شوند.
- **لینک‌های فردی** («کپی لینک»): لیست کامل تنظیمات جداگانه‌ای که در اشتراک قرار دارند، هر کدام با تگ پروتکل، دکمه کپی و QR-کد (برای لینک‌های QR، post-quantum ساخته نمی‌شود).

- **دکمه «کپی همه تنظیمات»** (بالای لیست لینک‌های فردی): با یک کلیک همه لینک‌های تنظیمات را (هر کدام در خطی جداگانه) در کلیپ‌بورد کپی می‌کند، بدون نیاز به کپی کردن یکی‌یکی؛ پس از اتمام اعلان «همه تنظیمات کپی شدند» نمایش داده می‌شود.

- **دکمه‌های وارد کردن سریع به برنامه‌ها** (منوهای کشویی بر اساس پلتفرم): برای Android — v2box، v2rayNG (deep-link `v2rayng://install-config?url=…`)، Happ، NPV Tunnel، V2RayTun، Sing-box (`happ://add/…`)، Incy (`incy://add/…`)؛ برای iOS — Shadowrocket (از طریق پارامتر `flag=shadowrocket`)، v2box (`v2box://install-sub?url=…&name=…`)، Streisand (`streisand://import/…`)، Incy، Happ، NPV Tunnel، V2RayTun.
  این دکمه‌ها یا deep-link برنامه مورد نظر را با آدرس اشتراک از پیش پر شده باز می‌کنند، یا لینک را در کلیپ‌بورد کپی می‌کنند.

صفحه اطلاعات با هدرهای جلوگیری از کش (`Cache-Control: no-cache`) ارائه می‌شود تا کاربر همیشه داده‌های به‌روز ترافیک و تاریخ انقضا را ببیند.

10.5. قالب‌های سفارشی صفحه اشتراک

از نسخه 3.3.0 می‌توانید صفحه لندینگ استاندارد اشتراک را با قالب HTML خودتان جایگزین کنید. به‌طور پیش‌فرض، صفحه داخلی از آدرس اشتراک ارائه می‌شود، اما اگر پوشه‌ای با قالب خودتان مشخص کنید، پنل آن را رندر کرده و داده‌های به‌روز کاربر (ترافیک، تاریخ انقضا، لینک‌ها و غیره) را در آن جای می‌دهد.

مهم: پنل **هیچ قالب آماده‌ای ارائه نمی‌دهد** — تم خود را باید به‌تنهایی بسازید. دستورالعمل ساخت و فهرست متغیرهای موجود در `docs/custom-subscription-templates.md` آمده است.

کجا فعال می‌شود

پوشه تم در تنظیمات پنل مشخص می‌شود:

تنظیمات → **اشتراک** → **بخش «اطلاعات اشتراک»**، فیلد **«پوشه تم اشتراک»** (`subThemeDir`).

توضیح فیلد در رابط کاربری: «مسیر مطلق به پوشه با قالب سفارشی (index.html/sub.html) برای صفحه اشتراک (مثلاً /etc/3x-ui/sub_templates/my-theme/). برای استفاده از صفحه پیش‌فرض خالی بگذارید.»

در همان بخش تنظیمات مرتبطی قرار دارند که مقادیر آن‌ها در قالب در دسترس است:

در توضیح فیلد «پوشه تم اشتراک» لینک **«راهنمای قالب‌ها ↗»** به مستندات ساخت قالب‌های اشتراک سفارشی وجود دارد.

- **«عنوان اشتراک»** (`subTitle`) — نامی که کاربر می‌بیند؛
- **«URL پشتیبانی»** (`subSupportUrl`) — لینک پشتیبانی فنی.

پارامتر تنظیمات

پارامتر	مقدار پیش‌فرض	هدف
`subThemeDir`	`""` (خالی)	مسیر مطلق به پوشه با قالب HTML شما. خالی = صفحه داخلی پیش‌فرض.

نحوه استفاده از قالب سفارشی

1. یک پوشه برای تم روی سرور بسازید (هر جایی)، مثلاً `/etc/3x-ui/sub_templates/my-theme/`.
2. یک فایل HTML با نام `index.html` یا `sub.html` داخل آن قرار دهید.

مثال: مسیر به تم. چیدمان نهایی روی سرور و مقدار فیلد در تنظیمات:

```
/etc/3x-ui/sub_templates/my-theme/
└── index.html        (اولویت دارد — sub.html یا)
```

```
تنظیمات → اشتراک → «پوشه تم اشتراک»:
/etc/3x-ui/sub_templates/my-theme/
```

مسیر باید **مطلق** باشد (با `/` شروع شود). اگر نه `index.html` و نه `sub.html` در پوشه نباشد، پنل صفحه داخلی را ارائه می‌دهد. 3. در پنل، **تنظیمات** → **اشتراک** را باز کنید و مسیر **مطلق** به این پوشه را در فیلد «پوشه تم اشتراک» وارد کنید. 4. تنظیمات را ذخیره کنید.

رفتار انتخاب فایل و رندرینگ:

- اگر `sub.html` در پوشه باشد، همان استفاده می‌شود؛ در غیر این صورت `index.html` گرفته می‌شود. یعنی `sub.html` بر `index.html` اولویت دارد.
- قالب با موتور استاندارد Go به نام `html/template` رندر می‌شود.
- قالب تجزیه‌شده **کش می‌شود** و فقط در صورت تغییر زمان اصلاح فایل از دیسک خوانده می‌شود. بنابراین ویرایش قالب بدون راه‌اندازی مجدد پنل اعمال می‌شود، اما بدون سربار خواندن/تجزیه در هر درخواست.
- پاسخ به‌طور کامل در بافر ساخته می‌شود و سپس به کاربر ارسال می‌شود: اگر قالب در حین اجرا خطا داشته باشد، صفحه نیمه‌تولیدشده (خراب) به کاربر نمی‌رسد.

رفتار پیش‌فرض و بازگشت (fallback)

- فیلد خالی است → صفحه SPA داخلی ارائه می‌شود (داده‌ها در `window.__SUB_PAGE_DATA__` تزریق می‌شوند).
- مسیر وجود ندارد یا پوشه نیست → صفحه پیش‌فرض استفاده می‌شود.
- نه `index.html` و نه `sub.html` در پوشه نیست → هشدار «subThemeDir set but no usable template found» در لاگ نوشته می‌شود، صفحه پیش‌فرض ارائه می‌شود.
- فایل قالب وجود دارد اما قابل تجزیه نیست → خطای «custom template parse failed» در لاگ نوشته می‌شود، صفحه پیش‌فرض ارائه می‌شود.
- خطا در اجرای قالب → «custom template execution failed» در لاگ نوشته می‌شود، صفحه پیش‌فرض ارائه می‌شود.

یعنی هر مشکلی در قالب سفارشی اشتراک را «خراب» نمی‌کند — پنل همیشه به صفحه داخلی بازمی‌گردد. همه صفحه‌های اشتراک (هم سفارشی و هم استاندارد) با هدرهای جلوگیری از کش (`Cache-Control: no-cache, no-store, must-revalidate`) ارائه می‌شوند تا کلاینت‌ها همیشه داده‌های تازه ترافیک و مدت اعتبار دریافت کنند.

متغیرهای موجود در قالب

مجموعه‌ای از داده‌های کاربر اشتراک در متن قالب منتقل می‌شود. دسترسی از طریق `{{ .نام }}`:

متغیر	نوع	توضیح
`{{ .sId }}`	string	ID اشتراک (UUID).
`{{ .enabled }}`	bool	آیا کاربر/اشتراک فعال است.
`{{ .download }}`	string	حجم دانلود قالب‌بندی‌شده (مثلاً «2.5 GB»).
`{{ .upload }}`	string	حجم آپلود قالب‌بندی‌شده.
`{{ .total }}`	string	محدودیت کل ترافیک قالب‌بندی‌شده.
`{{ .used }}`	string	ترافیک مصرف‌شده قالب‌بندی‌شده (دانلود + آپلود).

متغیر	نوع	توضیح
`{{ .remained }}`	string	ترافیک باقی‌مانده قالب‌بندی‌شده.
`{{ .expire }}`	int64	تاریخ انقضا — زمان Unix به **ثانیه** (`0` = بدون محدودیت). برای `Date` در JS ضربدر 1000 کنید.
`{{ .lastOnline }}`	int64	زمان آخرین آنلاین — زمان Unix به **میلی‌ثانیه** (`0` = هیچ‌وقت).
`{{ .downloadByte }}`	int64	دانلود به بایت دقیق.
`{{ .uploadByte }}`	int64	آپلود به بایت دقیق.
`{{ .totalByte }}`	int64	محدودیت کل به بایت دقیق.
`{{ .subUrl }}`	string	URL صفحه اشتراک.
`{{ .subJsonUrl }}`	string	URL تنظیمات JSON اشتراک.
`{{ .subClashUrl }}`	string	URL تنظیمات Clash/Mihomo.
`{{ .subTitle }}`	string	عنوان اشتراک از تنظیمات (ممکن است خالی باشد).
`{{ .subSupportUrl }}`	string	URL پشتیبانی از تنظیمات (ممکن است خالی باشد).
`{{ .links }}`	string[]	لیست رشته‌های تنظیمات (VLESS، VMess و غیره). پیمایش: `{{ range .links }}` … `{{ end }}`.
`{{ .emails }}`	string[]	لیست ایمیل‌های مرتبط با اشتراک.
`{{ .datepicker }}`	string	قالب تقویم فعلی پنل: `gregorian` یا `jalali` (از تنظیم «نوع تقویم» گرفته می‌شود؛ اگر خالی باشد — `gregorian`).

یک مثال حداقلی از بدنه قالب که از بخشی از متغیرها استفاده می‌کند:

```
<h1>{{ .subTitle }}</h1>
<p>مصرف‌شده: {{ .used }} از {{ .total }} (باقی‌مانده {{ .remained }})</p>
{{ range .links }}<div>{{ . }}</div>{{ end }}
```

مثال: تاریخ انقضا از `expire`. فیلد `{{ .expire }}` — زمان Unix به **ثانیه** است، بنابراین برای JavaScript آن را ضربدر 1000 می‌کنند؛ مقدار `0` به معنای «بدون محدودیت» است:

```
<script>
  var exp = {{ .expire }};
  document.write(exp === 0
    ? 'بدون محدودیت'
    : 'تا ' + new Date(exp * 1000).toLocaleDateString());
</script>
```

توجه: `{{ .lastOnline }}` از پیش به **میلی‌ثانیه** است — نیازی به ضرب در 1000 ندارد.

11. Xray: مسیریابی، DNS، outbounds و افزونه‌ها

بخش **«تنظیمات Xray»** ویرایشگر قالب پیکربندی Xray-core است که پنل بر اساس آن فایل نهایی `config.json` را برای اجرای هسته تولید می‌کند. راهنمای بخش قالب: «بر اساس این قالب، فایل پیکربندی *Xray* ساخته می‌شود.» برخلاف inboundها (که جداگانه در پایگاه داده ذخیره می‌شوند و هنگام ساخت پیکربندی در قالب جاگذاری می‌شوند)، همه چیز دیگر — لاگ‌ها، مسیریابی، outboundها، DNS، سیاست، آمار — دقیقاً اینجا تعریف می‌شود.

> مهم: مقدار قالب در پایگاه داده با کلید `xrayTemplateConfig` ذخیره می‌شود. هنگام ذخیره، پنل آن را از چند تبدیل خودکار عبور می‌دهد (ببینید 11.11). هر JSON نادرست از نظر ساختاری با خطای *«xray template config invalid»* رد می‌شود.

موقعیت در منو: «Outbounds» و «Routing»

«Outbounds» و **«Routing»** آیتم‌های جداگانه منوی کناری هستند (بلافاصله زیر «Hosts»، بالای «تنظیمات پنل»)، هر کدام آدرس خود را دارند: `/outbound` و `/routing`. لینک‌های مستقیم به این صفحات و بارگذاری مجدد صفحه به درستی کار می‌کنند. در زیرمنوی **«پیکربندی‌های Xray»** فقط این آیتم‌ها باقی می‌مانند: اصلی، DNS، Balancer و قالب پیشرفته. در توضیحات زیر، بخش‌های 11.3 و 11.4 به ترتیب با صفحات «Routing» و «Outbounds» مطابقت دارند.

11.1. ساختار ویرایشگر: تب‌ها/حالت‌ها

ویرایشگر چند حالت نمایش قالب را ارائه می‌دهد (فیلترها بر اساس بخش‌های JSON):

حالت	چه چیزی نشان می‌دهد
اصلی	بخش‌های پایه (لاگ، مسیریابی پایه، تنظیمات اصلی)
قالب پیشرفته	قالب کامل JSON Xray
همه	همه بخش‌ها به طور همزمان

گروه‌های منطقی تنظیمات درون ویرایشگر:

- **تنظیمات اصلی** (راهنما: «این پارامترها تنظیمات عمومی را توصیف می‌کنند»).
- **لاگ** (ببینید 11.10).
- **اتصالات پایه**: مسدودسازی‌ها و مسیرهای مستقیم.
- **Inboundها** (راهنما: «تغییر قالب پیکربندی برای اتصال کلاینت‌های خاص»).
- **Outboundها** (ببینید 11.4).
- **Balancer** (ببینید 11.5).
- **Routing** (راهنما: «اولویت هر قانون مهم است!»، ببینید 11.3).
- **DNS / Fake DNS** (ببینید 11.6).

11.2. تنظیمات اصلی (General)

Freedom Protocol Strategy

فیلد	برچسب	توضیح	پیش‌فرض
`FreedomStrategy`	**تنظیم استراتژی پروتکل Freedom**	استراتژی خروجی شبکه برای outbound مستقیم (freedom). راهنما: «تنظیم استراتژی خروجی شبکه در پروتکل *Freedom*». فیلد `domainStrategy` داخل `settings` outbound با پروتکل `freedom` را کنترل می‌کند.	در قالب مرجع، `domainStrategy` برای freedom-outbound `direct` برابر **`AsIs`** است (آدرس بدون resolve به صورت اصلی ارسال می‌شود).

`domainStrategy` برای freedom (مقادیر Xray-core): `AsIs` (domain) را در سمت سرور resolve نکن)، و همچنین خانواده `UseIP` / `UseIPv4` / `UseIPv6` و نسخه‌های «اجباری» آن‌ها `ForceIP*` که سرور خروجی را مجبور می‌کنند دامنه را resolve کرده و از طریق IP به دست آمده متصل شود. اگر سرور خروجی IPv6 ندارد یا نیاز است فقط از IPv4 استفاده شود، به `UseIPv4` تغییر دهید.

Freedom Happy Eyeballs (IPv4/IPv6)

فیلد	برچسب	توضیح
`FreedomHappyEyeballs`	**Freedom Happy Eyeballs (IPv4/ IPv6)**	راهنما: «*Dual-stack dialing* برای *outbound* مستقیم *(freedom)* — مفید در سرورهای خروجی با *IPv4* و *IPv6*.» الگوریتم Happy Eyeballs را برای freedom-outbound فعال می‌کند (تلاش همزمان از هر دو خانواده آدرس).
try delay	(راهنما)	«میلی‌ثانیه‌ها قبل از تلاش با خانواده آدرس دیگر. *150–250* میلی‌ثانیه نقطه شروع خوبی است.» تأخیر قبل از سوئیچ به خانواده آدرس جایگزین. بازه توصیه‌شده 150–250 میلی‌ثانیه است.

Overall Routing Strategy

فیلد	برچسب	توضیح	پیش‌فرض
`RoutingStrategy`	**تنظیم مسیریابی دامنه**	استراتژی کلی resolve DNS برای مسیریابی. راهنما: «تنظیم استراتژی کلی مسیریابی *resolve DNS*». فیلد `routing.domainStrategy` را کنترل می‌کند.	در قالب مرجع `routing.domainStrategy` = **`AsIs`**.

`routing.domainStrategy` نحوه تطبیق قوانین IP مسیریابی با درخواست‌های دامنه را مشخص می‌کند: `AsIs` (فقط قوانین دامنه، بدون resolve)، `IPIfNonMatch` (اگر دامنه با قوانین تطابق نداشت — resolve کن و قوانین IP را بررسی کن)، `IPOnDemand` (فوراً هنگام مواجهه با قانون IP resolve کن). برای کارکرد قوانین IP (مثلاً `geoip:*`) با درخواست دامنه معمولاً `IPIfNonMatch` لازم است.

Outbound Test URL

فیلد	برچسب	توضیح	پیش‌فرض
`outboundTestUrl`	**URL آزمون outbound**	URL برای بررسی اتصال هنگام آزمایش outbound. راهنما: «*URL* برای بررسی اتصال *outbound*». جداگانه از قالب، با کلید `xrayOutboundTestUrl` ذخیره می‌شود.	**`https://www.google.com/generate_204`**

مقدار sanitize می‌شود. در هنگام آزمایش outbound، به عنوان یک URL عمومی اضافی بررسی می‌شود — این محافظت در برابر SSRF است: کاربر نمی‌تواند URL دلخواه (از جمله داخلی) را از طریق کلاینت تزریق کند، URL آزمون همیشه از تنظیمات سرور گرفته می‌شود. مقدار خالی هنگام ذخیره/آزمون با `generate_204` پیش‌فرض جایگزین می‌شود.

Block BitTorrent

فیلد	برچسب	توضیح
`Torrent`	**مسدودسازی BitTorrent**	قانونی به `routing.rules` اضافه می‌کند که ترافیک با `protocol: ["bittorrent"]` را به outbound `blocked` می‌فرستد. در قالب مرجع این قانون به صورت پیش‌فرض وجود دارد.

محدودیت‌های اتصال (Connection Limits)

راهنما: «سیاست‌های سطح اتصال برای کاربران سطح *0*. برای استفاده از مقدار پیش‌فرض *Xray*، فیلد را خالی بگذارید.» این پارامترها در `policy.levels.0` نوشته می‌شوند.

فیلد	برچسب	توضیح	پیش‌فرض
`connIdle`	**تایم‌اوت بیکاری** (ثانیه)	«اتصال را پس از بیکاری به مدت تعداد ثانیه مشخص شده می‌بندد. کاهش مقدار، حافظه و *file descriptor*ها را در سرورهای پرترافیک سریع‌تر آزاد می‌کند (پیش‌فرض در *Xray*: *300*).»	خالی → پیش‌فرض Xray **300**
`bufferSize`	**اندازه بافر** (KB)	«اندازه بافر داخلی به ازای هر اتصال بر حسب *KB*. برای به حداقل رساندن استفاده از حافظه در سرورهای کم‌رم، *0* قرار دهید (مقدار پیش‌فرض *Xray* به پلتفرم بستگی دارد).» Placeholder: **«auto»**.	خالی → به پلتفرم بستگی دارد؛ `0` — به حداقل برسان

نمونه (`policy.levels.0`). فیلدهای این گروه در سیاست سطح 0 نوشته می‌شوند. برای آزادسازی سریع‌تر منابع در سرور پرترافیک با RAM کم:

```
"policy": {
  "levels": {
    "0": {
      "connIdle": 120,
      "bufferSize": 0
    }
  }
}
```

اینجا اتصال پس از 120 ثانیه بیکاری بسته می‌شود (به جای پیش‌فرض 300)، و `bufferSize: 0` مصرف حافظه بافر را به حداقل می‌رساند. فیلدی که در فرم خالی گذاشته شود، در JSON نخواهد بود — و Xray مقدار پیش‌فرض خود را اعمال می‌کند.

11.3. قوانین مسیریابی (routing)

لیست قوانین `routing.rules` . **ترتیب بحرانی است** («اولویت هر قانون مهم است!»): قوانین از بالا به پایین ارزیابی می‌شوند، اولین تطابق اجرا می‌شود. راهنما: «برای تغییر ترتیب، بکشید». دکمه‌های کنترل ترتیب: **اول**، **آخر**، **بالاتر**، **پایین‌تر**.

هر قانون دارای `type: "field"` است. دکمه‌ها: **ایجاد قانون**، **ویرایش قانون**. راهنمای فیلدهای لیست: «آیتم‌ها با ویرگول جدا می‌شوند».

در صفحه «Routing» دکمه‌های **«وارد کردن قوانین»** و **«خروجی قوانین»** در منوی کشویی **«بیشتر»** (more) گردآوری شده‌اند — مانند صفحه «Outbounds». دکمه **«خروجی قوانین»** فایل را بلافاصله دانلود نمی‌کند، بلکه پنجره مدالی با پیش‌نمایش JSON و دکمه‌های **«کپی»** و **«دانلود»** باز می‌کند: محتوا را می‌توان قبل از ذخیره مشاهده کرد. خروجی outboundها در صفحه «Outbounds» نیز به همین شکل عمل می‌کند.

Route Tester (آزمایشگر مسیر)

در تب Routing یک زیرتب **Route Tester** وجود دارد — از Xray در حال اجرا می‌پرسد که کدام outbound یک اتصال خاص را پردازش می‌کند، بدون ارسال ترافیک واقعی. دامنه یا IP، پورت، شبکه (TCP/UDP) و در صورت لزوم inbound و پروتکل intercept شده (`http` / `tls` / `quic` / `bittorrent`) را مشخص کنید، سپس **Test Route** را فشار دهید. تصمیم مستقیماً از موتور مسیریابی زنده گرفته می‌شود.

در پاسخ، outbound انتخاب‌شده نشان داده می‌شود، و هنگام استفاده از balancer — تگ balancer نیز نشان داده می‌شود. اگر هیچ قانونی تطابق نداشت، آزمایشگر اعلام می‌کند که ترافیک به outbound پیش‌فرض (اول در لیست `outbounds`) می‌رود. این برای بررسی ترتیب قوانین قبل از اعتماد به آن‌ها مفید است.

فعال و غیرفعال کردن یک قانون جداگانه

یک قانون مسیریابی جداگانه را می‌توان موقتاً با یک کلیدزن **غیرفعال** کرد، بدون حذف. در جدول قوانین یک ستون **«فعال»** با کلیدزن (Switch) وجود دارد، و در فرم قانون نیز فیلد **«فعال»** — همچنین کلیدزن. قانون غیرفعال در پیکربندی نهایی Xray قرار نمی‌گیرد، اما در قالب ذخیره می‌شود و می‌توان آن را در هر زمان دوباره فعال کرد.

قانون سرویس آمار (`inboundTag: ["api"] → outboundTag: "api"`) را نمی‌توان غیرفعال کرد — کلیدزن آن قفل شده تا حسابداری ترافیک پنل خراب نشود (ببینید 11.11).

فیلدهای فرم قانون

فیلد فرم	برچسب	فیلد JSON	توضیح
منبع	**منبع**	`source`	آدرس‌های IP/زیرشبکه منبع. لیست با ویرگول.
پورت منبع	**پورت منبع**	`sourcePort`	پورت(های) منبع.
مقصد	**مقصد**	`domain` + `ip` + `port`	دامنه‌ها، IP و پورت‌های هدف. دامنه‌ها از پیشوندهای `domain:` ، `full:` ، `regexp:` ، `keyword:` و همچنین `geosite:*` پشتیبانی می‌کنند؛ IPها — `geoip:*` و CIDR.
شبکه	—	`network`	`tcp` ، `udp` یا `tcp,udp` . از 3.4.2 با حروف کوچک در کانفیگ نوشته می‌شود (`tcp` / `udp`)، و در جدول مسیرها با حروف بزرگ نمایش داده می‌شود (TCP / UDP)؛ خالی — هر شبکه.
پروتکل	—	`protocol`	`http` ، `tls` ، `bittorrent` (از طریق sniffing تشخیص داده می‌شود).
کاربر	**کاربر**	`user`	فیلتر بر اساس ایمیل/شناسه کاربر.
ویژگی‌ها / مقدار	**ویژگی‌ها / مقدار**	`attrs`	ویژگی‌های هدر HTTP برای تطابق.
VLESS route	**VLESS route**	—	مسیریابی بر اساس فیلد route برای VLESS.

فیلد فرم	برچسب	فیلد JSON	توضیح
تگ‌های inbound	**تگ‌های inbound**	`inboundTag`	یک یا چند تگ inbound که قانون برای آن‌ها اعمال می‌شود (شامل `api` داخلی، و تگ DNS از تنظیمات DNS). در لیست‌های inbound به صورت «tag (remark)» نمایش داده می‌شود اگر inbound یادداشت جداگانه داشته باشد، در غیر این صورت فقط تگ؛ در قوانین ذخیره‌شده همچنان فقط تگ‌ها نگهداری می‌شوند.
تگ outbound	**تگ outbound / اتصال خروجی**	`outboundTag`	ترافیک تطابق‌یافته به کجا هدایت شود.
تگ balancer	**تگ balancer / Balancer**	`balancerTag`	راهنما: «ترافیک را از طریق یکی از *load balancer*های پیکربندی‌شده هدایت می‌کند».

> تضاد `outboundTag` و `balancerTag`: «نمی‌توان *balancerTag* و *outboundTag* را همزمان استفاده کرد. در صورت استفاده همزمان، فقط *outboundTag* کار می‌کند.» در یک قانون، یا تگ outbound یا تگ balancer تعیین کنید.

قوانین داخلی قالب مرجع

در `config.json` استاندارد، بخش `routing` شامل سه قانون است (به این ترتیب):

1. `inboundTag: ["api"] → outboundTag: "api"` — قانون سرویسی برای gRPC-API آمار پنل.
2. `ip: ["geoip:private"] → outboundTag: "blocked"` — مسدودسازی محدوده‌های خصوصی.
3. `protocol: ["bittorrent"] → outboundTag: "blocked"` — مسدودسازی BitTorrent.

> قانون `api → api` همیشه به طور خودکار هنگام ذخیره به موقعیت 0 منتقل می‌شود (ببینید 11.11)، تا یک قانون catch-all بالاتر، درخواست آمار را «بلعیده» نشود.

نمونه قانون. ارسال تمام ترافیک به سایت‌های روسی و شبکه‌های خصوصی به صورت مستقیم (بدون پروکسی)، و بقیه — به balancer. ترتیب مهم است: قانون «هدایت مستقیم» باید بالاتر از catch-all باشد. در `routing.rules`:

```
{
  "type": "field",
  "domain": ["geosite:category-ru", "domain:example.ru"],
  "ip": ["geoip:ru", "geoip:private"],
  "outboundTag": "direct"
}
```

برای اینکه قوانین IP (`geoip:ru`) برای درخواست‌های دامنه نیز کار کنند، معمولاً در سطح بالای مسیریابی به `routing.domainStrategy: "IPIfNonMatch"` نیاز است (ببینید 11.2).

گروه‌های از پیش پیکربندی‌شده مسیریابی (اتصالات پایه)

در حالت «اتصالات پایه» پنل کمک می‌کند قوانین معمول از لیست‌های آماده تهیه شوند:

گروه	فیلدها	راهنما
مسدودسازی بر اساس پروتکل/سایت	—	«پیکربندی کنید تا کلاینت‌ها به پروتکل‌های خاص دسترسی نداشته باشند»

گروه	فیلدها	راهنما
مسدودسازی بر اساس کشور	**IPهای مسدود شده، دامنه‌های مسدود شده**	«این پارامترها ترافیک را بر اساس کشور مقصد مسدود می‌کنند.»
اتصالات مستقیم	**IPهای مستقیم، دامنه‌های مستقیم**	«اتصال مستقیم به این معنی است که ترافیک خاص از طریق سرور دیگری هدایت نمی‌شود.»
قوانین IPv4	–	«این پارامترها به کلاینت‌ها اجازه می‌دهند فقط از طریق *IPv4* به دامنه‌های هدف مسیریابی کنند»
قوانین WARP	–	«این گزینه‌ها ترافیک را بر اساس مقصد خاص از طریق *WARP* هدایت می‌کنند.»
مسیریابی NordVPN	–	«این گزینه‌ها ترافیک را بر اساس مقصد خاص از طریق *NordVPN* هدایت می‌کنند.»

MTProto-inbound: مسیریابی ترافیک Telegram از طریق Xray

MTProto-inbound دارای کلیدزن **«Route through Xray»** (به طور پیش‌فرض خاموش) و یک انتخاب اختیاری **Outbound** است. وقتی فعال می‌شود، پنل یک پل SOCKS حلقه‌ای با تگ همان inbound به پیکربندی Xray اضافه می‌کند، و mtg ترافیک Telegram را از طریق آن هدایت می‌کند. پس از این، ترافیک خروجی Telegram توسط router مدیریت می‌شود: می‌توان آن را با قوانین معمول در تب Routing بر اساس تگ inbound تطابق داد یا به اجبار به outbound یا balancer انتخاب‌شده از طریق فیلد **Outbound** هدایت کرد. **Outbound** را خالی بگذارید تا قوانین مسیریابی تصمیم بگیرند.

11.4. Outbounds (اتصالات خروجی)

لیست `outbounds`. دکمه‌ها: **ایجاد اتصال خروجی**، **ویرایش اتصال خروجی**. راهنما: «تغییر قالب پیکربندی برای تعریف اتصالات خروجی این سرور».

در قالب مرجع دو outbound اجباری وجود دارد:

- `protocol: "freedom"`, `tag: "direct"` — خروج مستقیم به اینترنت (با `domainStrategy: "AsIs"` و `finalRules: [{action: "allow"}]`);
- `protocol: "blackhole"`, `tag: "blocked"` — «سیاهچاله» برای ترافیک مسدودشده.

فیلدهای عمومی فرم outbound

فیلد	برچسب	توضیح
تگ	**تگ** (راهنما: «تگ یکتا»)	شناسه یکتای outbound. Placeholder: «تگ-یکتا». اعتبارسنجی: «تگ الزامی است»، «تگ قبلاً توسط *outbound* دیگری استفاده شده».
پروتکل	–	نوع outbound (ببینید زیر).
آدرس / پورت	**آدرس** / پورت	هدف اتصال. آدرس و پورت الزامی هستند.
ارسال از طریق	**ارسال از طریق**	آدرس IP محلی رابط خروجی (`sendThrough`). Placeholder: «*IP* محلی».
Dialer proxy (زنجیر)	–	راهنما: «این *outbound* را از طریق *outbound* دیگری (با تگ) متصل کنید تا زنجیر پروکسی بسازید. برای اتصال مستقیم خالی بگذارید.» Placeholder: «برای زنجیر، *outbound* را انتخاب کنید». از طریق `streamSettings.sockopt.dialerProxy` پیاده‌سازی می‌شود.

لیست کشویی **Dialer Proxy** نه تنها outboundهای محلی، بلکه تگ‌های outboundهای از subscriptionها را نیز نشان می‌دهد — پس زنجیر را می‌توان از طریق خروجی دریافت‌شده از subscription نیز ساخت. blackhole-outbound و outbound در حال ویرایش از لیست حذف شده‌اند. برای اتصال مستقیم، فیلد را خالی بگذارید.

پروتکل‌های پشتیبانی‌شده outbound

پروتکل‌های پشتیبانی‌شده توسط فرم:

- **`freedom`** — خروج مستقیم. فیلدهای `settings.domainStrategy` ، `finalRules` (ببینید زیر)، Happy Eyeballs. قابل آزمایش نیست («*Outbound has no testable endpoint*»).
- **`blackhole`** — ترافیک را دور می‌اندازد. فیلد **نوع پاسخ** قابل آزمایش نیست.
- **`socks`** ، **`http`** — لیست `settings.servers[]` با `port` / `address` ؛ فیلد **رمز عبور احراز هویت**. برای پروتکل **`http`** زیر فیلدهای **Password/Username** یک ویرایشگر **Headers** (هدرها) وجود دارد — جفت‌های کلید/مقدار برای هدرهای CONNECT ارسال‌شده به HTTP-proxy بالادستی. این هدرها هنگام باز کردن مجدد و ذخیره outbound حفظ می‌شوند (قبلاً از دست می‌رفتند). توجه: فقط هدرهای سطح تنظیمات (`settings.headers`) اعمال می‌شوند؛ هدرهای در سطح سرور جداگانه توسط xray-core نادیده گرفته می‌شوند.
- **`vmess`** — `settings.vnext[]` (`port` / `address`).
- **`vless`** — `settings.port` / `settings.address` .
- **`trojan`** ، **`shadowsocks`** — `settings.servers[]` .
- **`wireguard`** — `settings.peers[]` با `endpoint` ، به علاوه کلیدها (ببینید 11.8).
- **`hysteria`** — `settings.port` / `settings.address` (انتقال UDP).

برای outbound از نوع **loopback** بلوک **Sniffing** با همان پارامترهای inbound موجود است: فعال‌سازی، **destOverride**، **Metadata Only**، **Route Only** و لیست **دامنه‌های مستثنی**.

در ماسک **UDP** (FinalMask) برای **Hysteria2** حالت‌های اضافی در دسترس هستند. ماسک **Salamander** یک انتخابگر **Mode** با مقادیر **Salamander** و **Gecko** دارد: حالت Gecko پدینگ تصادفی بسته را با فیلدهای **Max/Min** اندازه اضافه می‌کند (`packetSize` ، محدوده 1–2048، پیش‌فرض 512–1200) — این از fingerprinting بر اساس طول بسته محافظت می‌کند. ماسک **Realm** (UDP hole-punching) یک بلوک اختیاری **TLS Config** با فیلدهای **Server Name** (SNI)، **ALPN** (`h3` / `h2` / `http/1.1`)، **Fingerprint** (uTLS) و کلیدزن **Allow Insecure** دارد.

نمونه: زنجیر از طریق SOCKS بالادستی. Outbound `upstream` به یک SOCKS5-proxy خارجی می‌رود، و `chained` ترافیک خود را از طریق آن ارسال می‌کند (`dialerProxy`)، یک زنجیر می‌سازد. در `outbounds` :

```
[
  {
    "tag": "upstream",
    "protocol": "socks",
    "settings": {
      "servers": [{ "address": "203.0.113.10", "port": 1080 }]
    }
  },
  {
    "tag": "chained",
    "protocol": "freedom",
    "streamSettings": {
      "sockopt": { "dialerProxy": "upstream" }
    }
  }
]
```

حالا قانون مسیریابی با `outboundTag: "chained"` ترافیک را از طریق `upstream` به اینترنت خواهد فرستاد.

وارد کردن outbound از لینک اشتراک‌گذاری

Outbound را می‌توان از لینک اشتراک‌گذاری (`vless://` ، `vmess://` و غیره) وارد کرد. هنگام وارد کردن، تنظیمات مالتی‌پلکس **xmux** (XHTTP) که در بلوک `extra=` لینک منتقل شده‌اند نیز ذخیره می‌شوند: پس از وارد کردن، مقادیر آن‌ها در زیرفرم **XMUX** outbound ایجادشده جاگذاری می‌شوند.

فیلدهای Mux (مالتی‌پلکسینگ)

حداکثر موازی‌سازی، حداکثر اتصالات، حداکثر استفاده مجدد، حداکثر درخواست‌ها، حداکثر ثانیه‌های استفاده مجدد، دوره keep alive. این پارامترها رفتار mux/XUDP outbound را پیکربندی می‌کنند.

Sockopts (تنظیمات socket)

گروه **Sockopts**: **فاصله keep alive**، **Mark (fwmark)**، **رابط**، **فقط IPv6**، **پذیرش proxy protocol**، **Proxy protocol**، **TCP user timeout (ms)**، **TCP keep-alive idle (s)**. اینجا dialer-proxy زنجیر نیز تنظیم می‌شود.

Freedom finalRules (بازنویسی مسدودسازی IP خصوصی)

برای freedom-outbound گروه **قوانین نهایی** در دسترس است:

فیلد	برچسب	توضیح
`overrideXrayPrivateIp`	**بازنویسی مسدودسازی پیش‌فرض IP خصوصی در Xray**	ممنوعیت داخلی Xray برای اتصالات خروجی به IPهای خصوصی را برمی‌دارد.
`action`	**عمل**	`allow` (مانند قالب مرجع: `finalRules: [{action: "allow"}]`)، `redirect` (**Redirect**) یا سایر.
`blockDelay`	**تأخیر مسدودسازی (ms)**	تأخیر قبل از رد کردن اتصال.
`fragment` / `redirect`	**Fragment / Redirect**	اعمال هدایت و تکه‌بندی ترافیک.

ماسک fragment: Lengths و Delays به ازای هر تکه

در ماسک **fragment** (نوع fragment در FinalMask، برای TCP) فیلدهای تکی Length و Delay با لیست‌های **Lengths** و **Delays** جایگزین شده‌اند: برای هر سگمنت می‌توان محدوده طول جداگانه (مثلاً `100-200`) و تأخیر بر حسب میلی‌ثانیه (مثلاً `10-20` یا `0`) تعریف کرد. ردیف‌های لیست را می‌توان اضافه یا حذف کرد؛ مقادیر تکی قبلاً ذخیره‌شده به صورت خودکار به آرایه یک‌عنصری منتقل می‌شوند.

سایر فیلدهای فرم

- **UDP over TCP** و **نسخه UoT** — برای پروتکل‌های شبیه shadowsocks.
- **بدون هدر gRPC**، **اندازه chunk Uplink** — پارامترهای انتقال gRPC.
- فیلدهای TLS/uTLS: **بررسی نام peer**، **Pinned SHA256**، **Short ID**، **Vision testpre**، placeholder «نام سرور».

آزمایش outboundها

دکمه‌ها: **آزمون**، **آزمون همه**. وضعیت‌ها: **در حال آزمون اتصال...**، **آزمون موفق**، **آزمون ناموفق**، **آزمون اتصال خروجی ناموفق بود**. نتیجه: **نتیجه آزمون**، تأخیر بر حسب میلی‌ثانیه.

دو حالت (راهنما: *«TCP: probe سریع HTTP dial-only: درخواست کامل از طریق xray.»*):

- **TCP** (`mode=tcp`) — dial ساده به `host:port` ، به صورت موازی روی همه endpointها اجرا می‌شود، ~تایم‌اوت 5 ثانیه. فقط دسترسی‌پذیری TCP را بررسی می‌کند، پروتکل پروکسی را اعتبارسنجی نمی‌کند. برای `freedom` / `blackhole` /تگ `blocked` *«Outbound has no testable endpoint»* برمی‌گردد.
- **HTTP** (`mode=http` یا خالی) — یک نمونه موقت Xray راه‌اندازی می‌کند، یک درخواست HTTP واقعی اجرا می‌کند (probe URL = `outboundTestUrl` سرور)، تأخیر واقعی را اندازه می‌گیرد. معتبر اما پرهزینه: با یک قفل جهانی سریال می‌شود (*«Another outbound test is already running, please wait»*). تایم‌اوت یک تلاش — 10 ثانیه، پنجره انتظار نتیجه — 15 ثانیه (افزایش یافته تا outboundهای سالم روی کانال‌های کند یا tunneled به عنوان «Failed» علامت‌گذاری نشوند). در صورت شکست، دلیل واقعی (خطای connection refused، DNS، انقضای deadline، خطای TLS و غیره) در لاگ پنل/Xray نوشته می‌شود.

> پروتکل‌های UDP (`wireguard` ، `hysteria`) و انتقال‌های UDP (`kcp` ، `quic` ، `hysteria`) **همیشه** در حالت HTTP آزمایش می‌شوند، حتی اگر TCP درخواست شده باشد — یک dial UDP خالی نمی‌تواند endpoint «زنده» را از «مرده» تشخیص دهد. برای wireguard در پیکربندی آزمایشی `noKernelTun: true` به اجبار تنظیم می‌شود.

بررسی دسته‌ای و تجزیه به مراحل

آزمون و **آزمون همه** در حالت HTTP یک نمونه موقت مشترک Xray برای یک دسته outbound راه‌اندازی می‌کنند، برای هر کدام یک SOCKS-inbound حلقه‌ای با قانون ایجاد می‌کنند و به صورت موازی یک درخواست HTTP واقعی از طریق آن ارسال می‌کنند؛ **آزمون همه** outboundها را دسته‌ای بررسی می‌کند. **آزمون همه** outboundهای دریافت‌شده از subscriptionها را نیز بررسی می‌کند (جدول «از subscriptionها»، فقط خواندنی) — ردیف‌های آن‌ها نیز با نتیجه آزمون هایلایت می‌شوند. در این بین، outboundهای `freedom` («direct») و `dns` در هیچ حالتی آزمایش نمی‌شوند (این‌ها پروکسی نیستند): دکمه آزمون برای آن‌ها در دسترس نیست، **آزمون همه** آن‌ها را رد می‌کند، و محافظت سرور آزمون HTTP آن‌ها را حتی در صورت فراخوانی مستقیم API ممنوع می‌کند. علاوه بر موفقیت/خطا، popup نتیجه وضعیت HTTP پاسخ و تجزیه زمانی به مراحل را نشان می‌دهد: **Proxy connect** (اتصال به پروکسی)، **TLS via outbound** (TLS از طریق outbound) و **First byte** (زمان تا اولین بایت) — این کمک می‌کند بفهمید در کدام مرحله تأخیر یا خرابی رخ داده است.

آمار ترافیک outboundها

پنل شمارنده‌های ترافیک را بر اساس تگ‌ها (`total` / `down` / `up`) نگه می‌دارد. دکمه reset شمارنده‌ها را برای یک تگ خاص یا برای همه (`tag = "-alltags-"`) صفر می‌کند. فیلدهای **اطلاعات حساب** و **وضعیت اتصال خروجی** خلاصه را نشان می‌دهند.

11.5. Balancerها (Balancers)

لیست `routing.balancers` . دکمه‌ها: **ایجاد balancer**، **ویرایش balancer**.

در تب Balancers ستون‌های وضعیت زنده وجود دارد: **Live Target** هدف فعلی balancer را در Xray در حال اجرا نشان می‌دهد، و **Override** اجازه می‌دهد انتخاب هدف را به صورت دستی بازنویسی کنید (مقدار **Auto (strategy)** انتخاب را به استراتژی برمی‌گرداند). وضعیت با یک دکمه جداگانه به‌روز می‌شود. اگر balancer هنوز در Xray در حال اجرا فعال نباشد، پنل پیشنهاد می‌دهد ابتدا تغییرات را ذخیره کنید یا Xray را راه‌اندازی کنید.

فیلد	برچسب	توضیح
تگ	**تگ** (راهنما: «تگ یکتا»)	شناسه یکتا. Placeholder: «تگ یکتای *balancer*». اعتبارسنجی: «تگ الزامی است»، «تگ قبلاً توسط *balancer* دیگری استفاده شده».
انتخابگرها	**انتخابگرها**	لیست تگ‌های outbound (بر اساس زیررشته) که balancer از میان آن‌ها خروجی انتخاب می‌کند. حداقل یکی باید انتخاب شود: «حداقل یک *outbound* انتخاب کنید».
Fallback	**Fallback**	تگ outbound پشتیبان، اگر هیچ انتخابگری مناسب نبود.
استراتژی	**استراتژی**	الگوریتم انتخاب (ببینید زیر).

استراتژی و پارامترهای ناظر

استراتژی (`strategy.type`) تعیین می‌کند که balancer چگونه outbound را از انتخابگرها انتخاب می‌کند. مقادیر Xray-core: `random` (تصادفی)، `roundRobin` (به نوبت)، `leastPing` (حداقل تأخیر بر اساس نتایج observatory)، `leastLoad` (حداقل بار). برای `leastPing` / `leastLoad` از پارامترهای `strategy.settings` استفاده می‌شود:

فیلد	برچسب	توضیح
`expected`	**مورد انتظار**	Placeholder: «تعداد بهینه گره‌ها» — تعداد هدف گره‌های زنده.
`maxRtt`	**حداکثر RTT**	حد بالای RTT قابل قبول هنگام انتخاب نامزدها.
`tolerance`	**تحمل**	tolerance هنگام مقایسه تأخیرها/بار.
`baselines`	**Baselines**	مقادیر آستانه تأخیر برای گروه‌بندی گره‌ها.
`costs`	**Costs**	ضرایب وزنی (cost) برای تگ‌های جداگانه.

نمونه استراتژی‌ها. بلوک `strategy` داخل balancer قرار می‌گیرد (در JSON — کنار `tag` و `selector`):

```
"strategy": { "type": "random" }      // انتخابگرها از تصادفی انتخاب
"strategy": { "type": "roundRobin" }  // متناوباً ،نوبت به
"strategy": { "type": "leastPing" }   // (ناظر به نیاز) تأخیر حداقل
```

برای `leastLoad` پارامترها در `settings` تعریف می‌شوند:

```
"strategy": {
  "type": "leastLoad",
  "settings": {
    "expected": 2,
    "maxRTT": "1s",
    "tolerance": 0.05,
    "baselines": ["500ms", "1s", "2s"],
    "costs": [
      { "regexp": false, "match": "proxy-premium",   "value": 0.1 },
      { "regexp": true,  "match": "^proxy-cheap-.+$", "value": 5 }
    ]
  }
}
```

نحوه اجرا (با مثال). فرض کنید ناظر تأخیرها را اندازه گرفته: `A = 250 ms`، `B = 280 ms`، `C = 700 ms`، `D = 1500 ms`. با تنظیمات بالا انتخاب به این صورت است:

1. **`maxRTT: "1s"`** — خروجی‌هایی با تأخیر بیش از 1 ثانیه حذف می‌شوند: `D` (1500 ms) حذف می‌شود. `A`، `B`، `C` باقی می‌مانند.
2. **`expected + baselines`** — خروجی‌ها بر اساس آستانه‌های تأخیر گروه‌بندی می‌شوند، و **کوچک‌ترین** آستانه‌ای که حداقل `expected` خروجی در آن قرار می‌گیرد انتخاب می‌شود. آستانه `500ms` از قبل شامل `A` و `B` است — این 2 عدد (= `expected`) است، پس گروه { `A`، `B` } انتخاب می‌شود. `C` (700 ms) در انتخاب قرار نمی‌گیرد مادامی که خروجی‌های سریع کافی باشند («ذخیره داغ» است).
3. **`tolerance: 0.05`** — داخل گروه انتخاب‌شده، خروجی‌هایی که تأخیرشان بیش از 5٪ با هم اختلاف ندارند برابر در نظر گرفته می‌شوند و بار به طور مساوی تقسیم می‌شود. `A` (250) و `B` (280) ~12٪ (< 5٪) اختلاف دارند، پس با فرض برابری بقیه، `A` سریع‌تر ترجیح داده می‌شود؛ اگر اختلاف در محدوده 5٪ بود — ترافیک هم از `A` و هم از `B` می‌رفت.
4. **`costs`** — قبل از مقایسه، «هزینه» خروجی‌های جداگانه را تنظیم می‌کنند: `value` کمتر خروجی را جذاب‌تر می‌کند، بزرگتر — برعکس. در مثال `proxy-premium` مقدار `0.1` می‌گیرد (ارزان‌تر می‌شود و بیشتر انتخاب می‌شود)، و همه `proxy-cheap-*` (با regex، `regexp: true`) — `5` (گران‌تر می‌شوند و آخر استفاده می‌شوند). این روش اجازه می‌دهد خروجی‌ها را به نرمی اولویت‌بندی کنید بدون حذف سخت آن‌ها.

نتیجه: ترافیک بیشتر از طریق `A` می‌رود (با تأخیرهای نزدیک — به طور مساوی با `B`)، `C` ذخیره می‌ماند، `D` حذف شده تا زمانی که RTTش زیر `maxRTT` برسد.

ناظر: `observatory` و `burstObservatory` (اندازه‌گیری برای `leastPing` / `leastLoad`)

استراتژی‌های `leastPing` و `leastLoad` خودشان هیچ چیز اندازه نمی‌گیرند — به داده‌های تأخیر و دسترسی‌پذیری هر outbound نیاز دارند. این داده‌ها را **ناظر** (observatory) جمع‌آوری می‌کند: به طور دوره‌ای هر outbound تحت نظر را «ping» می‌کند و زمان پاسخ و دسترسی‌پذیری را ذخیره می‌کند. همان داده‌ها در تب **«Observatory»** نشان داده می‌شوند (وضعیت‌های **فعال / غیرقابل دسترس**، **«آخرین فعالیت»**، **«آخرین تلاش»**).

از نسخه 3.4.2 صفحه «بالانسرها» به تب‌های **«تنظیمات بالانسر»** (جدول و افزودن) و **«Observatory»** تقسیم شده است، و ناظر با یک **فرم ساختاری** ویرایش می‌شود (ویرایشگر JSON خام قبلی حذف شده است). پنل خودش ناظرها را در صورت نیاز ایجاد و حذف می‌کند: `observatory` معمولی — برای استراتژی **Least Ping**، `burstObservatory` پیشرفته — برای **Least Load**، و همچنین برای **Random/Round-robin** با `fallbackTag` **تعیین‌شده** (xray-core این را لازم دارد؛ هنگام پاک کردن آخرین `fallbackTag` ناظر حذف می‌شود). ایجاد یا حذف یک ناظر نیازمند راه‌اندازی مجدد کامل Xray است.

دو گزینه در دسترس است:

- **`observatory`** — ساده: `subjectSelector` + `probeURL` + `probeInterval`.

- **`burstObservatory`** — پیشرفته، با تنظیم دقیق ping از طریق `pingConfig`؛ برای چندین خروجی مناسب است.

نمونه بلوک `burstObservatory`:

```
{
  "subjectSelector": ["WS-SE", "WS-FR", "WS-PL"],
  "pingConfig": {
    "destination": "https://www.google.com/generate_204",
    "interval": "1m",
    "connectivity": "http://connectivitycheck.platform.hicloud.com/generate_204",
    "timeout": "5s",
    "sampling": 2
  }
}
```

کارکرد فیلدها:

فیلد	چه چیزی تعریف می‌کند
`subjectSelector`	لیست **پیشوند تگ‌های** outbound برای نظارت. Xray تمام outboundهایی که تگشان با رشته‌های مشخص‌شده شروع می‌شود را انتخاب می‌کند. در مثال خروجی‌های `WS-SE…`، `WS-FR…`، `WS-PL…` مورد نظارت هستند. این تگ‌ها باید با آنچه در **انتخابگرها** balancer انتخاب شده مطابقت داشته باشند.
`pingConfig.destination`	URL که **از طریق هر outbound** برای اندازه‌گیری تأخیر درخواست می‌شود. یک صفحه «سبک» با پاسخ `204` بدون بدنه انتخاب می‌شود — مثلاً `https://www.google.com/generate_204`. زمان تا پاسخ همان تأخیر اندازه‌گیری‌شده است.
`pingConfig.interval`	چند بار هر outbound پینگ شود. رشته مدت: `"1m"` — هر یک دقیقه، همچنین `"30s"`، `"5m"` و غیره. بیشتر — داده تازه‌تر، اما ترافیک پس‌زمینه بیشتر.
`pingConfig.connectivity`	(اختیاری) URL بررسی **اتصال پایه** خود سرور. اگر غیرقابل دسترس باشد — مشکل در شبکه سرور است، و ناظر outbound را به عنوان غیرقابل دسترس **علامت‌گذاری نمی‌کند** (محافظت در برابر false positive در صورت خرابی محلی). معمولاً هم endpoint با پاسخ `204`.
`pingConfig.timeout`	چقدر منتظر پاسخ یک ping بمانیم قبل از اینکه تلاش ناموفق شمرده شود (مثلاً `"5s"`).
`pingConfig.sampling`	چند اندازه‌گیری آخر برای هر outbound نگه داشته و میانگین گرفته شود. `2` — دو ping آخر را در نظر بگیر (نوسانات تصادفی را هموار می‌کند).
`pingConfig.httpMethod`	**جدید در 3.4.2.** متد HTTP پروب‌ها: `HEAD` (پیش‌فرض) یا `GET`.

در تب **«Observatory»** همین پارامترها با یک فرم تنظیم می‌شوند (نه JSON). برای `observatory` معمولی این‌ها عبارت‌اند از **Watched Outbounds** (`subjectSelector`، فقط‌خواندنی)، **Probe URL** (`probeURL`، به‌طور پیش‌فرض `https://www.google.com/generate_204`)، **Probe Interval** (`probeInterval`، `1m`) و **Enable Concurrency** (`enableConcurrency`، به‌طور پیش‌فرض فعال). برای `burstObservatory` — فیلدهای فهرست‌شده بالا به‌علاوه **«متد HTTP»** جدید. اگر بالانسر هر دو ناظر را داشته باشد، یک کلید بخش‌بندی‌شده بین فرم‌های Observatory و Burst جابه‌جا می‌کند.

نحوه ارتباط دادن همه چیز:

1. در تب **«Observatory»** پارامترهای ناظر را تنظیم کنید (ناظر به‌طور خودکار تحت استراتژی انتخاب‌شده ایجاد می‌شود؛ `subjectSelector` از انتخابگرهای بالانسر پیروی می‌کند).
2. یک balancer ایجاد کنید: **استراتژی** = `leastPing`، در **انتخابگرها** همان تگ‌های outbound را مشخص کنید (`WS-SE`، `WS-FR`، `WS-PL`).
3. ترافیک را با یک قانون مسیریابی به آن هدایت کنید (فیلد **تگ balancer**، ببینید 11.3).

4. Xray را راه‌اندازی مجدد کنید. در تب **«Observatory»** وضعیت خروجی‌ها ظاهر می‌شود، و balancer شروع به انتخاب سریع‌ترین خروجی زنده می‌کند.

> در یک قانون نمی‌توان هم `balancerTag` و هم `outboundTag` تعریف کرد — فقط `outboundTag` کار می‌کند.

چه اتفاقی هنگام حذف outbound یا بالانسر می‌افتد (از 3.4.2)

هنگام حذف یک outbound یا بالانسر، پنل در همان اقدام **ارجاعات به آن در مسیریابی را پاک می‌کند** و در دیالوگ تأیید **پیش‌نمایش پیامدها** را نشان می‌دهد (عنوان «با حذف، مسیریابی شما نیز به‌روز می‌شود:»، سپس برای هر قانون — «حذف شد (مقصدی باقی نماند)» یا «نگه داشته شد (اکنون از ... استفاده می‌کند)»، و «بالانسر ... حذف شد (هدفی باقی نماند)»). به این ترتیب از یک ارجاع «معلق» (`balancerTag` / `outboundTag` / `dialerProxy`) جلوگیری می‌شود که قبلاً ممکن بود xray-core را در راه‌اندازی مجدد بعدی از اجرا بازدارد. حذف بالانسر: قوانین `outboundTag` خود را حفظ می‌کنند، در غیر این صورت حذف می‌شوند. حذف outbound: تگ آن از انتخابگرها و `fallbackTag` بالانسرها برداشته می‌شود، `dialerProxy` سایر outboundها پاک می‌شود، بالانسرهای خالی‌شده همراه با قوانین اکنون بی‌مقصد حذف می‌شوند، و ناظرها بازسازی می‌شوند.

11.6. DNS

بخش `dns`. فعال‌سازی: **فعال‌سازی DNS** (راهنما: «فعال‌سازی سرور *DNS* داخلی»).

پارامترهای عمومی DNS

فیلد	برچسب	JSON	توضیح / راهنما
`tag`	**نام تگ DNS**	`dns.tag`	«این تگ در قوانین مسیریابی به عنوان تگ *inbound* در دسترس خواهد بود.» اجازه می‌دهد خود درخواست‌های DNS از طریق `inboundTag` مسیریابی شوند.
`clientIp`	**IP کلاینت**	`dns.clientIp`	«برای اطلاع‌رسانی به سرور درباره موقعیت *IP* مشخص‌شده در طول درخواست‌های *DNS* استفاده می‌شود» (EDNS Client Subnet).
`strategy`	**استراتژی درخواست**	`dns.queryStrategy`	«استراتژی کلی رزولو نام دامنه». مقادیر: `UseIP` ، `UseIPv4` ، `UseIPv6` .
`disableCache`	**غیرفعال کردن کَش**	`dns.disableCache`	«کَش‌گذاری *DNS* را غیرفعال می‌کند».
`disableFallback`	**غیرفعال کردن DNS پشتیبان**	`dns.disableFallback`	«درخواست‌های *DNS* پشتیبان را غیرفعال می‌کند».
`disableFallbackIfMatch`	**غیرفعال کردن DNS پشتیبان در صورت تطابق**	`dns.disableFallbackIfMatch`	«در صورت تطابق لیست دامنه سرور *DNS*، درخواست‌های *DNS* پشتیبان را غیرفعال می‌کند».

فیلد	برچسب	JSON	توضیح / راهنما
`enableParallelQuery`	**فعال‌سازی درخواست‌های موازی**	–	«فعال‌سازی درخواست‌های *DNS* موازی به چندین سرور برای رزولو سریع‌تر».
`useSystemHosts`	**استفاده از Hosts سیستم**	`dns.useSystemHosts`	«استفاده از فایل *hosts* سیستم نصب‌شده».

نمونه بلوک `dns`. درخواست‌های دامنه‌های Google از طریق سرور DoH Cloudflare رزولو می‌شوند، بقیه — از طریق `1.1.1.1`؛ برای Google فقط IPهای غیرخصوصی انتظار می‌رود. در سطح بالای کانفیگ:

```
"dns": {
  "tag": "dns-inbound",
  "queryStrategy": "UseIPv4",
  "servers": [
    {
      "address": "https://cloudflare-dns.com/dns-query",
      "domains": ["geosite:google"],
      "expectIPs": ["geoip:!private"]
    },
    "1.1.1.1"
  ]
}
```

رشته سرور (`"1.1.1.1"`) بدون فیلد — سرور پیش‌فرض برای همه دامنه‌های دیگر است. تگ `dns-inbound` سپس می‌تواند به عنوان `inboundTag` در قوانین مسیریابی استفاده شود تا خود درخواست‌های DNS از طریق outbound مناسب هدایت شوند.

کَش رکوردهای منقضی‌شده

فیلد	برچسب	توضیح
`serveStale`	**استفاده از منقضی‌شده**	«بازگرداندن نتایج منقضی‌شده از کَش در حین به‌روزرسانی در پس‌زمینه».
`serveExpiredTTL`	**TTL منقضی‌شده**	«مدت اعتبار (ثانیه) رکوردهای منقضی‌شده کَش؛ *0* = بدون پایان».

سرورهای DNS (لیست `dns.servers`)

دکمه‌ها: **ایجاد DNS**، **ویرایش DNS**، **حذف همه** (تأیید: «همه سرورهای *DNS* از لیست حذف خواهند شد. این عمل قابل بازگشت نیست.»). قالب‌ها: **استفاده از قالب**، پنجره **قالب‌های DNS**، شامل پیش‌تنظیم **خانوادگی**.

با فشار دادن **ویرایش DNS** روی رکورد سرور DNS (مانند رکورد Fake DNS)، پنجره ویرایش مقادیر ذخیره‌شده سرور را جاگذاری می‌کند، نه مقادیر پیش‌فرض.

فیلدهای سرور DNS:

فیلد	برچسب	توضیح
address	–	آدرس DoH URL، IP) DNS، `localhost`، `fakedns` و غیره).
`domains`	**دامنه‌ها**	لیست دامنه‌هایی که از این سرور استفاده می‌کنند.

فیلد	برچسب	توضیح
`expectIPs`	**IPهای مورد انتظار**	فقط در صورتی پاسخ را بپذیر که IP در لیست باشد.
`unexpectIPs`	**IPهای غیرمنتظره**	پاسخ‌های با IPهای مشخص‌شده را رد کن.
`skipFallback`	**رد کردن Fallback**	از این سرور به عنوان fallback استفاده نکن.
`finalQuery`	**درخواست نهایی**	سرور را به عنوان نهایی در زنجیر علامت‌گذاری می‌کند.
`timeoutMs`	**تایم‌اوت (ms)**	تایم‌اوت درخواست به سرور.

Hosts (رکوردهای ایستا)

گروه **Hosts** (`dns.hosts`). دکمه **افزودن Host**؛ وضعیت خالی **هیچ Hostی تعریف نشده**. فیلدها: دامنه (placeholder: «دامنه (مثلاً *domain:example.com*)») و مقادیر (placeholder: «*IP* یا دامنه — وارد کنید و *Enter* بزنید»).

لاگ‌های DNS

ببینید 11.10: پرچم **لاگ‌های DNS** (`dnsLog`) در بخش لاگ‌گیری.

11.7. Fake DNS

بخش `fakedns` . دکمه‌ها: **ایجاد Fake DNS**، **ویرایش Fake DNS**.

فیلد	برچسب	توضیح
`ipPool`	**زیرشبکه استخر IP**	محدوده CIDR که از آن IPهای مصنوعی صادر می‌شوند (مثلاً `198.18.0.0/15`).
`poolSize`	**اندازه استخر**	تعداد آدرس‌هایی که در استخر حلقه‌ای نگه داشته می‌شوند.

Fake DNS در ترکیب با sniffing روی inbound استفاده می‌شود: هسته یک IP مصنوعی به کلاینت می‌دهد، مطابقت دامنه⍰IP را به خاطر می‌سپارد و دامنه را هنگام مسیریابی بازیابی می‌کند. برای کارکرد Fake DNS، سرور DNS با آدرس `fakedns` باید به لیست سرورهای DNS اضافه شود.

نمونه: ترکیب Fake DNS + سرور DNS. ابتدا استخر آدرس‌های مصنوعی را تعریف می‌کنیم، سپس سرور DNS `fakedns` را اضافه می‌کنیم تا درخواست‌های دامنه IP از این استخر بگیرند:

```
"fakedns": [
  { "ipPool": "198.18.0.0/15", "poolSize": 65535 }
],
"dns": {
  "servers": [
    { "address": "fakedns", "domains": ["geosite:geolocation-!cn"] },
    "1.1.1.1"
  ]
}
```

علاوه بر این، روی inbound باید sniffing با `destOverride: ["fakedns"]` فعال باشد، در غیر این صورت هسته محلی برای بازیابی دامنه واقعی ندارد.

11.8. WireGuard / WARP / NordVPN

فیلدهای WireGuard (`wireguard`)

فیلد	برچسب	توضیح
`secretKey`	**کلید مخفی**	کلید خصوصی رابط محلی.
`publicKey`	**کلید عمومی**	کلید عمومی peer.
`psk`	**کلید مشترک**	PreShared Key (اختیاری).
`allowedIPs`	**آدرس‌های IP مجاز**	محدوده‌هایی که در تونل مسیریابی می‌شوند.
`endpoint`	**نقطه پایانی**	`host:port` peer.
`domainStrategy`	**استراتژی دامنه**	استراتژی رزولو برای WireGuard-outbound.

Cloudflare WARP (`warp`)

یکپارچه‌سازی از API `https://api.cloudflareclient.com/v0a4005` استفاده می‌کند (client-version `a-6.30-3596`).
اعمال controller (`/xray/warp/:action`): `config` ، `reg` ، `license` ، `data` ، `del` .

مرحله به مرحله:

1. **ایجاد حساب WARP** ← `reg` : پنل کلیدهای خصوصی (`privateKey`) و عمومی (`publicKey`) را تولید/می‌پذیرد، دستگاه را در Cloudflare ثبت می‌کند و `access_token` ، `device_id` ، `license_key` ، `private_key` (و همچنین `client_id`) را در تنظیم `warp` ذخیره می‌کند.
2. **کلید لایسنس WARP / WARP+** ← `license` : تنظیم کلید 26 کاراکتری (placeholder WARP+): «کلید 26 کاراکتری *WARP+*».
   در صورت خطا: «تنظیم لایسنس *WARP* ناموفق بود.» اگر کانفیگ هنوز دریافت نشده: «ابتدا کانفیگ *WARP* را دریافت کنید.»
3. **اطلاعات حساب: نام دستگاه، مدل دستگاه، دستگاه فعال، نوع حساب، نقش، WARP+ data، سهمیه، استفاده.**
4. **افزودن outbound** — یک WireGuard-outbound با کلیدها و endpoint Cloudflare دریافت‌شده ایجاد می‌کند.
5. **حذف حساب** ← `del` : داده‌های ذخیره‌شده WARP را پاک می‌کند.

NordVPN (`nord` / `nordvpn`)

یکپارچه‌سازی از NordLynx (= WireGuard) استفاده می‌کند. اعمال controller (`/xray/nord/:action`): `countries` ، `servers` ، `reg` ، `setKey` ، `data` ، `del` .

مرحله به مرحله:

1. **توکن دسترسی** ← `reg` : پنل از `api.nordvpn.com` اطلاعات ورود NordLynx را درخواست می‌کند و `nordlynx_private_key` را استخراج می‌کند. `private_key` و `token` را در تنظیم `nord` ذخیره می‌کند. جایگزین — `setKey` : وارد کردن مستقیم **کلید خصوصی** (نمی‌تواند خالی باشد).
2. **کشور** ← `countries` لیست کشورها را بارگذاری می‌کند؛ **شهر** (یا **همه شهرها**).
3. **سرور** ← `servers` سرورهای کشور انتخاب‌شده را بارگذاری می‌کند (`countryId` به عنوان عدد اعتبارسنجی می‌شود — محافظت در برابر injection). فیلتر: فقط سرورهایی با **بار** > 7٪ نشان داده می‌شوند. اگر سروری نبود: «سرور برای کشور انتخاب‌شده یافت نشد». اگر سرور کلید عمومی NordLynx نداشت: «سرور انتخاب‌شده کلید عمومی *NordLynx* را گزارش نمی‌دهد.»
4. ایجاد/به‌روزرسانی outbound: پیام‌های toast «*outbound NordVPN* اضافه شد» / «*outbound NordVPN* به‌روزرسانی شد».

اولویت IPv4 و userspace TUN

WireGuard-outboundهای تولیدشده توسط جادوگران WARP و NordVPN از `domainStrategy: "ForceIPv4v6"` استفاده می‌کنند (اولویت IPv4 با fallback به IPv6 روی هاست‌های v6-only) به جای `ForceIP` — این «قفل‌کردن» handshake روی هاست‌هایی با IPv6 نیمه‌پیکربندی‌شده را از بین می‌برد که رکورد AAAA endpoint Cloudflare انتخاب می‌شد. همچنین برای آن‌ها userspace TUN (`noKernelTun: true`) به جای kernel TUN فعال شده: kernel TUN به دسترسی‌ها و مسیریابی fwmark نیاز دارد و روی بسیاری از VPSها بی‌صدا شکست می‌خورد، درحالی‌که بررسی اتصال داخلی پنل همیشه از طریق userspace TUN آزمایش می‌کند — حالا ترافیک واقعی و بررسی یک مسیر می‌روند. تغییر فقط برای outboundهای تازه اضافه‌شده یا resetشده اعمال می‌شود؛ قالب‌های از قبل ذخیره‌شده تنظیمات خود را حفظ می‌کنند.

11.9. پروکسی معکوس و TUN

Reverse (پروکسی معکوس)

بخش `reverse` پیکربندی Xray. در فرم outbound یک کلیدزن به نوع **پروکسی معکوس** وجود دارد. دکمه‌ها: **ایجاد پروکسی معکوس**، **ویرایش پروکسی معکوس**.

فیلد	برچسب	توضیح
نوع	**نوع**	**Bridge** یا **Portal** — دو نقش پروکسی معکوس Xray.
دامنه	**دامنه**	دامنه-برچسب سرویسی برای جفت bridge?portal.
تگ / اتصال	**تگ / اتصال**	تگ‌ها برای اتصال bridge و portal.
Reverse Tag	**تگ پروکسی معکوس**	راهنما: «تگ *outbound* برای پروکسی معکوس ساده *VLESS*. برای غیرفعال‌سازی خالی بگذارید.» Placeholder: «تگ *outbound* (خالی = غیرفعال)». یک VLESS reverse ساده را پیاده‌سازی می‌کند.

در فرم outbound فیلدهای جریان معکوس نیز وجود دارد: **Sniffing معکوس**، **Worker**ها، **رزروشده**، **حداقل فاصله بارگذاری (ms)**، **حداکثر اندازه بارگذاری (بایت)**.

TUN (`tun`)

فیلد	برچسب	توضیح	پیش‌فرض
name	—	«نام رابط *TUN*.»	`xray0`
mtu	—	«حداکثر واحد انتقال. حداکثر اندازه بسته‌های داده.»	1500
`userLevel`	**سطح کاربر**	«همه اتصالات ایجادشده از طریق این جریان ورودی از این سطح کاربر استفاده خواهند کرد.»	0

11.10. لاگ‌ها و آمار (Stats, metrics)

لاگ (`log`)

راهنما: «لاگ‌ها می‌توانند سرور را کند کنند. فقط انواع لاگ‌های مورد نیاز را در صورت لزوم فعال کنید!» بخش `log` قالب مرجع: `access:` "none"، `error: ""`، `loglevel: "warning"`، `dnsLog: false`، `maskAddress: ""`.

فیلد	برچسب	JSON	توضیح	پیش‌فرض
`logLevel`	**سطح لاگ**	`loglevel`	«سطح لاگ برای لاگ‌های خطا...» مقادیر: `debug`، `info`، `warning`، `error`، `none`.	**`warning`**
`accessLog`	**لاگ‌های دسترسی**	`access`	«مسیر فایل لاگ دسترسی. مقدار ویژه *«none»* لاگ‌های دسترسی را غیرفعال می‌کند.»	**`none`**
`errorLog`	**لاگ‌های خطا**	`error`	«مسیر فایل لاگ خطا. مقدار ویژه *«none»* لاگ‌های خطا را غیرفعال می‌کند.»	**`""`** (پیش‌فرض)
`dnsLog`	**لاگ‌های DNS**	`dnsLog`	«فعال‌سازی لاگ‌های درخواست *DNS*»	**false**
`maskAddress`	**پوشاندن آدرس**	`maskAddress`	«با فعال‌سازی، آدرس *IP* واقعی با آدرس پوشانده‌شده در لاگ‌ها جایگزین می‌شود.»	**`""`** (خاموش)

آمار (`policy` / `stats`)

گروه **آمار** شمارنده‌ها را در `policy.system` و `policy.levels` فعال می‌کند. در قالب مرجع: `statsInboundUplink: true`، `statsInboundDownlink: true`، `statsOutboundUplink: false`، `statsOutboundDownlink: false`؛ برای سطح `0` — `statsUserUplink: true`، `statsUserDownlink: true`.

فیلد	برچسب	توضیح	پیش‌فرض
`statsInboundUplink`	**آمار uplink ورودی**	«جمع‌آوری آمار برای ترافیک خروجی همه پروکسی‌های ورودی را فعال می‌کند.»	**true**
`statsInboundDownlink`	**آمار downlink ورودی**	«جمع‌آوری آمار برای ترافیک ورودی همه پروکسی‌های ورودی را فعال می‌کند.»	**true**
`statsOutboundUplink`	**آمار uplink خروجی**	«جمع‌آوری آمار برای ترافیک خروجی همه پروکسی‌های خروجی را فعال می‌کند.»	**false**
`statsOutboundDownlink`	**آمار downlink خروجی**	«جمع‌آوری آمار برای ترافیک ورودی همه پروکسی‌های خروجی را فعال می‌کند.»	**false**

> آمار کلاینت‌ها و inboundها (uplink/downlink) — پایه نمایش ترافیک در dashboard و نزد کلاینت‌ها؛ غیرفعال‌سازی آن توصیه نمی‌شود. آمار outbound به طور پیش‌فرض خاموش است و فقط در صورتی لازم است که ترافیک را بر اساس تگ‌های خروجی ردیابی کنید.

Metrics

در قالب مرجع بخش `metrics` (`listen: "127.0.0.1:11111"`، `tag: "metrics_out"`) و API مربوطه `metrics_out` وجود دارد. پنل از این listener برای جمع‌آوری متریک‌ها و snapshots observatory استفاده می‌کند: `metrics.listen` را از قالب parse می‌کند، `/debug/vars` را poll می‌کند و تاریخچه تأخیر را بر اساس تگ‌ها جمع‌آوری می‌کند. اگر آدرس/پورت `metrics.listen` را تغییر دهید، پنل به آدرس جدید مراجعه می‌کند؛ حذف بخش `metrics` جمع‌آوری نمودارهای observatory را غیرفعال می‌کند.

> آزمایش outbound در حالت HTTP یک نمونه موقت **جداگانه** Xray با listener `metrics` خودش روی یک پورت تصادفی راه‌اندازی می‌کند — این همان listener موجود در کانفیگ اصلی نیست.

11.11. ذخیره، راه‌اندازی مجدد و تبدیل‌های خودکار

دکمه‌ها

دکمه	عمل
ذخیره	`POST /xray/update` : قالب + `outboundTestUrl` را اعتبارسنجی و ذخیره می‌کند.
راه‌اندازی مجدد Xray	سرویس را با پیکربندی ذخیره‌شده مجدداً بارگذاری می‌کند. تأیید: «راه‌اندازی مجدد *xray?*» / «سرویس *xray* را با پیکربندی ذخیره‌شده مجدداً بارگذاری می‌کند.»

پیام‌های toast: موفقیت — «*Xray* با موفقیت راه‌اندازی مجدد شد»، «*Xray* با موفقیت متوقف شد»؛ خطاها — «خطایی در راه‌اندازی مجدد *Xray* رخ داد.»، «خطایی در توقف *Xray* رخ داد.» پنجره **خروجی راه‌اندازی مجدد Xray** خروجی تشخیصی هسته را نشان می‌دهد.

اعمال گرم تغییرات (بدون راه‌اندازی مجدد کامل)

تغییرات inboundها، outboundها و قوانین مسیریابی «زنده» اعمال می‌شوند: هنگام فشار دادن **ذخیره** پنل تفاوت بین کانفیگ قدیم و جدید را محاسبه می‌کند و فقط بخش‌های تغییریافته را از طریق gRPC-API Xray (HandlerService/RoutingService) اعمال می‌کند، بدون راه‌اندازی مجدد فرایند. راه‌اندازی مجدد کامل فقط زمانی به صورت خودکار انجام می‌شود که بخش‌هایی بدون API بارگذاری گرم تغییر کنند (`log` ، `dns` ، `policy` ، `observatory` و غیره). بنابراین در صفحه Xray دکمه جداگانه «راه‌اندازی مجدد» لازم نیست — **ذخیره** خودش تغییرات را اعمال می‌کند. راه‌اندازی مجدد هسته در صورت لزوم همچنان به صورت خودکار انجام می‌شود (همچنین ببینید بارگذاری مجدد خودکار هنگام به‌روزرسانی subscriptionها و چرخش WARP).

بازگرداندن قالب به حالت پیش‌فرض

Endpoint `GET /xray/getDefaultJsonConfig` قالب مرجع (`config.json` داخلی باینری) را برمی‌گرداند. می‌توان از آن برای بازگرداندن پیکربندی به حالت کارخانه استفاده کرد.

تبدیل‌های خودکار هنگام ذخیره

هنگام ذخیره تنظیمات Xray، پنل (به این ترتیب) انجام می‌دهد:

1. **برداشتن لایه‌ها** — لایه‌هایی از نوع `{ "xraySetting": <کانفیگ>, "inboundTags": …, "outboundTestUrl": … }` را برمی‌دارد، اگر تصادفاً وارد مقدار شده باشند (وگرنه لایه‌ها با هر ذخیره انباشته می‌شوند). تا 8 لایه برداشته می‌شود.
2. **بررسی پیکربندی** — JSON به ساختار پیکربندی Xray parse می‌شود؛ در صورت خطا — رد با «*xray template config invalid*».
3. **تضمین قانون آمار** — قانون `inboundTag: ["api"] → outboundTag: "api"` به اجبار به موقعیت 0 در `routing.rules` منتقل می‌شود (یا اضافه می‌شود اگر وجود نداشته باشد). این تضمین می‌کند که درخواست gRPC آمار پنل توسط یک قانون catch-all بالاتر intercept نشود (در غیر این صورت کلاینت‌ها ممکن است با ترافیک صفر در حالی که پروکسی کار می‌کند offline نشان داده شوند).

> به دلیل بند 3، سعی نکنید قانون `api → api` را حذف یا جابجا کنید — پنل آن را در ذخیره بعدی برمی‌گرداند. این زیرساخت سرویس آمار است، نه یک مسیر کاربری.

11.12. Outbound از subscription (با به‌روزرسانی خودکار)

از نسخه 3.3.0 پنل می‌تواند `outbound` ها را مستقیماً از URL subscription وارد کند — همان فرمتی که ارائه‌دهندگان VPN برای برنامه‌های کلاینت ارائه می‌دهند. Subscriptionها به طور منظم در پس‌زمینه خوانده می‌شوند، بنابراین مجموعه `outbound` های سرور بدون ویرایش دستی قالب پیکربندی به‌روز نگه داشته می‌شود.

در رابط کاربری بخش **«Outbound Subscriptions»** نام دارد، توضیح: «وارد کردن outboundها از URL subscriptionهای راه دور (.../vmess/vless/trojan/ss). تگ‌ها برای استفاده در balancerها و قوانین مسیریابی ثابت می‌مانند. به‌روزرسانی به صورت خودکار انجام می‌شود.» بخش در صفحه Xray، بالای پانل تنظیمات `outbound` قرار دارد.

نحوه کارکرد

Subscriptionها جداگانه از قالب پیکربندی Xray ذخیره می‌شوند. قالب **هرگز بازنویسی نمی‌شود**: `outbound` های دریافت‌شده از subscriptionها هنگام هر بار تولید کانفیگ Xray به صورت زنده به پیکربندی نهایی اضافه می‌شوند.

افزودن subscription

در فرم «افزودن subscription» فیلدهای زیر در دسترس هستند:

فیلد	کلید	پیش‌فرض	هدف
URL subscription	`url`	— (الزامی)	آدرس «https Placeholder: subscription.://... (لیست لینک‌ها در base64)». فقط HTTP(S) پذیرفته می‌شود؛ آدرس برای امنیت بررسی می‌شود.
یادداشت	`remark`	خالی	برچسب دلخواه (placeholder «مثلاً گره‌های HK»).
پیشوند تگ	`tagPrefix`	`subN-`	پیشوندی که تگ‌های `outbound` های واردشده با آن شروع می‌شوند. اگر خالی گذاشته شود، پنل خودش کوچکترین شماره آزاد به صورت `sub1-` ، `sub2-` و غیره انتخاب می‌کند.
فاصله به‌روزرسانی	`updateInterval`	600 ثانیه (10 دقیقه)	هر چقدر subscription دوباره خوانده شود. در UI بر حسب ساعت/دقیقه تنظیم می‌شود.
فعال	`enabled`	بله (`true`)	فقط subscriptionهای فعال در کانفیگ قرار می‌گیرند و به صورت خودکار به‌روز می‌شوند.
مجاز بودن آدرس‌های خصوصی	`allowPrivate`	خیر (`false`)	URL روی localhost، LAN و IPهای خصوصی را مجاز می‌کند. به طور پیش‌فرض برای محافظت در برابر SSRF خاموش است — فقط برای منابع محلی قابل اعتماد فعال کنید.
قبل از outboundهای دستی	`prepend`	خیر (`false`)	اگر فعال باشد، `outbound` های این subscription **قبل از** `outbound` های دستی قالب قرار می‌گیرند، و یکی از آن‌ها می‌تواند `outbound` پیش‌فرض شود. در غیر این صورت **بعد** اضافه می‌شوند.

دکمه **«پیش‌نمایش»** (`POST /outbound-subs/parse`) اجازه می‌دهد قبل از ذخیره، URL را دانلود و parse کنید و ببینید چه `outbound` ها و تگ‌هایی به دست می‌آید؛ هیچ چیزی در پایگاه داده نوشته نمی‌شود. اگر هیچ چیزی از URL شناسایی نشد، «هیچ outboundی از این URL یافت نشد.» نمایش داده می‌شود.

ترتیب چندین subscription در لیست کلی `outbound` ها با priority تعیین می‌شود و با فلش‌های بالا/پایین تغییر می‌کند (`POST /outbound-subs/:id/move`).

کدام فرمت‌های subscription پذیرفته می‌شوند

بدنه پاسخ URL به این صورت پردازش می‌شود:

- محتوا ابتدا به عنوان **base64** (نسخه‌های استاندارد و URL-safe، با تکمیل خودکار padding و حذف فاصله/خط جدید) امتحان می‌شود. اگر base64 باشد — decode می‌شود؛ در غیر این صورت همان‌طور استفاده می‌شود.
- سپس بدنه به خطوط تقسیم می‌شود. هر خط غیرخالی که با `#` شروع نشود، به عنوان یک لینک parse می‌شود. خطوط ناشناخته (کامنت‌ها، پروتکل‌های پشتیبانی‌نشده) بی‌سروصدا رد می‌شوند.
- طرح‌های لینک پشتیبانی‌شده: `vmess://`، `vless://`، `trojan://`، `ss://` (Shadowsocks)، `hysteria2://` / `hy2://`، `wireguard://` / `wg://`.

به عبارت دیگر، یک subscription معمولی به صورت «لیست لینک‌های base64-encoded» که اکثر ارائه‌دهندگان ارائه می‌دهند مناسب است.

تگ‌های پایدار

برای هر لینک یک «هویت» پایدار محاسبه می‌شود (URI اصلی بدون fragment-یادداشت؛ برای vmess — JSON داخلی بدون فیلد `ps`). مطابقت «هویت ← تگ» ذخیره می‌شود، و در به‌روزرسانی بعدی همان سرور همان تگ را می‌گیرد، حتی اگر یادداشت یا پارامترهای فرعی تغییر کرده باشند. این عمداً طراحی شده تا balancerها و قوانین مسیریابی پس از به‌روزرسانی‌ها کار کنند:

- تگ دقیق در balancer/قانون همچنان به همان سرور اشاره خواهد کرد.
- انتخابگر پیشوند/wildcard (مثلاً `hk-*`) سرورهای جدیدی که subscription بعداً برمی‌گرداند را به طور خودکار می‌پذیرد — این روش توصیه‌شده برای «عضویت در یک pool» است.
- اگر سرور از subscription حذف شود، تگ آن به سادگی از آرایه نهایی `outbound` ها ناپدید می‌شود؛ در صورت وجود `fallbackTag` در Xray، balancer از آن استفاده می‌کند.
- اگر ارائه‌دهنده host/UUID/اطلاعات ورود سرور را تغییر داد، هویت تغییر می‌کند — این یک `outbound` جدید با تگ جدید به حساب می‌آید.

داخل یک دانلود، تگ‌ها با پسوند `-N` dedup می‌شوند. تگ‌های subscriptionها کاراکترهای non-ASCII (مثلاً کریلیک) را حفظ می‌کنند و خوانا می‌مانند: حروف و ارقام Unicode در slug حفظ می‌شوند، و علائم نگارشی با خط تیره جایگزین می‌شوند — تگ‌های از نام‌های کریلیک دیگر به تنها ارقام تقلیل نمی‌یابند.

نحوه کارکرد به‌روزرسانی خودکار

- وظیفه پس‌زمینه به‌روزرسانی subscription **هر 5 دقیقه** اجرا می‌شود.
- در هر اجرا، همه subscriptionهای فعال را پیمایش می‌کند و فقط آن‌هایی که فاصله‌شان منقضی شده را به‌روز می‌کند: یک subscription به‌روز می‌شود اگر هنوز هیچ‌وقت به‌روز نشده باشد یا از آخرین به‌روزرسانی حداقل `updateInterval` آن گذشته باشد. بنابراین وظیفه اغلب subscriptionها را بررسی می‌کند، اما هر subscription خاص بیشتر از `updateInterval` خودش (به طور پیش‌فرض 10 دقیقه) بازخوانی نمی‌شود. در UI این با راهنمای مربوطه منعکس شده.
- به‌روزرسانی URL دوباره به عنوان عمومی برای امنیت بررسی می‌شود (آدرس‌های خصوصی مسدود می‌شوند، مگر اینکه `allowPrivate` برای subscription تنظیم شده باشد)، درخواست از طریق پروکسی کلاینت پنل با هدر `User-Agent: 3x-ui-outbound-sub/1.0` می‌رود. زنجیره redirect به 10 hop محدود است، و هر hop نیز برای خصوصی بودن بررسی می‌شود (محافظت در برابر SSRF). HTTP 200 انتظار می‌رود؛ در غیر این صورت خطا ثبت می‌شود.

- پس از parse موفق، نتیجه ذخیره می‌شود، زمان آخرین به‌روزرسانی ثبت می‌شود، خطا پاک می‌شود. در صورت خطا متن آن در UI به عنوان «آخرین خطا» قابل مشاهده است، و `outbound` های دریافت‌شده قبلی همچنان معتبر می‌مانند.
- اگر حداقل یک subscription واقعاً به‌روز شد، وظیفه Xray را برای راه‌اندازی مجدد علامت می‌زند و یک UI-invalidation ارسال می‌کند تا رابط `outbound` های جدید را بکشد. راه‌اندازی مجدد واقعی Xray در نزدیک‌ترین چرخه 30 ثانیه‌ای مدیر رخ می‌دهد.

به‌روزرسانی دستی یک subscription — دکمه **«به‌روزرسانی اکنون»** (`POST /outbound-subs/:id/refresh`)؛ آن نیز Xray را برای راه‌اندازی مجدد علامت می‌زند. افزودن، تغییر، حذف subscription نیز پرچم راه‌اندازی مجدد Xray را فعال می‌کند (در صورت حذف، `outbound` هایش از کانفیگ در بارگذاری مجدد بعدی خارج می‌شوند). UI راهنمایی می‌کند: «پس از افزودن یا به‌روزرسانی، Xray را راه‌اندازی مجدد کنید (یا منتظر بارگذاری مجدد خودکار بعدی باشید) تا outboundها فعال شوند.»

چگونه وارد کانفیگ Xray می‌شود

در هر بار تولید پیکربندی Xray، `outbound` های فعال subscription به دو گروه تقسیم می‌شوند — `prepend` (پرچم «قبل از outboundهای دستی») و بقیه — و با قالب دوخته می‌شوند: `[prepend های subscriptionها] + [outbound های قالب] + [بقیه subscriptionها]`. داخل هر گروه، subscriptionها بر اساس priority مرتب می‌شوند. `outbound` های دستی قالب دست‌نخورده می‌مانند؛ اگر آرایه `outbound` های قالب به هر دلیلی parse نشد، `outbound` های subscription با آن ترکیب نمی‌شوند (تا outboundهای دستی از دست نروند).

`outbound` های واردشده علاوه بر این در خود پانل `outbound` ها در یک بلوک جداگانه **«از Outbound Subscriptionها (فقط خواندنی)»** نمایش داده می‌شوند — ویرایش آن‌ها در آنجا ممکن نیست، مدیریت فقط از طریق بخش «Outbound Subscriptions».

11.13. چرخش IP در WARP

در 3X-UI می‌توان یک WARP-outbound راه‌اندازی کرد — یک اتصال خروجی WireGuard به Cloudflare WARP (تگ `warp` در کانفیگ Xray). پنل خودش یک حساب-دستگاه را در سرورهای Cloudflare ثبت می‌کند، کلیدها و آدرس‌های WireGuard را می‌گیرد و آن‌ها را در outbound با تگ `warp` جاگذاری می‌کند. از طریق چنین outboundی ترافیک با آدرس IP Cloudflare WARP به اینترنت خارج می‌شود. نوآوری نسخه 3.3.0 — قابلیت تغییر این IP خروجی به صورت دستی یا بر اساس برنامه زمانی، بدون بازسازی دستی حساب WARP.

مدیریت در بخش **Xray** در کارت WARP قرار دارد (پس از فشار دادن «ایجاد حساب WARP» و دریافت کانفیگ؛ قبل از این اعمال در دسترس نیستند — پنل راهنمایی می‌کند «ابتدا کانفیگ WARP را دریافت کنید»).

چه اتفاقی هنگام تغییر IP می‌افتد

دکمه **«تغییر IP»** تغییر IP را آغاز می‌کند. منطق:

1. یک جفت کلید WireGuard جدید تولید می‌شود.
2. با کلید جدید، دستگاه WARP در سرورهای Cloudflare مجدداً ثبت می‌شود (نرم `device_id` ، `access_token` ، آدرس‌ها و داده‌های peer).
3. داده‌های جدید در WARP-outbound کانفیگ Xray نوشته می‌شوند: `secretKey` ، `address` (v4 `/32` و v6 `/128`)، `reserved` (از `client_id`)، و همچنین `publicKey` و `endpoint` در peer به‌روز می‌شوند.
4. اگر قبلاً کلید لایسنس WARP+ (با طول حداقل 26 کاراکتر) تنظیم شده بود، به طور خودکار روی حساب جدید مجدداً نصب می‌شود. در صورت شکست، این فقط یک هشدار در لاگ‌هاست — تغییر IP لغو نمی‌شود.
5. پس از تغییر موفق، Xray به عنوان نیازمند راه‌اندازی مجدد علامت می‌خورد تا outbound جدید فعال شود.

در صورت موفقیت، رابط کاربری «آدرس IP WARP با موفقیت تغییر کرد!» را نشان می‌دهد.

چرخش خودکار بر اساس برنامه زمانی

در کارت WARP کلیدزن **«به‌روزرسانی خودکار آدرس IP»** و فیلد **«فاصله (روز)»** وجود دارد. راهنما: «0 — غیرفعال. آدرس IP را به صورت خودکار تغییر می‌دهد.»

پارامتر	مقدار
تنظیم در پایگاه داده	`warpUpdateInterval` (عدد صحیح، ⩾ 0)
مقدار پیش‌فرض	`0` (چرخش خودکار خاموش)
واحد اندازه‌گیری	روز
`0`	تغییر خودکار را غیرفعال می‌کند
`> 0`	IP را هر N روز تغییر بده

ذخیره فاصله، `warpUpdateInterval` را ذخیره می‌کند، و اگر مقدار بزرگ‌تر از 0 باشد «زمان آخرین به‌روزرسانی» را به لحظه فعلی reset می‌کند — وگرنه برنامه‌ریز IP را در نزدیک‌ترین تیک تغییر می‌داد.

برنامه زمانی توسط یک وظیفه پس‌زمینه اجرا می‌شود که هر ساعت یک بار اجرا می‌شود — یعنی پنل هر ساعت بررسی می‌کند آیا وقت چرخش است. الگوریتم بررسی:

- اگر فاصله ⩾ 0 — هیچ کاری نمی‌کند؛
- اگر «زمان آخرین به‌روزرسانی» برابر 0 باشد (مثلاً فاصله با ویرایش مستقیم پایگاه داده تنظیم شده) — این اولین اجرا است: وظیفه فقط نشانگر زمان پایه را ثبت می‌کند و IP را فوراً **تغییر نمی‌دهد**؛
- اگر از آخرین به‌روزرسانی حداقل `فاصله × 24 × 3600` ثانیه گذشته باشد — همان تغییر IP انجام می‌شود، نشانگر زمان به‌روز می‌شود و راه‌اندازی مجدد Xray برنامه‌ریزی می‌شود.

جزئیت مهم: تغییر دستی با دکمه «تغییر IP» نیز نشانگر زمان آخرین به‌روزرسانی را reset می‌کند. بنابراین پس از چرخش دستی، شمارش فاصله خودکار از نو شروع می‌شود و تغییر برنامه‌ریزی‌شده فوراً پس از آن اجرا نمی‌شود.

«از طریق پروکسی پنل»

> **تغییر در 3.3.1.** تنظیم جداگانه «پروکسی شبکه پنل» (`panelProxy`) حذف شده. ترافیک خروجی خود پنل (از جمله درخواست‌های به WARP API) حالا از طریق **outbound انتخاب‌شده برای ترافیک پنل** هدایت می‌شود — Xray-outbound یا balancer (ببینید بخش 13). توضیح زیر به نسخه‌های قبل از 3.3.1 مربوط می‌شود.

همه درخواست‌های به API Cloudflare WARP (ثبت‌نام، دریافت کانفیگ، نصب لایسنس، تغییر IP) مستقیم نمی‌روند، بلکه از طریق HTTP-client پنل با timeout 15 ثانیه می‌روند. این client تنظیم **«پروکسی شبکه پنل»** (`panelProxy`) از تنظیمات پنل را رعایت می‌کند.

از توضیح تنظیم: پروکسی درخواست‌های خروجی خود پنل (به‌روزرسانی‌های پایگاه‌داده geo، بررسی‌های نسخه Xray/پنل، Telegram، و حالا مراجعات به WARP) را مسیریابی می‌کند — تا فیلتر سرور را دور زند. آدرس‌هایی به صورت `socks5://` یا `http(s)://` پذیرفته می‌شوند، مثلاً یک SOCKS-inbound محلی خود Xray. اگر فیلد خالی باشد یا پروکسی نادرست باشد — از اتصال مستقیم استفاده می‌شود (رفتار خراب نمی‌شود).

مزیت برای WARP: اگر سرور نمی‌توانست مستقیماً به `api.cloudflareclient.com` برسد، ثبت‌نام و چرخش قبلاً شکست می‌خورد. حالا با مشخص کردن `panelProxy` یک پروکسی فعال (از جمله inbound Xray خودی)، می‌توان دسترسی‌پذیری WARP API و کارکرد هم دکمه دستی و هم چرخش برنامه‌ریزی‌شده را تضمین کرد.

چه وقت مفید است

- تغییر منظم IP خروجی برای outboundی که از طریق WARP می‌رود — ریسک بلاک‌شدن و ردیابی با یک آدرس را کاهش می‌دهد.
- «تازه کردن» IP به صورت دستی، اگر آدرس فعلی Cloudflare در لیست سیاه افتاده یا کند است.
- سرورهایی که دسترسی مستقیم به Cloudflare WARP API ندارند: مسیریابی درخواست‌ها از طریق `panelProxy` ثبت‌نام و چرخش را کارساز می‌کند.

---

12. گره‌ها (چندپانل، master/slave)

بخش **گره‌ها** یک نصب معمولی 3X-UI را به یک **پانل مرکزی (master)** تبدیل می‌کند که از راه دور پانل‌های دیگر 3X-UI (فرزند) را نظارت و مدیریت می‌کند. هر گره یک نصب جداگانه 3X-UI روی سرور خودش است؛ master از طریق API HTTP اختصاصی آن به گره متصل می‌شود، وضعیتش را بررسی می‌کند و inbounds و کلاینت‌هایی که به آن اختصاص یافته‌اند را با آن همگام‌سازی می‌کند. این همان قابلیت **چندپانل** است: به جای اینکه به هر پانل جداگانه وارد شوید، تمام سرورها را در یک فهرست می‌بینید و به صورت متمرکز مدیریت می‌کنید.

اصل مهم: **گره یک agent نیست، بلکه یک پانل کامل 3X-UI است.** master هیچ چیزی روی آن «نصب» نمی‌کند — فقط از طریق توکن به API آن متصل می‌شود. حذف گره از فهرست تنها نظارت را متوقف می‌کند؛ پانل راه دور به خودی خود تحت تأثیر قرار نمی‌گیرد (راهنما: «این کار نظارت بر گره را متوقف می‌کند. پانل راه دور تحت تأثیر قرار نخواهد گرفت»).

12.1. خلاصه بالای فهرست

بالای جدول گره‌ها، شمارنده‌های تجمیعی نمایش داده می‌شوند:

فیلد	توضیح
کل گره‌ها	تعداد کل گره‌ها در فهرست.
آنلاین	تعداد گره‌هایی که وضعیت `online` دارند.
آفلاین	تعداد گره‌هایی که وضعیت `offline` دارند.
میانگین تأخیر	میانگین تأخیر (ping) تا گره‌ها، بر حسب میلی‌ثانیه.

12.2. افزودن و ویرایش گره

دکمه‌های **افزودن گره** و **ویرایش گره** فرمی با فیلدهای گره را باز می‌کنند.

فیلدهای **نام**، **آدرس**، **پورت** و **توکن API** اجباری هستند (راهنما: «نام، آدرس، پورت و توکن API الزامی هستند»).

هنگام کلیک روی «ذخیره» (چه در افزودن و چه در ویرایش)، پانل **ابتدا قابل دسترس بودن گره را با وقفه زمانی 6 ثانیه بررسی می‌کند**. اگر گره پاسخ ندهد، رکورد ذخیره نمی‌شود و خطا نمایش داده می‌شود. یعنی نمی‌توان گره‌ای را که مشخصاً در دسترس نیست اضافه کرد.

فیلدهای فرم

فیلد	پیش‌فرض	مقادیر مجاز	توضیح
نام	— (اجباری)	رشته غیرخالی، **یکتا**	نام داخلی گره. ستون نام دارای محدودیت یکتایی است — نمی‌توان دو گره با نام یکسان ایجاد کرد. متن راهنمای placeholder: `napr. de-frankfurt-1`. در هنگام ذخیره، فاصله‌های ابتدا و انتها حذف می‌شوند.
یادداشت	خالی	هر رشته‌ای	یادداشت/توضیح اختیاری برای گره. تأثیری بر عملکرد ندارد.
طرح	`https`	`http` / `https`	پروتکل اتصال به پانل راه دور. اگر خالی بماند یا مقدار نامعتبری داشته باشد، نرمال‌سازی `https` را تنظیم می‌کند. اگر گره از طریق HTTP معمولی پاسخ دهد ولی طرح روی `https` باشد، پانل راهنمای واضحی برمی‌گرداند: «the server speaks HTTP, not HTTPS; set the node scheme to http».
آدرس	— (اجباری)	هاست یا IP	آدرس پانل راه دور. متن راهنما: `panel.example.com` یا `1.2.3.4`. آدرس نرمال‌سازی می‌شود؛ به طور پیش‌فرض، آدرس‌های خصوصی/محلی برای محافظت در برابر SSRF ممنوع هستند — «مجوز آدرس خصوصی» را ببینید.

فیلد	پیش‌فرض	مقادیر مجاز	توضیح
پورت	– (اجباری)	عدد صحیح **1–65535**	پورت وب‌پانل گره راه دور. مقادیر خارج از بازه رد می‌شوند («node port must be 1-65535»).
مسیر پایه	`/`	رشته مسیر	مسیر پایه (web base path) پانل راه دور، در صورت تنظیم نرمال‌سازی می‌شود: تضمیناً با `/` شروع و ختم می‌شود (مقدار خالی → `/`). پانل `panel/api/server/status` را هنگام بررسی به آن اضافه می‌کند.
توکن API	– (اجباری)	توکن پانل راه دور	Bearer توکن برای دسترسی به API گره. در هدر `Authorization: Bearer <token>` ارسال می‌شود. متن راهنما: «توکن از صفحه تنظیمات پانل راه دور». راهنما: «پانل راه دور توکن API خود را در بخش تنظیمات → توکن API نمایش می‌دهد». یعنی توکن باید **روی خود گره** (تنظیمات → توکن API) ایجاد و سپس اینجا وارد شود.
فعال	`true`	بله/خیر	نظارت و همگام‌سازی گره را فعال می‌کند. گره‌های غیرفعال **توسط وظایف پس‌زمینه بررسی نمی‌شوند** (heartbeat و traffic-sync از آن‌ها رد می‌شوند) و در به‌روزرسانی انبوه پانل شرکت نمی‌کنند.
مجوز آدرس خصوصی	`false`	بله/خیر	محافظت SSRF را برمی‌دارد و اتصال به گره از طریق آدرس خصوصی/محلی را مجاز می‌کند. راهنما: «فقط برای گره‌های در شبکه خصوصی یا VPN فعال کنید». تنها زمانی فعال کنید که گره واقعاً در شبکه خصوصی است یا از طریق VPN در دسترس است.

دریافت و تولید مجدد توکن در سمت گره

توکن از پانل راه دور در بخش **تنظیمات** → **توکن API** گرفته می‌شود. در همانجا می‌توان آن را مجدداً صادر کرد: دکمه **تولید مجدد توکن** با هشدار: «تولید مجدد، توکن فعلی را باطل می‌کند. هر پانل مرکزی که از آن استفاده می‌کند تا زمان به‌روزرسانی، دسترسی خود را از دست می‌دهد. ادامه می‌دهید؟». پس از تولید مجدد، توکن قدیمی در پانل master دیگر کار نخواهد کرد — باید آن را در فرم گره به‌روز کرد.

اتصال خروجی (Connection outbound)

فیلد **Connection outbound** (اتصال خروجی، `outboundTag`) تعیین می‌کند که ترافیک درخواست‌های master به API این گره چگونه از سرور خارج شود. اگر تگ یک Xray-outbound در آن انتخاب شود، درخواست‌های پانل به گره مستقیم نخواهند بود، بلکه از طریق outbound مشخص‌شده هدایت می‌شوند؛ پانل خودش یک bridge-inbound روی loopback به کانفیگ در حال اجرا اضافه می‌کند و این تغییر را بدون راه‌اندازی مجدد اعمال می‌کند. راهنما: «Route this node's panel API traffic through the selected Xray outbound. A loopback bridge inbound is added to the running config automatically and applied live. Leave empty for a direct connection».

انتخابگر مانند انتخاب outbound پانل کار می‌کند: تگ‌ها به دو گروه **Outbounds** (خروجی‌های معمولی) و **Balancers** (متوازن‌کننده‌ها) دسته‌بندی شده‌اند و خروجی‌های blackhole از فهرست پنهان هستند. مقدار خالی («placeholder «Direct connection») = اتصال مستقیم به گره.

وارد کردن inbound (انتخاب inbound های همگام‌سازی‌شده)

در فرم گره تنظیمی به نام **وارد کردن inbound** (`inboundSyncMode`) با دو حالت وجود دارد: **همه inbound ها** (`all`، پیش‌فرض) و **انتخاب‌شده** (`selected`). به طور پیش‌فرض، master همه inbound هایی را که این گره برایشان انتخاب شده به گره همگام می‌کند؛ گره‌های موجود در حالت «همه inbound ها» به کار خود ادامه می‌دهند.

در حالت **انتخاب‌شده**، یک چندانتخابی از تگ‌های inbound زیر فیلد ظاهر می‌شود. روی **بارگذاری inbound ها** کلیک کنید — master با استفاده از پارامترهای اتصال واردشده (که هنوز ذخیره نشده‌اند) از گره فهرست inbound های آن را درخواست می‌کند (endpoint: `POST /panel/api/nodes/inbounds`) و تگ‌های آن‌ها را نمایش می‌دهد؛ مورد نیاز را علامت بزنید. پانل فقط تگ‌های علامت‌زده‌شده را با گره

همگام و روی آن مستقر می‌کند، و بقیه inbound هایی که مستقیماً روی گره وجود دارند دست‌نخورده باقی می‌مانند — master آن‌ها را حذف نمی‌کند و مدیریت نمی‌کند.

مثال: درخواست فهرست inbound های گره برای وارد کردن انتخابی. پارامترهای اتصال هنوز ذخیره‌نشده در body ارسال می‌شوند؛ در پاسخ، تگ‌های inbound های موجود در گره برگردانده می‌شوند:

```
POST /panel/api/nodes/inbounds
Content-Type: application/json

{ "name": "de-fra-1", "scheme": "https", "address": "node1.example.com",
  "port": 2053, "basePath": "/", "apiToken": "abcdef..." }
```

12.3. بررسی TLS (برای گره‌های https)

گروهی از فیلدها تعیین می‌کند که master چگونه گواهی HTTPS گره را بررسی کند. این تنظیمات **فقط برای طرح `https`** معتبر هستند؛ برای گره‌های `http` نادیده گرفته می‌شوند.

بررسی TLS — لیست کشویی، راهنما: «نحوه بررسی گواهی HTTPS گره توسط پانل. پیوست یا رد کردن — برای گواهی‌های خودامضا (فقط گره‌های https)».

حالت	مقدار	پیش‌فرض	توضیح
بررسی (CA استاندارد)	`verify`	بله (پیش‌فرض)	بررسی معمولی زنجیره گواهی توسط CA معتمد. برای گره‌هایی با گواهی عمومی/Let's Encrypt مناسب است. همچنین برای تمام گره‌های `http` استفاده می‌شود.
پیوست گواهی (SHA-256)	`pin`	—	زنجیره CA بررسی نمی‌شود، اما SHA-256 گواهی برگ گره با اثرانگشت ذخیره‌شده مقایسه می‌شود (مقایسه constant-time). برای گواهی‌های **خودامضا** محافظت در برابر MITM را حفظ می‌کند. نیاز به پر کردن فیلد اثرانگشت دارد.
رد کردن بررسی	`skip`	—	بررسی گواهی کاملاً غیرفعال می‌شود. هشدار: «رد کردن بررسی، محافظت در برابر حملات مرد میانی را حذف می‌کند — توکن API ممکن است رهگیری شود. بهتر است گواهی را پیوست کنید».

به سه حالت بالا در نسخه 3.4.0 یک حالت چهارم اضافه شد — **Mutual TLS (client certificate)** (`mtls`)، که مانند بقیه فقط برای طرح `https` در دسترس است.

حالت	مقدار	پیش‌فرض	توضیح
Mutual TLS (گواهی کلاینت)	`mtls`	—	علاوه بر بررسی گواهی گره، master خود را با یک **گواهی کلاینت** صادرشده توسط CA خودش به گره اثبات می‌کند. برای گره در این حالت، **توکن API اختیاری می‌شود** — گره master را از طریق گواهی شناسایی می‌کند. با انتخاب این حالت راهنمایی نمایش داده می‌شود: «This node authenticates the panel with a client certificate. Copy this panel's CA from the Node mTLS section onto the node, set its Trusted parent CA, then restart it».

برای فعال کردن TLS متقابل برای گره: در سمت گره حالت **Mutual TLS** را تنظیم کنید، CA پانل مدیریت‌کننده را از بخش **Node mTLS** (زیر را ببینید) کپی کنید، آن را به عنوان **CA والد معتمد** در گره تنظیم کنید و گره را راه‌اندازی مجدد کنید.

اگر هر مقداری غیر از `skip`، `pin` یا `mtls` انتخاب شود، نرمال‌سازی اجباراً `verify` را تنظیم می‌کند.

پیوست گواهی

با انتخاب **پیوست گواهی** موارد زیر ظاهر می‌شوند:

- **SHA-256 گواهی پیوست‌شده** — فیلد ورودی اثرانگشت را در قالب **base64** (فرمت `pinnedPeerCertSha256` از Xray) یا **hex** با دونقطه یا بدون (سبک `openssl -fingerprint`) می‌پذیرد. راهنما: «SHA-256 گواهی گره به صورت base64 یا hex. روی «دریافت» کلیک کنید تا همین الان از گره خوانده شود». متن placeholder: «SHA-256 به صورت base64 یا hex». هنگام انتخاب `pin`، اثرانگشت خالی یا نادرست باعث خطای اعتبارسنجی هنگام ذخیره می‌شود.

مثال: همان اثرانگشت در دو فرمت. فیلد هر یک از گزینه‌ها را می‌پذیرد — هر دو همان گواهی را مشخص می‌کنند:

```
# base64 (فرمت pinnedPeerCertSha256 از Xray)
607TNg3l2k0pq8R1sT2uV3wX4yZ5a6B7c8D9e0F1g2=

# hex با دونقطه (سبک openssl x509 -fingerprint -sha256)
E8:E2:D3:60:DE:5D:9A:4D:29:AB:CF:11:B2:7C:34:...
```

اگر اثرانگشت هنوز مشخص نیست، روی **دریافت** کلیک کنید — master خودش آن را از گره از طریق HTTPS می‌خواند و در فیلد قرار می‌دهد.

- دکمه **دریافت** — بدون بررسی گواهی از طریق HTTPS به گره متصل می‌شود و SHA-256 گواهی برگ فعلی را می‌خواند (endpoint: `POST /certFingerprint`) و آن را در فیلد قرار می‌دهد. در صورت موفقیت — «گواهی فعلی گره دریافت شد»؛ در صورت شکست — «دریافت گواهی ناموفق بود». فقط برای گره‌های https در دسترس است.

Node mTLS (احراز هویت TLS متقابل بین پانل‌ها)

در صفحه **گره‌ها** یک بخش جداگانه **Node mTLS** وجود دارد — تنظیم احراز هویت TLS متقابل که یک عامل دوم (گواهی کلاینت) را برای فراخوانی‌های «پانل → گره» به توکن API اضافه می‌کند. TLS متقابل اختیاری است؛ اگر فیلدهای بخش خالی باشند، گره‌ها با طرح قبلی کار می‌کنند — **فقط با توکن API** (راهنما: «Mutual TLS adds a client-certificate factor on top of the API token for node-to-node calls. It is opt-in: leave it empty to keep token-only auth»). در این بخش دو عملیات وجود دارد:

- **کپی CA این پانل** (`POST /panel/api/nodes/mtls/ca`) — گواهی ریشه (CA) این پانل را در کلیپ‌بورد کپی می‌کند. این CA باید به گره‌های مدیریت‌شده داده شود تا به گواهی کلاینت پانل اعتماد کنند؛ در خود گره‌ها پس از آن حالت بررسی TLS **Mutual TLS** تنظیم می‌شود (راهنما: «Hand this CA to the nodes this panel manages, then set their TLS verification to Mutual TLS»). پس از کپی — «گواهی CA در کلیپ‌بورد کپی شد».
- **CA والد معتمد** (`Trusted parent CA`، `POST /panel/api/nodes/mtls/trustCA`) — فیلدی که وقتی خود این پانل به عنوان گره برای یک پانل مافوق (مدیریت‌کننده) عمل می‌کند، استفاده می‌شود. CA پانل مدیریت‌کننده را اینجا بچسبانید تا از آن گواهی کلاینت بخواهید، سپس روی **Save trust CA** کلیک کنید. این تغییر نیاز به **راه‌اندازی مجدد پانل** دارد (راهنما: «When this panel is itself a node, paste the managing panel's CA here to require its client certificate. Restart the panel to apply»).

12.4. اطلاعات نمایش داده‌شده برای هر گره

ستون‌های جدول و فیلدهای کارت گره (وضعیت مشاهده‌شده، که در هر بررسی heartbeat پر می‌شود):

فیلد	توضیح
وضعیت	`online` / `offline` / `unknown` — زیر را ببینید.
CPU	درصد بار پردازنده سرور راه دور.

فیلد	توضیح
حافظه	درصد استفاده از RAM (به صورت `current/total*100` محاسبه می‌شود).
آپتایم	مدت زمان کارکرد پیوسته سرور (بر حسب ثانیه).
تأخیر	زمان پاسخ گره به آخرین بررسی (میلی‌ثانیه).
آخرین ping	زمان آخرین heartbeat موفق (unix-seconds؛ `0` = «هرگز»؛ مقدار اخیر به صورت «همین الان» نمایش داده می‌شود).
نسخه Xray	نسخه Xray-core در حال اجرا روی گره.
نسخه پانل	نسخه 3X-UI روی گره — برای نشانگر به‌روزرسانی با نسخه فعلی مقایسه می‌شود.
(inbounds)	تعداد inbound هایی که روی این گره مستقر هستند.
(کلاینت‌ها)	تعداد کلاینت‌ها در inbound های گره.
(آنلاین)	تعداد کلاینت‌های گره که در حال حاضر آنلاین هستند.
(منقضی‌شده)	تعداد کلاینت‌های گره که **منقضی شده‌اند یا محدودیت ترافیک آن‌ها تمام شده است**. کلاینت‌هایی که به صورت دستی غیرفعال شده‌اند در این شمارنده محاسبه نمی‌شوند.
(سرعت)	سرعت انتقال فعلی (زنده) در inbound هایی که روی گره مستقر هستند.

شمارنده‌های inbound/کلاینت/آنلاین بر اساس GUID پایدار گره (`panelGuid`) به گره نسبت داده می‌شوند، نه شناسه محلی — تا کلاینت روی گره فرزند دقیقاً زیر همان گره فرزند محاسبه شود، نه زیر گره میانی که از طریق آن همگام‌سازی شده است.

برای inbound هایی که روی گره مستقر هستند، صفحه کلاینت‌های آنلاین، شمارنده‌ها و **سرعت انتقال فعلی** را نمایش می‌دهد. نسبت‌دهی بر اساس GUID پایدار، گره‌های «کلون‌شده» با `panelGuid` یکسان را هم به درستی تفکیک می‌کند.

وضعیت‌های گره

وضعیت	توضیح	زمان تنظیم
`online`	آنلاین	گره به بررسی `panel/api/server/status` با `success=true` پاسخ داده است.
`offline`	آفلاین	گره پاسخ نداده، خطای HTTP برگردانده، `success=false` یا پاسخ غیرقابل تشخیص داده است.
`unknown`	نامشخص	مقدار اولیه، تا زمانی که گره هنوز یک بار هم بررسی نشده است.

در صورت شکست بررسی، متن خطا ذخیره شده و با عبارتی واضح نمایش داده می‌شود که به تشخیص علت «offline» کمک می‌کند.

12.5. اقدامات روی گره

- **بررسی اتصال** (`POST /test`) — در فرم گره، اتصال را با پارامترهای واردشده (که هنوز ذخیره نشده‌اند) با وقفه زمانی 6 ثانیه بررسی می‌کند. نتیجه: «اتصال برقرار است ({ms} میلی‌ثانیه)» یا «اتصال ناموفق بود». برای رفع اشکال آدرس/پورت/توکن/TLS قبل از ذخیره مناسب است.
- **بررسی همین الان** (دکمه «بررسی همین الان»، `POST /probe/:id`) — بررسی برنامه‌نریزی‌نشده از گره‌ای که قبلاً ذخیره شده؛ وضعیت و معیارها (CPU/حافظه/آپتایم/تأخیر/نسخه‌ها) را فوری به‌روز می‌کند و heartbeat را ثبت می‌کند. در صورت شکست — «بررسی ناموفق بود».

مثال: بررسی و بررسی گره از طریق API مدیر. «بررسی اتصال» پارامترهای هنوز ذخیره‌نشده از فرم را آزمایش می‌کند:

```
POST /panel/api/nodes/test
Content-Type: application/json

{ "scheme": "https", "address": "de-frankfurt-1.example.com", "port": 2053,
  "basePath": "/", "apiToken": "eyJhbGci...", "tlsMode": "verify" }
```

بررسی برنامه‌نریزی‌نشده از گره ذخیره‌شده با id 7:

```
POST /panel/api/nodes/probe/7
```

- **به‌روزرسانی پانل** (`POST /updatePanel` با body `{ids:[…]}`) — به‌روزرسان استاندارد خودش را روی گره اجرا می‌کند: گره آخرین نسخه 3X-UI را دانلود کرده و روی آن راه‌اندازی مجدد می‌شود. دکمه **به‌روزرسانی انتخاب‌شده‌ها ({count})** این کار را برای چند گره علامت‌زده‌شده به طور همزمان انجام می‌دهد. کنار هر گره نشانگری نمایش داده می‌شود: **به‌روزرسانی موجود است** یا **به‌روز است**، بر اساس مقایسه نسخه پانل گره با آخرین نسخه.

مثال: به‌روزرسانی چندین گره با یک درخواست. id گره‌های علامت‌زده‌شده در body ارسال می‌شوند؛ فقط گره‌های فعال و `online` به‌روزرسانی می‌شوند، بقیه به عنوان رد شده برگردانده می‌شوند.

```
POST /panel/api/nodes/updatePanel
Content-Type: application/json

{ "ids": [3, 7, 12] }
```

پاسخی مانند «به‌روزرسانی روی 2 گره شروع شد، 1 شکست خورد»: گره 12 مثلاً ممکن است آفلاین بوده و به همین دلیل رد شده باشد.

- تأیید: «{count} گره به آخرین نسخه به‌روزرسانی شود؟ هر گره انتخاب‌شده آخرین نسخه را دانلود کرده و راه‌اندازی مجدد می‌شود. فقط گره‌های فعال و آنلاین به‌روزرسانی می‌شوند».
- **فقط گره‌های فعال با وضعیت `online` به‌روزرسانی می‌شوند.** گره غیرفعال در نتایج به عنوان «node is disabled» و گره آفلاین به عنوان «node is offline» علامت‌گذاری می‌شود. نتیجه: «به‌روزرسانی روی {ok} گره شروع شد، {failed} شکست خورد». اگر هیچ گره مناسبی انتخاب نشده باشد — «حداقل یک گره فعال و آنلاین انتخاب کنید».

در دیالوگ تأیید به‌روزرسانی (هم برای گره تکی و هم برای گروهی) یک چک‌باکس **به‌روزرسانی به کانال توسعه (آخرین commit)** وجود دارد. اگر فعال شود، گره‌های انتخاب‌شده نسخه rolling build dev-latest (آخرین commit شاخه main) را به جای نسخه پایدار نصب می‌کنند؛ اگر غیرفعال باشد، گره بر اساس کانال معمولی خودش به‌روزرسانی می‌شود. با فعال بودن این چک‌باکس، هشداری در زیر آن نمایش داده می‌شود: «نسخه‌های توسعه هر commit در شاخه main را ردیابی می‌کنند و نسخه‌های پایدار نیستند — rollback خودکار وجود ندارد». پرچم dev از طریق `POST /panel/api/nodes/updatePanel` به گره ارسال می‌شود و گره دقیقاً از طریق کانال dev به‌روزرسانی را شروع می‌کند.

- **Set Cert from Panel** (کمکی، `GET /webCert/:id`) — هنگام ایجاد inbound روی گره، امکان قرار دادن مسیرهای گواهی web-TLS **خود** گره (نه پانل مرکزی) را می‌دهد تا فایل‌ها دقیقاً روی گره وجود داشته باشند. نیاز دارد که گره فعال و قابل دسترس باشد.
- **حذف گره** (`POST /del/:id`) — تأیید: «گره "{name}" حذف شود؟ این کار نظارت بر گره را متوقف می‌کند. پانل راه دور تحت تأثیر قرار نخواهد گرفت». رکورد گره و آمار ترافیک انباشته آن را حذف می‌کند؛ پانل راه دور به طور معمول به کار خود ادامه می‌دهد. **یک گره فقط پس از جدا کردن همه inbound ها از آن قابل حذف است.** اگر حداقل یک inbound هنوز به گره متصل باشد (از طریق `node_id`)، پانل حذف را با خطایی مانند «cannot delete node: N inbound(s) still attached to it; detach or delete them first» رد می‌کند — ابتدا این inbound ها را جدا یا حذف کنید، سپس گره را حذف کنید. این کار از وجود inbound های «یتیم» با ارجاع آویزان به گره حذف‌شده جلوگیری می‌کند.

12.6. تاریخچه معیارها

دکمه/نمودار تاریخچه به `GET /history/:id/:metric/:bucket` مراجعه می‌کند. معیارهای موجود: **`cpu`** و **`mem`** — که در هر heartbeat موفق انباشته می‌شوند. اندازه بازه تجمیع (`bucket` ، بر حسب ثانیه) با یک لیست سفید محدود شده است:

مثال: درخواست تاریخچه. نمودار بار CPU گره 7 با تجمیع بازه‌های 60 ثانیه‌ای (تا 60 نقطه برگردانده می‌شود):

```
GET /panel/api/nodes/history/7/cpu/60
```

برای حافظه و حالت «زمان واقعی» (2 ثانیه) — به ترتیب `…/7/mem/60` و `…/7/cpu/2`. مقادیر خارج از لیست سفید رد می‌شوند («invalid bucket» / «invalid metric»).

Bucket (ثانیه)	کاربرد
2	حالت زمان واقعی
30	بازه‌های 30 ثانیه‌ای
60	بازه‌های 1 دقیقه‌ای
120	بازه‌های 2 دقیقه‌ای
180	بازه‌های 3 دقیقه‌ای
300	بازه‌های 5 دقیقه‌ای

تا 60 نقطه برگردانده می‌شود. معیار یا bucket نامعتبر رد می‌شوند («invalid bucket» / «invalid metric»).

12.7. نحوه همگام‌سازی inbound ها و کلاینت‌ها

یک inbound از طریق فیلد `node_id` به گره «تعلق دارد» (در ویرایشگر inbound، گره انتخاب می‌شود):

مثال: توکن در فرم گره. توکن از پانل فرزند (تنظیمات → توکن API) گرفته و در فیلد **توکن API** مدیر قرار می‌گیرد. در هر بررسی، مدیر آن را در هدر ارسال می‌کند:

```
GET https://panel.example.com:2053/panel/api/server/status
Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.abc123...
```

اگر پانل فرزند دارای **مسیر پایه** (web base path) باشد، مثلاً `/secret/` ، مدیر خودش آن را قبل از `panel/api/server/status` قرار می‌دهد ← `https://panel.example.com:2053/secret/panel/api/server/status` .

1. **استقرار پیکربندی (reconcile).** با هر تغییری در inbound/کلاینت متصل به گره، گره به عنوان «کثیف» علامت‌گذاری می‌شود. یک وظیفه پس‌زمینه برای هر گره فعال **با وضعیت `online`** ، در صورت وجود تغییرات، inbound های آن را (بر اساس `node_id`) روی گره مستقر می‌کند و سپس پرچم «کثیف» را ریست می‌کند. گره‌ای که غیرفعال، آفلاین یا «کثیف» است به عنوان «در انتظار» تلقی می‌شود — استقرار روی آن تا برقراری مجدد ارتباط به تعویق می‌افتد.

2. **جمع‌آوری ترافیک.** همان وظیفه از گره یک تصویر لحظه‌ای از ترافیک می‌خواهد و آن را در آمار محلی ادغام می‌کند. بر اساس ترافیک ادغام‌شده، بررسی تجاوز از محدودیت‌ها/مواعد انجام می‌شود و در صورت لزوم کلاینت‌ها غیرفعال می‌شوند؛ شمارنده «منقضی‌شده» در گره دقیقاً همین را منعکس می‌کند. اگر گره در دسترس نباشد، کلاینت‌های آنلاین آن پاک می‌شوند.

برای کلاینتی که به چند پانل به طور همزمان متصل است، مدیر در همان وظیفه **مصرف کل ترافیک آن کلاینت در تمام پانل‌ها** را به صورت اضافی به گره‌ها ارسال می‌کند (در یک جدول جداگانه روی گره، کلید — GUID مدیر؛ در هر ارسال بازنویسی می‌شود، بنابراین ریست در سمت مدیر هم توزیع می‌شود). روی گره، مقدار بزرگ‌تر از دو مقدار — محلی یا ارسال‌شده — در ترافیک کلاینت نمایش داده می‌شود، و در صورت تجاوز از سهمیه کلی، کلاینت **به صورت محلی روی خود گره** غیرفعال می‌شود (از طریق همان مکانیزم راه‌اندازی مجدد Xray در غیرفعال‌سازی خودکار که اتصالات موجود را قطع می‌کند). این وضعیتی را که گره فقط سهم خودش از ترافیک را می‌دید، مصرف را کم محاسبه می‌کرد و به ارائه خدمات به کلاینتی ادامه می‌داد که سهمیه کلی را تمام کرده بود، برطرف می‌کند. با ریست ترافیک، تمدید خودکار یا حذف کلاینت، شمارنده‌های ارسال‌شده پاک می‌شوند.

در **اولین** همگام‌سازی inbound مستقرشده روی گره (افزودن گره جدید یا وارد کردن مجدد inbound)، مدیر شمارنده‌های ترافیک کلاینت‌ها را با مقادیر واقعی از گره مقداردهی اولیه می‌کند. قبلاً در این وضعیت، شمارنده کل inbound درست منتقل می‌شد، اما شمارنده‌های کلاینت‌های فردی صفر می‌شدند، و مدیر مصرف کلاینت‌ها را برای تمام تاریخچه انباشته‌شده قبل از اتصال گره کمتر نشان می‌داد. اکنون، اگر inbound در همان همگام‌سازی ایجاد شود، ردیف جدید `client_traffics` مقدار شمارنده از گره را به ارث می‌برد (baseline برابر با آن تنظیم می‌شود، بنابراین دلتای بعدی صفر است و ترافیک دو بار محاسبه نمی‌شود). مقداردهی اولیه شمارنده فقط برای inbound ایجادشده در همین پاس اعمال می‌شود: کلاینتی که دوباره زیر یک inbound موجود ظاهر می‌شود همچنان از صفر شروع می‌کند (محافظت در برابر ترافیک «شبح»)، و کلاینت تازه حذف‌شده با بازسازی inbound آن «زنده نمی‌شود». 3. **Heartbeat.** یک وظیفه پس‌زمینه جداگانه به صورت دوره‌ای تمام گره‌های **فعال** را (با محدودیت همزمانی) از طریق `panel/api/server/status` بررسی می‌کند، وضعیت/معیارها/نسخه‌ها را به‌روز می‌کند و در صورت وجود کلاینت‌های وب، درخت به‌روزشده گره‌ها را از طریق WebSocket ارسال می‌کند.

12.8. زنجیره گره‌ها (زیرگره‌ها / گره‌های ترانزیتی)

توپولوژی می‌تواند تخت نباشد: یک گره خودش می‌تواند master برای گره‌های خودش باشد. چنین پانل‌های پایین‌دستی به عنوان **زیرگره‌ها** نزد شما نمایش داده می‌شوند — این‌ها **پروجکشن‌های «فقط خواندنی»** هستند که از گره مستقیم دریافت شده‌اند.

- راهنما: «فقط خواندنی: گره تابع، قابل دسترس از طریق {parent}. آن را از پانل خود {parent} مدیریت کنید». یعنی زیرگره را نمی‌توان اینجا ویرایش، حذف یا به‌روزرسانی کرد — تمام عملیات روی آن از پانل والد مستقیم آن انجام می‌شود.
- هویت زیرگره با GUID آن تعیین می‌شود؛ به لطف این، کلاینت‌های آنلاین و inbound ها دقیقاً زیر گره فیزیکی که آن‌ها را میزبانی می‌کند محاسبه می‌شوند، حتی در زنجیره `Node1 → Node2 → Node3` (مدیر یک سطح عمیق‌تر از طریق هر گره مستقیم «عبور می‌کند»).
- اگر گره مستقیم در دسترس نباشد، حافظه پنهان زیرگره‌های آن پاک می‌شود و زیرگره‌ها تا برقراری مجدد ارتباط از درخت ناپدید می‌شوند.

12.9. گره‌ها: جدیدها در نسخه 3.3.0

در نسخه 3.3.0 بخش **گره‌ها** سه بهبود قابل توجه دریافت کرد: نسبت‌دهی صحیح ترافیک و کلاینت‌های آنلاین در توپولوژی‌های چندلایه (multi-hop)، همگام‌سازی client-IP بین گره‌ها و یک نشانگر وضعیت جداگانه برای حالتی که پانل گره آنلاین است اما هسته Xray روی آن خراب شده است.

1. Multi-hop: نسبت‌دهی صحیح ترافیک در زنجیره زیرگره‌ها

قبلاً شمارنده‌ها (تعداد inbound، کلاینت‌های آنلاین، منقضی‌شده) در سطح گره «مستقیم» محاسبه می‌شدند. اگر زنجیره‌ای مانند `مدیر → گره 1 → گره 2 → گره 3` داشتید، همه چیزی که فیزیکاً روی `گره 2` / `گره 3` بود، اشتباهاً به `گره 1` که از طریق آن به مدیر رسیده بود نسبت داده می‌شد. در نسخه 3.3.0 نسبت‌دهی بر اساس منبع واقعی انجام می‌شود.

نحوه کارکرد:

- **زیرگره‌ها به عنوان ردیف‌های جداگانه قابل مشاهده می‌شوند.** هر پانل فهرست گره‌های مستقیم خود را منتشر می‌کند؛ فقط گره‌هایی با `Guid` مشخص وارد می‌شوند — هویت پایدار برای نسبت دادن گره یک «پرش» بالاتر لازم است. مدیر به طور دوره‌ای (از وظیفه heartbeat) این فهرست‌ها را می‌خواند و کش می‌کند، سپس زیرگره‌های «ترانزیتی» را به گره‌های مستقیم اضافه می‌کند.

- **گره‌های ترانزیتی فقط خواندنی هستند.** در رابط کاربری با علامت **«زیرگره»** مشخص می‌شوند با راهنما: «فقط خواندنی: گره تابع، قابل دسترس از طریق *{والد}*. آن را از پانل خود *{والد}* مدیریت کنید.» این ردیف دکمه‌های مدیریتی ندارد — گره از پانل والد مستقیم آن مدیریت می‌شود.

- **سلسله‌مراتب از طریق GUID.** `ParentGuid` گره مستقیم، GUID خود مدیر است؛ گره ترانزیتی، GUID گره والدش است. به این ترتیب درخت ساخته می‌شود.

- **منبع حقیقت برای شمارنده‌ها — `origin_node_guid` روی inbound است.** این `panelGuid` گره‌ای است که فیزیکاً این inbound را نگه می‌دارد. در هنگام همگام‌سازی inbound از گره تنظیم می‌شود و **در پرش‌های بعدی همان‌طور نگه داشته می‌شود،** بنابراین یک inbound به عمق تو در تو به گره واقعی نسبت داده می‌شود، نه گره میانی. شمارنده‌های تعداد inbound ها، کلاینت‌های آنلاین و کلاینت‌های منقضی‌شده بر اساس این GUID مجدداً محاسبه می‌شوند. منطق انتخاب کلید:

وضعیت inbound	به چه چیزی نسبت داده می‌شود
`origin_node_guid` تنظیم شده	به این GUID (گره منبع واقعی)
خالی، اما `node_id` تنظیم شده	GUID مصنوعی گره (نسخه قدیمی، هنوز `panelGuid` خود را گزارش نداده)
خالی و `node_id` هم خالی	GUID خود مدیر (inbound روی Xray محلی)

کلاینت‌های آنلاین هم بر اساس GUID گروه‌بندی می‌شوند، بنابراین هر ردیف گره فقط کسانی را نشان می‌دهد که واقعاً به آن متصل هستند.

آنچه کاربر می‌بیند: در توپولوژی تخت (گره‌ها مستقیماً زیر مدیر) هیچ چیز تغییر نمی‌کند — شمارنده‌های بر اساس GUID و `id` یکسان هستند. اما به محض اینکه زنجیره‌ای از گره‌ها ایجاد شود، ردیف‌های «زیرگره» در فهرست ظاهر می‌شوند، و اعداد inbound/آنلاین/منقضی‌شده برای هر گره اکنون دقیقاً بار خودش را منعکس می‌کند، نه مجموع همه چیزی که به صورت ترانزیتی از آن عبور کرده است.

2. همگام‌سازی client-IP از access.log بین گره‌ها

محدودیت IP (`limitIp` کلاینت) بر اساس آدرس‌هایی است که Xray در access.log خود می‌نویسد. قبلاً هر گره فقط اتصالات به خودش را می‌دید، بنابراین محدودیت «حداکثر N IP برای هر کلاینت» در یک خوشه کار نمی‌کرد: کلاینت می‌توانست به گره‌های مختلف متصل شده و از محدودیت عبور کند. در نسخه 3.3.0 IP های مشاهده‌شده در کل خوشه همگام می‌شوند.

نحوه کارکرد:

- در هر گره، یک وظیفه پس‌زمینه access.log را تجزیه می‌کند، IP، email کلاینت و زمان‌بند را از هر خط استخراج می‌کند و آن‌ها را در یک جدول محلی ذخیره می‌کند (یک رکورد به ازای هر IP، email ها به صورت آرایه JSON `{ip, timestamp}` ذخیره می‌شوند). آدرس‌های محلی `127.0.0.1` و `::1` دور انداخته می‌شوند.
- همگام‌سازی **هر 10 ثانیه** یک تبادل دوطرفه به ازای هر گره فعال و آنلاین انجام می‌دهد: IP ها را از گره می‌کشد و در جدول محلی ادغام می‌کند، سپس تصویر کلی مدیر را به گره می‌دهد.
- ادغام مشاهدات قدیمی و ورودی را **بدون محاسبه دوباره** یک IP مشاهده‌شده روی چند گره، و **بدون زنده کردن مجدد** رکوردهای منسوخ ترکیب می‌کند: همان آستانه قدیمی که در وظیفه محلی هم استفاده می‌شود — **30 دقیقه** — اعمال می‌شود. برای هر IP تازه‌ترین زمان‌بند ذخیره می‌شود. رکوردهای گره‌های دیگر id محلی جدیدی دریافت می‌کنند (فضاهای id گره‌ها مستقل هستند)؛ درج همزمان همان email در برابر تکراری بودن محافظت شده است.
- در محاسبه محدودیت، یک IP «زنده» تلقی می‌شود که یا در اسکن محلی فعلی مشاهده شده یا در پایگاه داده همگام‌شده زمان‌بند خیلی تازه‌ای دارد (**در محدوده 2 دقیقه**). دقیقاً همین باعث می‌شود محدودیت در مقیاس کل خوشه کار کند، حتی اگر آدرس روی گره دیگری مشاهده شده باشد. در صورت تجاوز از محدودیت، قدیمی‌ترین IP های «زنده» به لاگ fail2ban فرستاده می‌شوند و اتصالات اجباراً قطع می‌شوند (حذف/افزودن مجدد کلاینت از طریق Xray API).

آنچه کاربر می‌بیند: محدودیت تعداد IP اکنون روی کل خوشه عمل می‌کند، نه هر گره به تنهایی؛ در پانل برای هر کلاینت IP های مشاهده‌شده روی هر گره (در محدوده پنجره 30 دقیقه) قابل مشاهده است. دکمه/تنظیم جداگانه‌ای برای این وجود ندارد — همگام‌سازی به صورت خودکار در

پس‌زمینه انجام می‌شود، مشروط بر اینکه access.log در گره فعال و در دسترس باشد (برای خود محدودیت هم به Fail2Ban روی گره نیاز است).

3. نشانگر وضعیت جداگانه: پانل گره آنلاین، اما Xray خراب شده

قبلاً وضعیت گره در واقع «آنلاین / آفلاین» بود. اگر پانل گره پاسخ می‌داد، گره آنلاین تلقی می‌شد — حتی وقتی هسته Xray روی آن کار نمی‌کرد و کلاینت‌ها عملاً نمی‌توانستند متصل شوند. در نسخه 3.3.0 سلامت پانل و سلامت هسته Xray از هم جدا شدند.

نحوه کارکرد:

- هنگام بررسی گره، مدیر از پاسخ `/panel/api/server/status` راه دور، فیلدهای `xray.state` و `xray.errorMsg` را می‌گیرد و در گره ذخیره می‌کند. این فیلدها حتی هنگام ping موفق پانل، وقتی هسته ناسالم است، پر می‌شوند — دقیقاً برای تشخیص در دسترس بودن پانل از وضعیت Xray.
- مقادیر `xray.state`: `"running"` (در حال اجرا)، `"stop"` (متوقف)، `"error"` (خطا).
- این مقادیر به وضعیت‌های گره ترجمه می‌شوند. به موارد آشنا موارد جدیدی اضافه شدند:

کلید وضعیت	توضیح	زمان نمایش
`online`	«آنلاین»	پانل پاسخ می‌دهد، Xray در حال اجرا است (`running`)
`offline`	«آفلاین»	پانل در دسترس نیست / ping موفق نبود
`unknown`	«نامشخص»	وضعیت هنوز تعیین نشده
`xrayError`	«خطای Xray»	پانل آنلاین است، اما هسته Xray در وضعیت `error` است (دارای `errorMsg`)
`xrayStopped`	«متوقف‌شده»	پانل آنلاین است، اما Xray متوقف شده است (`stop`)

- برای چنین وضعیتی در رابط کاربری از یک **نشانگر بنفش جداگانه** استفاده می‌شود (رنگی متفاوت از سبز «آنلاین» و قرمز «آفلاین»). بنفش مستقیماً نشان می‌دهد: به گره می‌توان متصل شد، اما مشکل در خود هسته Xray است، نه در شبکه یا خود پانل.

آنچه کاربر می‌بیند: به جای «سبز» گمراه‌کننده هنگام از کار افتادن هسته، گره با رنگ **بنفش** و وضعیت **«خطای Xray»** یا **«متوقف‌شده»** مشخص می‌شود. این فوراً نشان می‌دهد که باید Xray روی گره را تعمیر کرد (راه‌اندازی مجدد هسته، بررسی `errorMsg`)، نه اینکه مشغول عیب‌یابی دسترسی به خود گره باشید. همان `xrayError` / `xrayState` به زیرگره‌های ترانزیتی هم (نقطه 1 را ببینید) منتقل می‌شود، بنابراین وضعیت نادرست هسته در کل زنجیره قابل مشاهده است.

---

13. تنظیمات پنل

بخش «تنظیمات» (عنوان صفحه — **تنظیمات**، انگلیسی *Panel Settings*) رفتار خودِ وب‌پنل 3X-UI را مدیریت می‌کند: اینکه روی چه آدرس و پورتی گوش می‌دهد، چگونه محافظت می‌شود، چگونه با ربات Telegram و سرویس‌های خارجی تعامل دارد، و در چه منطقهٔ زمانی‌ای وظایف زمان‌بندی‌شده را اجرا می‌کند. هر پارامتر در جدول `settings` پایگاه داده به‌صورت جفت «کلید — مقدار» نگه‌داری می‌شود؛ اگر مقدار در پایگاه داده موجود نباشد، مقدار پیش‌فرض اعمال می‌گردد.

> **مهم — اعمال تغییرات.** هر تغییری در این صفحه باید با دکمهٔ **ذخیره** (*Save*) ذخیره شود و سپس پنل برای اعمال تغییرات راه‌اندازی مجدد گردد. راهنمای دقیق: «تغییرات را ذخیره کنید و پنل را برای اعمال آن‌ها راه‌اندازی مجدد کنید.» هنگام ذخیره، اعلان «تنظیمات تغییر کرد» نمایش داده می‌شود.

13.1. ذخیره و راه‌اندازی مجدد پنل

عنصر	کارکرد
ذخیره (*Save*)	همهٔ فیلدهای فرم را در پایگاه داده می‌نویسد (`POST /panel/setting/update`). پیش از نوشتن، مقادیر اعتبارسنجی می‌شوند — آدرس‌ها، پورت‌ها یا مسیرهای نادرست رد می‌شوند و پنل خطا بازمی‌گرداند.
راه‌اندازی مجدد پنل (*Restart Panel*)	وب‌سرور پنل را راه‌اندازی مجدد می‌کند (`POST /panel/setting/restartPanel`). راه‌اندازی مجدد با تأخیر 3 ثانیه انجام می‌شود. راهنما: «آیا مطمئن هستید که می‌خواهید پنل را راه‌اندازی مجدد کنید؟ تأیید کنید و راه‌اندازی مجدد پس از 3 ثانیه انجام می‌شود. اگر پنل در دسترس نبود، لاگ سرور را بررسی کنید». در صورت موفقیت — «پنل با موفقیت راه‌اندازی مجدد شد».
بازنشانی به مقادیر پیش‌فرض (*Reset to Default*)	همهٔ تنظیمات ذخیره‌شده در پایگاه داده را حذف می‌کند، پس از آن پنل از مقادیر پیش‌فرض استفاده می‌کند. اطلاعات ورود مدیر با این عملیات بازنشانی نمی‌شود.

راه‌اندازی مجدد با ارسال سیگنال `SIGHUP` به فرایند پنل (یا از طریق هوک ثبت‌شدهٔ راه‌اندازی مجدد) انجام می‌شود. در ویندوز راه‌اندازی مجدد خودکار از طریق سیگنال پشتیبانی نمی‌شود. **تغییرات پارامترهای گوش‌دادن (IP، پورت، مسیر، دامنه، گواهی‌ها، منطقهٔ زمانی) فقط پس از راه‌اندازی مجدد پنل اعمال می‌شوند.**

13.2. تنظیمات عمومی (زبانهٔ «پنل» / *General*)

زبان رابط کاربری (*Language*)

زبان وب‌رابط پنل. زبان‌های در دسترس: `en-US` (انگلیسی)، `ru-RU` (روسی)، `zh-CN` ، `zh-TW` ، `fa-IR` ، `ar-EG` ، `es-ES` ، `id-ID` ، `ja-JP` ، `pt-BR` ، `tr-TR` ، `uk-UA` ، `vi-VN` . این یک تنظیم نمایشی است و بر کار Xray تأثیری ندارد.

نوع تقویم (*Calendar Type*)

- **کلید:** `datepicker`
- **مقدار پیش‌فرض:** `gregorian` (میلادی).
- **کارکرد:** نوع تقویم مورد استفاده در انتخاب تاریخ‌ها (برای مثال، هنگام تعیین مدت اعتبار کلاینت‌ها). راهنما: «وظایف زمان‌بندی‌شده مطابق این تقویم اجرا خواهند شد.» مقدار جایگزین — تقویم فارسی (جلالی) است که در میان مخاطبان ایرانی پنل پرطرفدار است.

اندازهٔ صفحه‌بندی (*Pagination Size*)

- **کلید:** `pageSize`
- **مقدار پیش‌فرض:** `25`
- **مقادیر مجاز:** عدد صحیح از `0` تا `1000`.
- **کارکرد:** تعداد سطرها در هر صفحه از جدول‌ها (فهرست اتصال‌ها/inbound). راهنما: «اندازهٔ صفحه را برای جدول اتصال‌ها تعیین کنید. برای غیرفعال‌سازی روی 0 بگذارید» — با `0` خروجی صفحه‌بندی‌شده غیرفعال می‌شود و همهٔ رکوردها به‌صورت یک فهرست واحد نمایش داده می‌شوند.
- **راه‌اندازی مجدد پنل لازم نیست** (تنظیم نمایشی).

راه‌اندازی مجدد Xray پس از غیرفعال‌سازی خودکار (*Restart Xray After Auto Disable*)

- **کلید:** `restartXrayOnClientDisable`
- **مقدار پیش‌فرض:** `true`
- **کارکرد:** هنگام غیرفعال‌سازی خودکار کلاینت (به‌دلیل پایان مدت اعتبار یا رسیدن به حد ترافیک)، Xray راه‌اندازی مجدد می‌شود تا اتصال‌های قبلاً برقرارشدهٔ آن کلاینت قطع گردند. راهنما: «وقتی کلاینت به‌دلیل پایان مدت اعتبار یا حد ترافیک به‌طور خودکار غیرفعال می‌شود، Xray را راه‌اندازی مجدد کن.» خودِ این قابلیت تغییری نکرده است — کلید تغییر فقط روی زبانهٔ «پنل» (*General*) در کنار سایر تنظیمات عمومی قرار دارد.

مدل نام و نویسهٔ جداکننده (*Remark Model & Separation Character*)

- **کلید:** `remarkModel`
- **مقدار پیش‌فرض:** `-ieo`
- **کارکرد:** تعیین می‌کند که نام (remark) پیکربندی در اشتراک چگونه ساخته شود. این رشته از **نخستین نویسه** — جداکننده، و سپس **دنباله‌ای از حروفِ ترتیب** تشکیل می‌شود:
  - `i` — نام inbound (*inbound remark*)؛
  - `e` — ایمیل کلاینت؛
  - `o` — برچسب اضافی (*extra*).

با مقدار پیش‌فرض `-ieo` جداکننده `-` است و ترتیب بخش‌ها چنین است: extra ← email ← inbound (برای مثال، `MyInbound-user@mail-extra`). بخش‌های خالی نادیده گرفته می‌شوند. فیلد «نمونهٔ نام» (*Sample Remark*) در رابط کاربری پیش‌نمایش نامِ ساخته‌شده را نشان می‌دهد. گنجاندن ایمیل در نام علاوه بر این به پارامتر «گنجاندن Email در عنوان» در تنظیمات اشتراک (بخش مربوط به اشتراک) نیز بستگی دارد.

مثال: چگونه مقدار `remarkModel` بر نام پیکربندی اثر می‌گذارد. فرض کنید نام inbound برابر `VLESS-Reality`، ایمیل کلاینت `alex@vpn` و برچسب اضافی `RU` است. در این صورت:

مقدار فیلد	نام نهایی (remark)
`-ieo` (پیش‌فرض)	`VLESS-Reality-alex@vpn-RU`
`_ie`	`VLESS-Reality_alex@vpn`
`-ei`	`alex@vpn-VLESS-Reality`
`o` (فاصله به‌عنوان جداکننده، فقط برچسب)	`RU`

نخستین نویسهٔ رشته همیشه جداکننده است؛ بقیهٔ حروف تعیین می‌کنند چه بخش‌هایی و با چه ترتیبی وارد نام شوند.

13.3. دسترسی به پنل: IP، پورت، مسیر، دامنه، گواهی

این گروه نقطهٔ ورود شبکه‌ای پنل را تعیین می‌کند. **همهٔ تغییرات اینجا فقط پس از راه‌اندازی مجدد پنل اعمال می‌شوند.**

فیلد	کلید	مقدار پیش‌فرض	توضیح
آدرس IP برای مدیریت پنل (*Listen IP*)	`webListen`	`""` (خالی)	IP ای که وب‌پنل روی آن گوش می‌دهد. خالی = گوش‌دادن روی همهٔ IP ها. راهنما: «برای اتصال از هر IP خالی بگذارید». اگر تعیین شود، باید یک آدرس IP معتبر باشد (در غیر این صورت ذخیره رد می‌شود).
دامنهٔ پنل (*Listen Domain*)	`webDomain`	`""` (خالی)	نام دامنهٔ پنل برای بررسی درخواست بر اساس دامنه. خالی = پذیرش اتصال از هر دامنه و IP. راهنما: «برای اتصال از هر دامنه و IP خالی بگذارید.»
پورت پنل (*Listen Port*)	`webPort`	`2053`	پورتی که پنل روی آن کار می‌کند. راهنما: «پورتی که پنل روی آن کار می‌کند». مقدار مجاز `1–65535` . پورت باید آزاد باشد؛ پنل و سرویس اشتراک نمی‌توانند همزمان از جفت یکسان `IP:پورت` استفاده کنند.
مسیر URI (*URI Path*)	`webBasePath`	`/`	مسیر پایهٔ URL پنل (basePath). راهنما: «باید با '/' شروع و با '/' تمام شود». هنگام ذخیره، پنل به‌طور خودکار `/` ابتدایی و انتهایی را در صورت نبودن اضافه می‌کند. نویسه‌های ممنوعه در مسیر رد می‌شوند.

گواهی پنل (TLS / HTTPS)

فیلد	کلید	مقدار پیش‌فرض	توضیح
مسیر فایل کلید عمومی گواهی پنل (*Public Key Path*)	`webCertFile`	`""`	مسیر کامل فایل گواهی (زنجیره). راهنما: «مسیر کامل را که با '/' شروع می‌شود وارد کنید».
مسیر فایل کلید خصوصی گواهی پنل (*Private Key Path*)	`webKeyFile`	`""`	مسیر کامل فایل کلید خصوصی. راهنما: «مسیر کامل را که با '/' شروع می‌شود وارد کنید».

اگر **دست‌کم یکی** از مسیرهای گواهی/کلید تعیین شود، پنل هنگام ذخیره تلاش می‌کند جفت «گواهی + کلید» را بارگذاری کند؛ در صورت خطا (فایل ناموجود، عدم تطابق کلید و گواهی) ذخیره رد می‌شود. وقتی هر دو مسیر درست تعیین شوند، پنل به HTTPS منتقل می‌شود. خالی‌بودن هر دو فیلد = پنل با HTTP معمولی کار می‌کند.

> **هشدارهای امنیتی** (*Security warnings*). پنل بلوک «پنل شما ممکن است در معرض دسترسی باشد:» را با هشدارهایی نمایش می‌دهد، اگر پیکربندی ناامن تشخیص دهد:
>
> - کار با HTTP معمولی — «برای محیط عملیاتی TLS را تنظیم کنید»؛
> - پورت استاندارد 2053 — «آن را به یک پورت تصادفی تغییر دهید»؛
> - مسیر پایهٔ پیش‌فرض `/` — «آن را به یک مسیر تصادفی تغییر دهید»؛
> - مسیر استاندارد اشتراک `/sub/` و اشتراک JSON ای `/json/` — «آن را تغییر دهید». این‌ها توصیه هستند، نه مسدودسازی.

13.4. نشست، پراکسی پنل و پراکسی‌های مورد اعتماد (زبانهٔ «پراکسی و سرور» / *Proxy and Server*)

مدت نشست (*Session Duration*)

- **کلید:** `sessionMaxAge`
- **مقدار پیش‌فرض:** `360` (دقیقه، یعنی 6 ساعت).
- **مقادیر مجاز:** از `1` تا `525600` دقیقه (1 سال).
- **کارکرد:** مدتی که مدیر بدون ورود مجدد وارد سیستم باقی می‌ماند. واحد — **دقیقه** است. راهنما: «مدت نشست در سیستم (مقدار: دقیقه)».

outbound برای ترافیک پنل (*Panel Traffic Outbound*)

- **کلید:** `panelOutbound`
- **مقدار پیش‌فرض:** `""` (خالی = اتصال مستقیم).
- **کارکرد:** یک **outbound** ی از Xray را تعیین می‌کند که پنل از طریق آن **درخواست‌های خود** را می‌فرستد — بررسی نسخه‌ها و دانلود پنل/Xray، ارتباط با Telegram، به‌روزرسانی معمول فایل‌های geo — تا فیلترینگ سمت سرور GitHub/Telegram دور زده شود. این فیلد یک **فهرست کشویی** است: در آن outbound های قالب پیکربندی Xray، outbound های اشتراک‌های outbound، و همچنین **توازن‌گرهای** مسیریابی (به‌صورت گروه جداگانه) فهرست شده‌اند. outbound های نوع `blackhole` از فهرست کنار گذاشته شده‌اند — مسیریابی دانلود به «سیاه‌چاله» بی‌معنی است. راهنمای دقیق: «درخواست‌های خودِ پنل را — بررسی نسخه‌ها و دانلودهای پنل/Xray، Telegram و به‌روزرسانی معمول فایل‌های geo — از طریق این outbound از Xray مسیریابی می‌کند تا فیلترینگ سمت سرور GitHub/Telegram دور زده شود. یک پل-inbound محلی به‌طور خودکار به پیکربندی در حال اجرا افزوده می‌شود و در لحظه اعمال می‌گردد. به‌روزرسانی خودکار Geodata تعبیه‌شده در Xray تحت تأثیر قرار نمی‌گیرد؛ outbound مخصوص خود را برای دانلود دارد. برای اتصال مستقیم خالی بگذارید.»

> **چگونه کار می‌کند.** هنگام انتخاب یک outbound، پنل خودش یک loopback-inbound خدماتی (پل SOCKS با تگ `panel-egress`) و یک قاعدهٔ مسیریابی را به پیکربندی در حال اجرا می‌افزاید که ترافیک HTTP خودِ پنل را به outbound انتخاب‌شده هدایت می‌کند. اگر توازن‌گری انتخاب شود، `balancerTag` در قاعده جای‌گذاری می‌شود و ترافیک پنل میان اعضای آن توزیع می‌گردد. پل و قاعده **در لحظه** اعمال می‌شوند، بدون راه‌اندازی مجدد کامل پنل. برای اتصال مستقیم فیلد را خالی بگذارید. به‌روزرسانی خودکار داده‌های geo تعبیه‌شده در Xray با این تنظیم تحت تأثیر **قرار نمی‌گیرد** — outbound مخصوص خود را درون مسیریابی Xray دارد.

- **قالب:** `socks5://` (یا `socks5h://`) یا `http(s)://`، در صورت نیاز همراه با احراز هویت به شکل `socks5://user:pass@host:port`. طرح‌های پشتیبانی‌شده دقیقاً عبارت‌اند از: `socks5`، `socks5h`، `http`، `https` — طرح‌های دیگر نامعتبر محسوب می‌شوند و پنل آنگاه به اتصال مستقیم بازمی‌گردد. مثال متداول — یک SOCKS-inbound محلی خودِ Xray است.
- راهنمای دقیق: «درخواست‌های خروجی خودِ پنل (به‌روزرسانی geo، بررسی نسخه‌های Xray/پنل، Telegram) را از طریق این پراکسی مسیریابی می‌کند تا فیلترینگ سمت سرور GitHub/Telegram دور زده شود. socks5:// یا http(s):// را می‌پذیرد، مثلاً SOCKS-inbound محلی Xray. برای اتصال مستقیم خالی بگذارید.»
- یک پراکسی نامعتبر باعث خطای ذخیره نمی‌شود — پنل صرفاً از اتصال مستقیم استفاده می‌کند و یک هشدار در لاگ می‌نویسد.

مثال مقادیر فیلد. اگر روی سرور پیش‌تر یک SOCKS-inbound محلی Xray روی پورت `10808` بالا آمده باشد، درخواست‌های خودِ پنل را از طریق آن هدایت کنید:

```
socks5://127.0.0.1:10808
```

برای یک HTTP-پراکسی خارجی با احراز هویت:

```
http://user:pass@proxy.example.com:8080
```

پس از ذخیره و راه‌اندازی مجدد، پنل به‌روزرسانی پایگاه‌های geo را می‌کشد، نسخه‌ها را بررسی می‌کند و با Telegram از طریق پراکسی مشخص‌شده ارتباط می‌گیرد.

CIDR های پراکسی مورد اعتماد (*Trusted proxy CIDRs*)

- **کلید:** `trustedProxyCIDRs`
- **مقدار پیش‌فرض:** `127.0.0.1/32,::1/128` (فقط هاست محلی).
- **قالب:** فهرستی از آدرس‌های IP یا زیرشبکه‌های CIDR جداشده با ویرگول (برای مثال `10.0.0.0/8, 192.168.1.5`). هر عنصر به‌عنوان IP یا CIDR بررسی می‌شود — مقدار نادرست هنگام ذخیره رد می‌شود.
- **کارکرد:** منابعی را فهرست می‌کند که مجازند هدرهای `X-Forwarded-Host`، `X-Forwarded-Proto` و هدر IP واقعی کلاینت را تنظیم کنند. راهنمای دقیق: «IP/CIDR های جداشده با ویرگول که مجازند هدرهای proto، forwarded host و client IP را تنظیم کنند.» اگر پنل پشت یک پراکسی معکوس (Caddy، nginx و مانند آن) کار می‌کند باید تنظیم شود تا IP کلاینت و طرح به‌درستی تشخیص داده شوند.

مثال: پنل پشت یک پراکسی معکوس. اگر nginx روی همان هاست قرار دارد و درخواست‌ها را به پنل پراکسی می‌کند، اعتماد را فقط به هاست محلی نگه دارید (مقدار پیش‌فرض):

```
127.0.0.1/32,::1/128
```

اگر پراکسی روی سرور جداگانه‌ای در شبکهٔ داخلی `10.0.0.0/8` قرار دارد، زیرشبکهٔ آن را اضافه کنید، در غیر این صورت پنل هدرهای ارسالی آن را نادیده می‌گیرد و به‌جای کلاینت واقعی IP پراکسی را می‌بیند:

```
127.0.0.1/32,::1/128,10.0.0.0/8
```

نمونهٔ بلوک متناظر nginx که IP واقعی و طرح را منتقل می‌کند:

```
proxy_set_header X-Real-IP        $remote_addr;
proxy_set_header X-Forwarded-For  $proxy_add_x_forwarded_for;
proxy_set_header X-Forwarded-Proto $scheme;
proxy_set_header X-Forwarded-Host $host;
```

13.5. ربات Telegram (زبانهٔ «ربات Telegram» / *Telegram Bot*)

فعال‌سازی ربات Telegram (*Enable Telegram Bot*)

- **کلید:** `tgBotEnable`
- **نوع/پیش‌فرض:** بولی، `false`.
- **کارکرد:** کار ربات Telegram را فعال می‌کند. راهنما: «دسترسی به قابلیت‌های پنل از طریق ربات Telegram».

توکن Telegram (*Telegram Token*)

- **کلید:** `tgBotToken`
- **پیش‌فرض:** `""`.
- **کارکرد:** توکن ربات. راهنما: «باید توکن را از مدیر ربات‌های Telegram یعنی @botfather دریافت کنید».

- **ویژگی امنیتی:** توکن از مقادیر محرمانه است. در پاسخ پنل به خواندن تنظیمات بازگردانده نمی‌شود (فیلد پاک می‌شود و فقط پرچم «تنظیم‌شده/تنظیم‌نشده» داده می‌شود). اگر هنگام ذخیره فیلد را خالی بگذارید، توکن قبلاً ذخیره‌شده **حفظ می‌شود** (پاک نمی‌شود).

زبان ربات Telegram (*Telegram Bot Language*)

- **کلید:** `tgLang`
- **پیش‌فرض:** `en-US` .
- **کارکرد:** زبان پیام‌های ربات (مستقل از زبان وب‌رابط). فهرست زبان‌های در دسترس با زبان‌های پنل یکسان است.

User ID مدیر ربات (*Admin Chat ID*)

- **کلید:** `tgBotChatId`
- **پیش‌فرض:** `""` .
- **قالب:** یک یا چند User ID عددی Telegram **جداشده با ویرگول**.
- **کارکرد:** گیرندگان اعلان‌ها و مدیرانی که اجازه دارند پنل را از طریق ربات مدیریت کنند. راهنما: «یک یا چند User ID مدیر(ان) ربات Telegram. برای دریافت User ID از @userinfobot یا دستور '/id' در ربات استفاده کنید.»

تناوب اعلان‌ها (*Notification Time*)

- **کلید:** `tgRunTime`
- **پیش‌فرض:** `@daily` (روزی یک‌بار).
- **قالب:** رشته‌ای در قالب **Crontab** (هم عبارت‌های استاندارد cron و هم اختصارهایی مانند `@daily` ، `@hourly` ، `@every 1h` پشتیبانی می‌شوند). راهنما: «بازهٔ اعلان‌ها را در قالب Crontab مشخص کنید». گزارش‌های دوره‌ای ربات را مدیریت می‌کند.

مثال‌های مقادیر فیلد.

مقدار	چه زمانی ربات گزارش می‌فرستد
`@daily`	روزی یک‌بار در نیمه‌شب (پیش‌فرض)
`@hourly`	هر ساعت
`@every 6h`	هر 6 ساعت
`0 9 * * *`	روزانه در ساعت 09:00
`30 8 * * 1`	هر دوشنبه در ساعت 08:30

زمان در منطقهٔ زمانی از تنظیم «منطقهٔ زمانی» (بند 13.6) محاسبه می‌شود.

پراکسی SOCKS (*SOCKS Proxy*)

- **کلید:** `tgBotProxy`
- **پیش‌فرض:** `""` .
- **کارکرد:** پراکسی SOCKS5 جداگانه برای اتصال ربات به Telegram. راهنما: «اگر برای اتصال به Telegram به پراکسی Socks5 نیاز دارید، پارامترهای آن را مطابق راهنما تنظیم کنید.» دقیقاً بر ترافیک ربات اعمال می‌شود (با «پراکسی شبکه‌ای پنل» عمومی از بند 13.4 فرق دارد).

Telegram API Server (*Telegram API Server*)

- **کلید:** `tgBotAPIServer`
- **پیش‌فرض:** `""` (استفاده از سرور استاندارد `api.telegram.org`).
- **قالب:** آدرس `http(s)://...`؛ هنگام ذخیره بررسی صحت URL انجام می‌شود — آدرس نامعتبر رد می‌شود. راهنما: «سرور API ای از Telegram که استفاده می‌شود. برای استفاده از سرور پیش‌فرض خالی بگذارید.» برای یک Telegram Bot API server که خودتان مستقر کرده‌اید لازم است.

اعلان‌های ربات (گروه «اعلان‌ها» / *Notifications*)

فیلد	کلید	پیش‌فرض	توضیح
پشتیبان‌گیری از پایگاه داده (*Database Backup*)	`tgBotBackup`	`false`	ارسال فایل پشتیبان پایگاه داده به Telegram همراه با گزارش. راهنما: «اعلانی به همراه فایل پشتیبان پایگاه داده ارسال کن».
اعلان ورود (*Login Notification*)	`tgBotLoginNotify`	`true`	اعلان هنگام تلاش برای ورود به پنل. راهنما: «نام کاربری، آدرس IP و زمانی را که کسی تلاش می‌کند به پنل شما وارد شود نمایش می‌دهد.»
تأخیر اعلان انقضای نشست (*Expiration Date Notification*)	`expireDiff`	`0`	چند **روز** پیش از پایان مدت اعتبار کلاینت اعلان فرستاده شود. `0` — غیرفعال. مقدار مجاز `>= 0`. راهنما: «دریافت اعلان دربارهٔ انقضای مدت نشست پیش از رسیدن به مقدار آستانه (مقدار: روز)».
آستانهٔ ترافیک برای اعلان (*Traffic Cap Notification*)	`trafficDiff`	`0`	آستانهٔ ترافیک باقی‌مانده برای اعلان. راهنما: «دریافت اعلان دربارهٔ اتمام ترافیک پیش از رسیدن به آستانه (مقدار: GB)». مقدار مجاز `0–100`.
آستانهٔ بار CPU (*CPU Load Notification*)	`tgCpu`	`80`	اعلان به مدیران در صورتی که بار CPU از آستانه فراتر رود (به ٪). مقدار مجاز `0–100`. راهنما: «اعلان به مدیران در Telegram در صورتی که بار CPU از این آستانه فراتر رود (مقدار: ٪)».

13.6. تاریخ و زمان (زبانهٔ «تاریخ و زمان» / *Date and Time*)

منطقهٔ زمانی (*Time Zone*)

- **کلید:** `timeLocation`
- **مقدار پیش‌فرض:** `Local` (منطقهٔ زمانی سیستمی سرور).
- **قالب:** نام منطقه از پایگاه دادهٔ IANA tz (برای مثال، `Europe/Moscow`، `UTC`، `Asia/Tehran`).
- **کارکرد:** منطقهٔ زمانی‌ای که پنل وظایف زمان‌بندی‌شده را در آن اجرا می‌کند (گزارش‌های ربات، بازنشانی/بررسی ترافیک، انقضای مدت‌ها). راهنما: «وظایف زمان‌بندی‌شده مطابق زمان این منطقهٔ زمانی اجرا می‌شوند».
- **اعتبارسنجی:** هنگام ذخیره منطقه بررسی می‌شود — منطقهٔ ناموجود رد می‌شود. اگر بعداً در پایگاه داده مقدار نادرستی قرار گیرد، پنل در زمان اجرا به `Local` بازمی‌گردد، و در صورت در دسترس نبودن آن نیز — به `UTC`.

13.7. ترافیک خارجی و رفتار Xray (زبانهٔ «ترافیک خارجی» / *External Traffic*)

فیلد	کلید	پیش‌فرض	توضیح
اطلاع‌رسانی ترافیک خارجی (*External Traffic Inform*)	`externalTrafficInformEnable`	`false`	اطلاع‌رسانی به API خارجی در هر به‌روزرسانی ترافیک. راهنما: «در هر به‌روزرسانی ترافیک به API خارجی اطلاع بده.»

فیلد	کلید	پیش‌فرض	توضیح
URI اطلاع‌رسانی ترافیک خارجی (*External Traffic Inform URI*)	`externalTrafficInformURI`	`""`	آدرسی که پنل به‌روزرسانی‌های ترافیک را به آن می‌فرستد. هنگام ذخیره بررسی صحت URL انجام می‌شود. راهنما: «به‌روزرسانی‌های ترافیک به این URI فرستاده می‌شوند».
راه‌اندازی مجدد Xray پس از غیرفعال‌سازی خودکار (*Restart Xray After Auto Disable*)	`restartXrayOnClientDisable`	`true`	راه‌اندازی مجدد Xray هنگامی که کلاینت به‌دلیل پایان مدت یا فراتر رفتن از حد ترافیک به‌طور خودکار غیرفعال می‌شود. راهنما: «وقتی کلاینت به‌دلیل پایان مدت اعتبار یا حد ترافیک به‌طور خودکار غیرفعال می‌شود، Xray را راه‌اندازی مجدد کن.» **کلید تغییر روی زبانهٔ «پنل» (*General*) قرار دارد** — بند 13.2 را ببینید؛ اینجا برای کامل‌بودن آورده شده است.

13.8. متفرقه: قالب پیکربندی Xray و URL بررسی

قالب پیکربندی Xray (*xrayTemplateConfig*)

- **کلید:** `xrayTemplateConfig`
- **پیش‌فرض:** قالب JSON تعبیه‌شده (embedded) که همراه با بیلد عرضه می‌شود.
- **کارکرد:** قالب پایهٔ JSON پیکربندی Xray-core که پنل inbound/outbound را روی آن می‌سازد. این مقدار در خروجی معمول همهٔ تنظیمات **بازگردانده نمی‌شود** و در صفحهٔ جداگانهٔ پیکربندی Xray ویرایش می‌شود، نه در فهرست عمومی فیلدهای تنظیمات پنل. قالب استاندارد پیش‌فرض از طریق `GET /panel/setting/getDefaultJsonConfig` در دسترس است.

URL بررسی outbound ها (*xrayOutboundTestUrl*)

- **کلید:** `xrayOutboundTestUrl`
- **پیش‌فرض:** `https://www.google.com/generate_204`
- **کارکرد:** آدرسی که هنگام بررسی سلامت اتصال‌های خروجی (outbound) استفاده می‌شود. هنگام تنظیم به‌عنوان یک HTTP(S)-URL پاک‌سازی (sanitize) می‌شود.

13.9. حساب مدیر و توکن‌های API

این پارامترها روی زبانهٔ مجاور («حساب کاربری» / *Authentication*) قرار دارند و به‌تفصیل در بخش امنیت بررسی شده‌اند؛ اینجا — خلاصه‌ای کوتاه از کلیدها آمده است.

- **تغییر اطلاعات ورود** (فیلدهای «نام کاربری فعلی»، «گذرواژهٔ فعلی»، «نام کاربری جدید»، «گذرواژهٔ جدید») با یک درخواست جداگانهٔ `POST /panel/setting/updateUser` ذخیره می‌شود. نام کاربری و گذرواژهٔ فعلیِ درست لازم است؛ نام کاربری و گذرواژهٔ جدید نباید خالی باشند. پیام‌ها: «اطلاعات ورود مدیر را با موفقیت تغییر دادید.» / «نام کاربری یا گذرواژه نادرست است».
- **احراز هویت دومرحله‌ای (2FA)** — کلیدهای `twoFactorEnable` (پیش‌فرض `false`) و کلید محرمانهٔ `twoFactorToken`. توکن یک راز است: هنگام فعال‌بودن 2FA، فیلد خالی هنگام ذخیره توکن موجود را پاک نمی‌کند. هنگام **نخستین** فعال‌سازی 2FA، پنل نشست‌های فعلی را باطل می‌کند («دورهٔ ورود» بالا می‌رود).
- **توکن‌های API** با اندپوینت‌های جداگانه (`/panel/setting/apiTokens…`) مدیریت می‌شوند: فهرست، ساخت (`apiTokens/create`)، حذف، فعال/غیرفعال‌سازی. خودِ توکن **فقط یک‌بار هنگام ساخت** نمایش داده می‌شود و به‌صورت خوانا ذخیره نمی‌شود: «این توکن را اکنون کپی کنید. به‌دلایل امنیتی به‌صورت خوانا ذخیره نمی‌شود و دیگر نمایش داده نخواهد شد.»

جزئیات مربوط به 2FA، گذرواژه‌ها، همگام‌سازی LDAP و قالب‌های اشتراک (mux، noises، fragmentation، JSON/Clash) به بخش‌های جداگانهٔ متناظر راهنما منتقل شده‌اند.

13.10. تغییرات API در 3.3.0 (مهم برای یکپارچه‌سازی‌ها)

در نسخهٔ 3.3.0 ساختار مسیرهای API سمت سرور تغییر کرده است. اگر یکپارچه‌سازی‌های خارجی دارید (اسکریپت‌ها، ربات‌ها، پنل‌های مرکزی، وظایف CI) که از طریق HTTP با پنل ارتباط می‌گیرند، باید آن‌ها را **اصلاح کنید**، در غیر این صورت از کار خواهند افتاد.

⚠️ BREAKING CHANGE: اندپوینت‌های `/panel/setting/*` و `/panel/xray/*` به زیر `/panel/api` منتقل شدند

پیش‌تر مدیریت تنظیمات پنل و پیکربندی Xray جداگانه و زیر مسیرهای `/panel/setting/*` و `/panel/xray/*` قرار داشت. اکنون هر دو مجموعه درون گروه API مشترک `/panel/api` ثبت شده‌اند. مسیرهای قدیمی **به‌طور کامل حذف شده‌اند** — درخواست به آن‌ها کد 404 بازمی‌گرداند.

چرا این کار انجام شد: کل گروه `/panel/api` از یک بررسی دسترسی واحد عبور می‌کند، یعنی این اندپوینت‌ها اکنون همان هدر `Authorization: Bearer <token>` را می‌پذیرند که بقیهٔ API می‌پذیرد. توکن API یک دسترسی کامل مدیریتی است، و بدین ترتیب کل سطح API یکدست شد.

آنچه تغییر نکرده است: صفحات وب‌رابط (مسیرهای SPA) یعنی `/panel/settings` و `/panel/xray` سر جای خود باقی مانده‌اند — صحبت فقط از اندپوینت‌های API سمت سرور است.

جدول تطابق مسیرها (قدیمی ← جدید)

پیشوند همهٔ مسیرهای زیر — صرفاً `api/` بعد از `/panel/` افزوده شده است.

بود (x.3.2 ⩽)	شد (3.3.0)	متد
`/panel/setting/all`	`/panel/api/setting/all`	POST
`/panel/setting/defaultSettings`	`/panel/api/setting/defaultSettings`	POST
`/panel/setting/update`	`/panel/api/setting/update`	POST
`/panel/setting/updateUser`	`/panel/api/setting/updateUser`	POST
`/panel/setting/restartPanel`	`/panel/api/setting/restartPanel`	POST
`/panel/setting/getDefaultJsonConfig`	`/panel/api/setting/getDefaultJsonConfig`	GET
`/panel/setting/apiTokens`	`/panel/api/setting/apiTokens`	GET
`/panel/setting/apiTokens/create`	`/panel/api/setting/apiTokens/create`	POST
`/panel/setting/apiTokens/delete/:id`	`/panel/api/setting/apiTokens/delete/:id`	POST
`/panel/setting/apiTokens/setEnabled/:id`	`/panel/api/setting/apiTokens/setEnabled/:id`	POST
`/panel/xray/`	`/panel/api/xray/`	POST
`/panel/xray/update`	`/panel/api/xray/update`	POST
`/panel/xray/getDefaultJsonConfig`	`/panel/api/xray/getDefaultJsonConfig`	GET
`/panel/xray/getXrayResult`	`/panel/api/xray/getXrayResult`	GET

بود (x.3.2 ⩽)	شد (3.3.0)	متد
`/panel/xray/getOutboundsTraffic`	`/panel/api/xray/getOutboundsTraffic`	GET
`/panel/xray/resetOutboundsTraffic`	`/panel/api/xray/resetOutboundsTraffic`	POST
`/panel/xray/testOutbound`	`/panel/api/xray/testOutbound`	POST
`/panel/xray/warp/:action`	`/panel/api/xray/warp/:action`	POST
`/panel/xray/nord/:action`	`/panel/api/xray/nord/:action`	POST
`/panel/xray/outbound-subs` (و `/outbound-subs/*`)	`/panel/api/xray/outbound-subs` (و `/outbound-subs/*`)	GET/POST/DELETE

خودِ نام‌های زیرمسیرها، بدنهٔ درخواست‌ها و قالب پاسخ‌ها تغییری نکرده‌اند — **فقط پیشوند** عوض شده است.

چگونه یکپارچه‌سازی‌های موجود را تعمیر کنیم

1. در اسکریپت‌ها/پیکربندی‌های خود همهٔ موارد `/panel/setting/` و `/panel/xray/` را پیدا کنید.
2. پیشوند را جایگزین کنید: بلافاصله پس از `/panel/` عبارت `api/` را اضافه کنید (برای مثال، `/panel/setting/all` ← `/panel/api/setting/all`).
3. بدنهٔ درخواست‌ها، پارامترها و قالب پاسخ‌ها نیازی به اصلاح ندارند — فقط URL تغییر می‌کند.
4. از آنجا که تنظیمات و پیکربندی Xray اکنون زیر `/panel/api` قرار دارند، می‌توان (و باید) با همان توکن API یعنی `Authorization: Bearer <token>` به آن‌ها دسترسی داشت که به `/panel/api/inbounds/*` و سایر اندپوینت‌ها دسترسی پیدا می‌کنید. CSRF-middleware را فراموش نکنید که روی کل گروه `/panel/api` فعال است.

مثال: خواندن همهٔ تنظیمات از طریق API. پیش‌تر (x.3.2 ⩽):

```
curl -sk -X POST "https://panel.example.com:2053/MyPath/panel/setting/all" \
  -H "Authorization: Bearer <token>"
```

اکنون (3.3.0) — `/api` بعد از `/panel/` افزوده شد:

```
curl -sk -X POST "https://panel.example.com:2053/MyPath/panel/api/setting/all" \
  -H "Authorization: Bearer <token>"
```

به‌همین ترتیب راه‌اندازی مجدد پنل: `POST /panel/api/setting/restartPanel`. مسیر قدیمی `/panel/setting/restartPanel` اکنون کد 404 بازمی‌گرداند.

API نوع‌دار: شِماها و مستندات (Swagger / OpenAPI)

در 3.3.0 مشخصات OpenAPI به‌طور کامل نوع‌دار شد. پیش‌تر پاسخ‌های نوع‌دار با یک شیء خالی `{}` توصیف می‌شدند؛ اکنون مؤلفه‌ها و شِماها (`components.schemas`) مستقیماً از مدل‌های داده تولید می‌شوند. به‌لطف این:

- Swagger UI مدل‌های واقعی داده را نشان می‌دهد، نه جای‌گزین‌های بی‌چهره.
- مولدهای خارجی (`openapi-generator` و مانند آن) می‌توانند بر اساس مشخصات، کلاینت‌های آمادهٔ زبان دلخواه را بسازند.
- به هر پاسخ نوع‌دار یک `$ref` به یک مدل مشخص پیوست شده و نمونه‌های پاسخ ضمیمه شده‌اند.

کجا مستندات API را ببینیم:

- **صفحهٔ Swagger تعبیه‌شده.** در منوی پنل — گزینهٔ **«مستندات API»** (مسیر SPA یعنی `/panel/api-docs`). اینجا همهٔ اندپوینت‌ها به‌صورت تعاملی همراه با توضیحات، بدنهٔ درخواست‌ها و نمونه‌های پاسخ فهرست شده‌اند.
- **مشخصات خام OpenAPI 3.0** از آدرس `/panel/api/openapi.json` ارائه می‌شود. این URL را می‌توان مستقیماً به Postman، Insomnia یا `openapi-generator` داد. مشخصات در مرحلهٔ بیلد در باینری تعبیه شده است؛ هنگام کار پنل تحت `webBasePath` غیراستاندارد، فیلد `servers` در مشخصات به‌طور خودکار بر اساس مسیر پایهٔ جاری بازنویسی می‌شود تا دکمهٔ «Try it out» و مولدهای خارجی به پیشوند درست برخورد کنند.

---

14. ربات تلگرام

پنل 3X-UI دارای یک ربات تلگرام داخلی است که از طریق آن می‌توان اعلان‌هایی درباره وضعیت سرور و کلاینت‌ها دریافت کرد و برخی کلاینت‌ها را مستقیماً از طریق پیام‌رسان مدیریت نمود. ربات بر پایه فناوری long polling (نظرسنجی مداوم از تلگرام) کار می‌کند، بنابراین نیازی به دامنه خارجی یا پورت باز ندارد — تنها دسترسی خروجی به سرورهای تلگرام کافی است.

ربات دو نوع مخاطب را از هم تشخیص می‌دهد:

- **مدیر** — کاربری که Telegram User ID او در تنظیمات ربات مشخص شده است (فیلد «User ID مدیر ربات»). به تمام امکانات دسترسی دارد: آمار سرور، پشتیبان‌گیری، مدیریت کلاینت‌ها، راه‌اندازی مجدد Xray.
- **کلاینت** — هر کاربر دیگری که Telegram User ID او به یک کلاینت اتصال ورودی خاص متصل شده باشد (فیلد `tgId` کلاینت). فقط اطلاعات اشتراک‌های خود را می‌بیند.

مثال: اتصال کلاینت به تلگرام. برای اینکه کاربر بتواند آمار اشتراک خود را دریافت کند، Telegram User ID عددی او در فیلد `tgId` کلاینت ثبت می‌شود. در تنظیمات JSON کلاینت این چنین به نظر می‌رسد:

```
{
  "email": "ivan",
  "id": "6f1e6b1a-0c3d-4f2a-9b7e-1a2b3c4d5e6f",
  "tgId": "123456789",
  "enable": true,
  "limitIp": 2,
  "totalGB": 53687091200,
  "expiryTime": 0
}
```

پس از این، کاربر با User ID `123456789` می‌تواند از ربات `/usage ivan` را درخواست کند و آمار خود را ببیند. همین ID را مدیر می‌تواند از طریق دکمه « 👤 تنظیم کاربر تلگرام» در کارت کلاینت وارد کند — نیازی به ویرایش دستی JSON نیست.

14.1. فعال‌سازی و پیکربندی ربات

تمام پارامترهای ربات در پنل از طریق بخش **تنظیمات** → **ربات تلگرام** تعریف می‌شوند. پس از تغییر تنظیمات، کافی است آن‌ها را ذخیره کنید — پنل بلافاصله آن‌ها را اعمال می‌کند و نیازی به راه‌اندازی مجدد پنل نیست. اگر پرچم فعال‌سازی (`tgBotEnable`)، توکن، User ID مدیران یا آدرس سرور API تغییر کند، پنل به‌طور خودکار ربات را متوقف کرده و با پارامترهای جدید مجدداً راه‌اندازی می‌کند. قانون قدیمی مبنی بر لزوم راه‌اندازی مجدد پنل پس از تغییر توکن دیگر معتبر نیست.

فیلد (UI)	کلید تنظیم	مقدار پیش‌فرض	توضیح
فعال کردن ربات تلگرام	`tgBotEnable`	`false`	کلید اصلی. راهنما: «دسترسی به امکانات پنل از طریق ربات تلگرام». تا زمانی که غیرفعال است، ربات راه‌اندازی نمی‌شود و وظایف اعلان برنامه‌ریزی نمی‌شوند.
توکن تلگرام	`tgBotToken`	(خالی)	توکن ربات. راهنما: «توکن را از مدیر ربات‌های تلگرام @botfather دریافت کنید». بدون توکن غیرخالی، ربات راه‌اندازی نمی‌شود.
پروکسی SOCKS	`tgBotProxy`	(خالی)	پروکسی برای اتصال به تلگرام. راهنما: «اگر برای اتصال به تلگرام به پروکسی Socks5 نیاز دارید، پارامترهای آن را طبق راهنما تنظیم کنید».
سرور API تلگرام	`tgBotAPIServer`	(خالی)	سرور API جایگزین تلگرام. راهنما: «سرور API تلگرام مورد استفاده برای استفاده از سرور پیش‌فرض خالی بگذارید».

فیلد (UI)	کلید تنظیم	مقدار پیش‌فرض	توضیح
User ID مدیر ربات	`tgBotChatId`	(خالی)	یک یا چند Telegram User ID مدیر، جداشده با کاما. راهنما: «برای دریافت User ID از @userinfobot استفاده کنید یا دستور `/id` را در ربات ارسال کنید».
دفعات اعلان برای مدیران از ربات	`tgRunTime`	`@daily`	زمان‌بندی گزارش دوره‌ای به فرمت crontab. راهنما: «فاصله اعلان‌ها را به فرمت Crontab وارد کنید».
پشتیبان‌گیری از پایگاه داده	`tgBotBackup`	`false`	راهنما: «ارسال اعلان همراه با فایل پشتیبان پایگاه داده». فایل پشتیبان را به گزارش دوره‌ای پیوست می‌کند.
اعلان ورود	`tgBotLoginNotify`	`true`	راهنما: «نام کاربری، آدرس IP و زمان را هنگامی که کسی سعی می‌کند وارد پنل شما شود نمایش می‌دهد».
آستانه بار CPU برای اعلان	`tgCpu`	`80`	آستانه بار CPU به درصد (اعتبارسنجی 0–100). راهنما: «اگر بار CPU از این آستانه بیشتر شود، به مدیران در تلگرام اطلاع می‌دهد (مقدار: %)». با مقدار 0 بررسی CPU غیرفعال می‌شود.
زبان ربات تلگرام	—	—	زبانی که ربات تمام پیام‌ها را به آن می‌سازد.

دریافت توکن از طریق @BotFather

1. در تلگرام گفتگو با **@BotFather** را باز کنید.
2. دستور `/newbot` را ارسال کرده و دستورالعمل‌ها را دنبال کنید (نام ربات و `username` منحصربه‌فرد که باید به `bot` ختم شود).
3. BotFather توکنی به شکل `123456789:AA...` ارائه می‌دهد. آن را در فیلد **توکن تلگرام** وارد کنید.

دریافت User ID مدیر

User ID یک شناسه عددی حساب است (نه username). می‌توان آن را به دو روش به دست آورد:

- پیام دادن به ربات **@userinfobot**.
- راه‌اندازی ربات پیکربندی‌شده و ارسال دستور **`/id`** — ربات ID شما را برمی‌گرداند.

عدد به دست آمده را در فیلد **User ID مدیر ربات** وارد کنید. برای تعیین چندین مدیر، IDهای آن‌ها را با کاما جدا کنید (مثلاً `11111111,22222222`). هر ID به عنوان عدد صحیح بررسی می‌شود؛ مقدار نادرست باعث خطای راه‌اندازی ربات می‌شود.

مثال: مقدار فیلد «User ID مدیر ربات». یک مدیر — فقط یک عدد:

```
123456789
```

دو مدیر جداشده با کاما (فاصله اختیاری است):

```
123456789,987654321
```

هر مقدار باید عدد صحیح باشد. فرمت `@username` یا `123 456` (با فاصله داخل عدد) قابل قبول نیست — ربات راه‌اندازی نمی‌شود.

پروکسی

طرح‌های `socks5://` ، `http://` و `https://` پشتیبانی می‌شوند. اگر فیلد پروکسی خالی باشد، ربات سعی می‌کند از پروکسی عمومی پنل استفاده کند (اگر تعریف شده و طرح آن پشتیبانی شود). URL با طرح پشتیبانی‌نشده یا سینتکس نادرست نادیده گرفته می‌شود — ربات به‌صورت مستقیم متصل می‌شود. پروکسی زمانی مفید است که دسترسی مستقیم به API تلگرام از سرور مسدود باشد.

اعلان‌های ایمیل (SMTP)

علاوه بر تلگرام، همان رویدادها را می‌توان از طریق ایمیل نیز دریافت کرد. این کانال در بخش **تنظیمات** → **Email** در تب **SMTP Settings** پیکربندی می‌شود:

فیلد (UI)	کلید تنظیم	مقدار پیش‌فرض	توضیح
Enable Email Notifications	`smtpEnable`	`false`	کلید اصلی اعلان‌های ایمیل از طریق SMTP.
SMTP Host	`smtpHost`	(خالی)	هاست سرور SMTP (مثلاً `smtp.gmail.com`).
SMTP Port	`smtpPort`	`587`	پورت سرور SMTP.
SMTP Username	`smtpUsername`	(خالی)	نام کاربری برای احراز هویت SMTP. همچنین به عنوان آدرس فرستنده (From) استفاده می‌شود.
SMTP Password	`smtpPassword`	(خالی)	رمز عبور برای احراز هویت SMTP. به‌صورت مخفی ذخیره می‌شود؛ اگر رمز از قبل تنظیم شده باشد، فیلد نشانه «تنظیم شده» را نمایش می‌دهد و می‌توان آن را خالی گذاشت تا مقدار فعلی حفظ شود.
Recipients	`smtpTo`	(خالی)	فهرست گیرندگان جداشده با کاما (مثلاً `admin@example.com, ops@example.com`).
Encryption	`smtpEncryptionType`	`starttls`	نوع رمزگذاری اتصال: `none` (بدون رمزگذاری)، `starttls` (STARTTLS) یا `tls` (TLS ضمنی).

دکمه **Send Test Email** یک ایمیل آزمایشی ارسال می‌کند و نتیجه را مرحله به مرحله نمایش می‌دهد: **Connection** (اتصال)، **Authentication** (احراز هویت) و **Send** (ارسال). در صورت بروز مشکل، تشخیص نشان می‌دهد در کدام مرحله خطا رخ داده (مثلاً «Authentication failed — check username and password» یا «Server requires STARTTLS — change encryption type»)، که انتخاب پارامترها را آسان‌تر می‌کند.

در تب دوم (**Notifications**) رویدادهایی انتخاب می‌شوند که ایمیل آن‌ها ارسال شود — با همان کارت‌های گروهی مورد استفاده در تلگرام (به «شبکه رویدادها و انتخاب اعلان‌ها» در بخش 14.5 مراجعه کنید).

سرور API تلگرام

به‌طور پیش‌فرض ربات با API رسمی تلگرام ارتباط برقرار می‌کند. در فیلد **سرور API تلگرام** می‌توان آدرس سرور Bot API شخصی (`telegram-bot-api`) را وارد کرد. URL از نظر امنیتی بررسی می‌شود؛ آدرس مسدود یا نادرست رد می‌شود و از سرور پیش‌فرض استفاده خواهد شد.

14.2. منوی اصلی و دکمه‌ها

منو با دستور `/start` فراخوانی می‌شود. دکمه‌ها یک صفحه‌کلید inline هستند که به پیام پیوست می‌شوند؛ مجموعه دکمه‌ها بستگی دارد به اینکه شما مدیر هستید یا کلاینت.

منوی مدیر

دکمه	عملکرد
گزارش مرتب‌شده مصرف ترافیک	فهرستی از تمام کلاینت‌ها مرتب شده بر اساس ترافیک، با مصرف هر یک؛ ایمیل‌های اضافی بدون داده با « نتیجه‌ای یافت نشد» مشخص می‌شوند.
وضعیت سرور	خلاصه‌ای از سرور (به بخش 14.5 مراجعه کنید). دکمه « بروزرسانی» داده‌ها را تازه می‌کند.
بازنشانی تمام ترافیک	شمارنده‌های ترافیک **تمام** کلاینت‌ها را بازنشانی می‌کند. درخواست تأیید («مطمئن هستید؟ »)، سپس برای هر کلاینت « موفق» یا « ناموفق» نمایش می‌دهد و در پایان « بازنشانی ترافیک برای تمام کلاینت‌ها انجام شد».
پشتیبان‌گیری از پایگاه داده	فایل پایگاه داده و `config.json` را ارسال می‌کند (به بخش 14.6 مراجعه کنید).
لاگ مسدودسازی	فایل‌های لاگ آدرس‌های IP مسدودشده به دلیل تجاوز از محدودیت IP را ارسال می‌کند.
اتصالات ورودی	خلاصه‌ای از تمام inbound: Remark، پورت، ترافیک، تعداد کلاینت‌ها، تاریخ انقضا.
به زودی تمام می‌شود	فهرستی از inbound و کلاینت‌هایی که ترافیک یا مهلت آن‌ها به زودی تمام می‌شود (به بخش 14.5 مراجعه کنید).
دستورات	راهنمای دستورات مدیر را نمایش می‌دهد.
آنلاین	تعداد و فهرست کلاینت‌های آنلاین؛ ضربه زدن روی ایمیل کارت کلاینت را باز می‌کند. دکمه « بروزرسانی».
تمام کلاینت‌ها	انتخاب inbound را باز می‌کند، سپس فهرست کلاینت‌های آن برای مشاهده/مدیریت.
کلاینت جدید	جادوگر افزودن کلاینت را راه‌اندازی می‌کند (انتخاب inbound → پیش‌نویس → تأیید).
تنظیمات اشتراک / لینک‌های فردی / QR-کد	انتخاب inbound و کلاینت برای دریافت لینک اشتراک، لینک‌های فردی یا QR-کدها.

منوی کلاینت

مجموعه محدودی از دکمه‌ها برای کلاینت در دسترس است:

دکمه	عملکرد
آمار کلاینت	داده‌های مربوط به تمام اشتراک‌های متصل به Telegram User ID کلاینت را نمایش می‌دهد.
دستورات	راهنمای دستورات کلاینت را نمایش می‌دهد.
تنظیمات اشتراک	انتخاب کلاینت خودتان → لینک اشتراک.
لینک‌های فردی	انتخاب کلاینت خودتان → لینک‌های فردی.
QR-کد	انتخاب کلاینت خودتان → QR-کدها.

اگر کاربری هیچ کلاینتی با Telegram User ID او تعریف نشده باشد، ربات پاسخ می‌دهد: « پیکربندی شما یافت نشد! لطفاً از مدیر بخواهید Telegram User ID شما را در پیکربندی وارد کند. User ID شما: ...». این ID را باید به مدیر بدهید تا در فیلد کلاینت وارد کند.

14.3. دستورات ربات

چهار دستور در ربات ثبت شده‌اند که در منوی «/» تلگرام قابل مشاهده‌اند:

دستور	توضیح (از منو)	دسترسی	عملکرد
`/start`	نمایش منوی اصلی	همه	خوش‌آمدگویی؛ به مدیر علاوه بر این « خوش آمدید به ربات مدیریت !» و منوی اصلی نمایش داده می‌شود.
`/help`	راهنمای ربات	همه	پیام خوش‌آمدگویی کلی و پیشنهاد انتخاب آیتم منو را نمایش می‌دهد.
`/status`	بررسی وضعیت ربات	همه	پاسخ « ربات به درستی کار می‌کند» را می‌دهد.
`/id`	نمایش Telegram ID شما	همه	«User ID شما: `...` » را برمی‌گرداند. برای دریافت User ID خودتان مفید است.

علاوه بر دستورات ثبت‌شده، سه دستور آرگومانی نیز پردازش می‌شوند (در منوی «/» نمایش داده نمی‌شوند اما کار می‌کنند):

- **`/usage [Email]`** — جستجوی کلاینت بر اساس ایمیل.
  - برای **مدیر** کارت کامل کلاینت را نمایش می‌دهد (با دکمه‌های مدیریت).
  - برای **کلاینت** فقط اشتراک خودش با ایمیل مشخص‌شده را نمایش می‌دهد (بر اساس Telegram User ID متصل). بدون آرگومان ربات می‌خواهد ایمیل وارد شود: « لطفاً ایمیل را برای جستجو وارد کنید».
- **`/inbound [نام اتصال]`** — فقط برای مدیر. inbound را بر اساس Remark جستجو کرده و پارامترهای آن را با آمار تمام کلاینت‌ها نمایش می‌دهد. بدون آرگومان (یا برای کلاینت) — « دستور ناشناخته».
- **`/restart`** — فقط برای مدیر. Xray Core را مجدداً راه‌اندازی می‌کند. پاسخ‌های ممکن: « هسته Xray با موفقیت راه‌اندازی مجدد شد»، « Xray Core در حال اجرا نیست» (اگر هسته کار نکند)، « خطا در راه‌اندازی مجدد Xray-core. <خطا>». هر آرگومانی پس از `/restart` منجر به پیام دستور ناشناخته با راهنمای `/restart` می‌شود.

در چت‌های گروهی، دستور به شکل `/دستور@botusername` فقط در صورتی پردازش می‌شود که username با نام ربات فعلی مطابقت داشته باشد.

راهنمای مدیر (دکمه «دستورات»):

```
Xray Core: /restart برای راه‌اندازی مجدد
/usage [Email] :برای جستجوی کلاینت بر اساس ایمیل
/inbound [نام اتصال] :برای جستجوی اتصالات ورودی (با آمار کلاینت‌ها)
Telegram User ID شما: /id
```

راهنمای کلاینت:

```
/usage [Email] :برای مشاهده اطلاعات اشتراک خود
Telegram User ID شما: /id
```

14.4. مدیریت کلاینت (فقط مدیر)

با باز کردن کارت کلاینت (از طریق «تمام کلاینت‌ها»، «آنلاین»، «به زودی تمام می‌شود» یا `/usage`)، مدیر اطلاعات کلاینت (ایمیل، inbound های متصل، وضعیت «فعال»، وضعیت اتصال، تاریخ انقضا، مصرف ترافیک) و دکمه‌های inline مدیریت را می‌بیند:

دکمه	هدف
بروزرسانی	بارگیری مجدد کارت کلاینت.
بازنشانی ترافیک	بازنشانی شمارنده ترافیک کلاینت. نیاز به تأیید « تأیید بازنشانی ترافیک؟» دارد.

دکمه	هدف
محدودیت ترافیک	تعیین محدودیت ترافیک. مقادیر آماده: ♾ بی‌محدود (0)، 200/150/100/80/60/50/40/30/20/10/5/1 GB یا « سفارشی» — ورود عدد با صفحه‌کلید عددی داخلی (دکمه‌های 0–9، « » — بازنشانی به 0، « » — حذف آخرین رقم، « تأیید: N»). مقدار بر حسب گیگابایت تعیین می‌شود.
تغییر تاریخ انقضا	گزینه‌های آماده: ♾ بی‌محدود، « سفارشی»، افزودن 7/10/14/20 روز، 1/3/6/12 ماه. عدد مثبت مهلت را تمدید می‌کند (روزها را به تاریخ انقضای فعلی یا «الان» اگر مهلت منقضی شده اضافه می‌کند)؛ 0 محدودیت زمانی را حذف می‌کند.
لاگ IP	آدرس‌های IP ثبت‌شده کلاینت را (با برچسب زمانی در صورت وجود) نمایش می‌دهد. از لاگ « بروزرسانی» و « پاک کردن IP» (با تأیید « تأیید پاک کردن IP؟») در دسترس هستند.
محدودیت IP	محدودیت IP همزمان. گزینه‌ها: ♾ بی‌محدود (0)، 1–10 یا « سفارشی» (صفحه‌کلید عددی).
تنظیم کاربر تلگرام	Telegram User ID متصل فعلی کلاینت را نمایش می‌دهد؛ امکان حذف اتصال (« حذف کاربر تلگرام» با تأیید) را فراهم می‌کند. اتصال کاربر جدید از طریق انتخاب مخاطب سیستم تلگرام انجام می‌شود.
فعال/غیرفعال	کلاینت را فعال یا غیرفعال می‌کند. نیاز به تأیید « تأیید فعال/غیرفعال کردن کاربر؟» دارد.

تمام عملیاتی که پیکربندی را تغییر می‌دهند (محدودیت ترافیک/IP، تاریخ انقضا، اتصال/قطع کاربر تلگرام، فعال/غیرفعال)، در صورت لزوم Xray را برای راه‌اندازی مجدد علامت‌گذاری می‌کنند تا تغییرات اعمال شوند. پس از عملیات موفق، ربات تأییدیه‌ای به شکل « : ...» نمایش می‌دهد و کارت را مجدداً نشان می‌دهد.

هر ورودی عددی در جادوگرها به مقدار < 999999 محدود می‌شود.

14.5. اعلان‌ها و گزارش‌ها

اعلان‌ها برای تمام مدیران (تمام User IDهای موجود در `tgBotChatId`) ارسال می‌شوند.

شبکه رویدادها و انتخاب اعلان‌ها

اعلان‌ها بر پایه یک شبکه رویداد واحد ساخته شده‌اند و دو کانال تحویل وجود دارد — **تلگرام** و **ایمیل (SMTP)**. برای هر کانال به‌طور جداگانه مشخص می‌شود که چه رویدادهایی اعلان داشته باشند. در **تنظیمات** → **تلگرام** این کار در تب **Notifications** انجام می‌شود، در **تنظیمات** → **Email** — در تب همنام.

رویدادها به‌صورت کارت‌های گروهی دسته‌بندی شده‌اند؛ هر گروه یک کلید اصلی با شمارنده رویدادهای فعال (n/کل) و حالت میانی دارد که نشان می‌دهد تنها بخشی انتخاب شده است. گروه‌های موجود:

- **Outbound** — «Down» (`outbound.down`) و «Up» (`outbound.up`): خرابی و بازیابی outbound.
- **Xray Core** — «Crash» (`xray.crash`): خاتمه غیرعادی هسته Xray.
- **Nodes** — «Down» (`node.down`) و «Up» (`node.up`): گره غیرقابل دسترس شد یا بازیابی شد.
- **System** — «CPU high (%)» (`cpu.high`) و «Memory high (%)» (`memory.high`): بار بالای پردازنده و حافظه RAM. هر دو رویداد دارای فیلد inline آستانه به درصد هستند.
- **Security** — «Login attempt» (`login.attempt`): تلاش برای ورود به پنل.

مجموعه رویدادهای فعال به‌طور جداگانه ذخیره می‌شود: برای تلگرام — در `tgEnabledEvents`، برای ایمیل — در `smtpEnabledEvents`. به‌طور پیش‌فرض در هر دو کانال «Login attempt» و «CPU high» فعال هستند (مقدار `login.attempt,cpu.high`).

اعلان ورود به پنل

با تیک **اعلان ورود** (`tgBotLoginNotify` ، به‌طور پیش‌فرض فعال) کنترل می‌شود. در هر بار تلاش برای ورود به پنل وب، مدیران پیامی دریافت می‌کنند:

- در صورت موفقیت: « ✅ ورود موفق به پنل.» + هاست، نام کاربری، IP، زمان.
- در صورت شکست: « ❗ خطا در ورود به پنل.» + هاست، **دلیل** (مثلاً «خطای 2FA» در صورت اشتباه بودن عامل دوم)، نام کاربری، IP، زمان.

تجاوز از بار CPU و حافظه

هر دقیقه یک بار پنل بار پردازنده و حافظه RAM را بررسی می‌کند. اگر آستانه **`tgCpu`** > 0 باشد و میانگین بار CPU در یک دقیقه از آن بیشتر شود، برای مدیران ارسال می‌شود: « 🔴 بار پردازنده N% است که از آستانه M% بیشتر است». به‌طور مشابه بار RAM در برابر آستانه **`tgMemory`** (به‌طور پیش‌فرض 80٪) بررسی می‌شود — رویداد «Memory high (%)».

هر دو آستانه از طریق فیلدهای inline کنار رویدادهای «CPU high (%)» و «Memory high (%)» در گروه **System** در تب Notifications تعریف می‌شوند (به «شبکه رویدادها و انتخاب اعلان‌ها» مراجعه کنید). برای کانال ایمیل کلیدهای جداگانه `smtpCpu` و `smtpMemory` وجود دارند. با مقدار آستانه 0، بررسی مربوطه غیرفعال می‌شود.

گزارش دوره‌ای (طبق برنامه)

بر اساس عبارت cron از فیلد **دفعات اعلان** (`tgRunTime` ، به‌طور پیش‌فرض `@daily`) برنامه‌ریزی می‌شود. اگر مقدار خالی یا نادرست باشد، از `@daily` استفاده می‌شود. گزارش شامل:

سازنده برنامه زمانی

فیلد **دفعات اعلان برای مدیران از ربات** نه با وارد کردن رشته به‌صورت دستی، بلکه از طریق سازنده برنامه زمانی تنظیم می‌شود. ابتدا از لیست کشویی یک حالت انتخاب می‌شود:

- **`@every` — تکرار با فاصله زمانی** — فیلد عدد و انتخاب واحد (**ثانیه‌ها / دقیقه‌ها / ساعت‌ها**) ظاهر می‌شوند؛ نتیجه به عبارتی مثل `@every 6h` تبدیل می‌شود.
- **`@hourly` — هر ساعت**، **`@daily` — هر روز در 00:00**، **`@weekly` — هر هفته**، **`@monthly` — هر ماه** — پیش‌تنظیم‌های آماده که به عنوان ماکروی مربوطه ذخیره می‌شوند (`@hourly` ، `@daily` ، `@weekly` ، `@monthly`).
- **سفارشی (crontab)** — فیلد برای عبارت crontab شخصی زمان‌بند پنل با ثانیه‌های فعال کار می‌کند، بنابراین عبارت سفارشی از **6 فیلد** تشکیل شده است: ثانیه، دقیقه، ساعت، روز ماه، ماه، روز هفته (مثلاً `0 30 8 * * *` — هر روز در 08:30:00). هنگام تغییر به این حالت، فیلد با معادل crontab انتخاب فعلی پر می‌شود تا نقطه شروعی برای ویرایش وجود داشته باشد.

مثال: مقادیر فیلد «دفعات اعلان» (`tgRunTime`). هم اختصارات آماده و هم فرمت crontab کامل پشتیبانی می‌شوند:

مقدار	زمان اجرا
`@daily`	یک بار در شبانه‌روز در نیمه‌شب (مقدار پیش‌فرض)
`@hourly`	هر ساعت
`@every 6h`	هر 6 ساعت
`0 9 * * *`	هر روز در 09:00
`0 9 * * 1`	هر دوشنبه در 09:00

مقدار	زمان اجرا
`0 */12 * * *`	هر 12 ساعت (در 00:00 و 12:00)

ترتیب فیلدها در crontab: دقیقه، ساعت، روز ماه، ماه، روز هفته.

1. رشته «?» گزارش‌های برنامه‌ریزی‌شده: <برنامه>» و تاریخ/زمان فعلی.
2. **وضعیت سرور** (به زیر مراجعه کنید).
3. بلوک «به زودی تمام می‌شود» برای inbound و کلاینت‌ها.
4. اعلان‌های شخصی برای کلاینت‌های دارای Telegram User ID متصل — به هر کلاینت غیرمدیر فهرستی از اشتراک‌هایش که به زودی ترافیک/مهلت آن‌ها تمام می‌شود ارسال می‌شود (با در نظر گرفتن غیرفعال‌شده‌ها).
5. اگر **پشتیبان‌گیری از پایگاه داده** (`tgBotBackup`) فعال باشد — پشتیبان پایگاه داده برای مدیران.

وضعیت سرور شامل: هاست، نسخه 3X-UI و Xray، IPv4/IPv6، مدت زمان فعالیت (بر حسب روز)، میانگین بار (Load1/2/3)، RAM (فعلی/کل)، تعداد کلاینت‌های آنلاین، شمارنده‌های اتصالات TCP/UDP، کل ترافیک شبکه (↑/↓) و وضعیت Xray.

«به زودی تمام می‌شود» نشان می‌دهد:

- برای inbound: تعداد غیرفعال‌شده‌ها و تعداد «به زودی تمام‌شونده‌ها»، سپس فهرست این inbound (Remark، پورت، ترافیک، تاریخ انقضا)؛
- برای کلاینت‌ها: همان موارد، به علاوه کارت‌های کلاینت و دکمه‌هایی با ایمیل آن‌ها (ضربه زدن کارت کلاینت را باز می‌کند).

آستانه‌های «به زودی تمام شدن» از تنظیمات کلی پنل گرفته می‌شوند: ذخیره ترافیک (بر حسب GB) و ذخیره مهلت (بر حسب روز). inbound/کلاینتی «رو به اتمام» محسوب می‌شود که کمتر از آستانه ترافیک تا محدودیت داشته باشد یا کمتر از آستانه روز تا تاریخ انقضا.

14.6. پشتیبان‌گیری و لاگ‌ها

- **پشتیبان‌گیری از پایگاه داده** (دکمه «پشتیبان‌گیری از پایگاه داده» یا تیک در گزارش دوره‌ای): ربات زمان پشتیبان‌گیری، فایل پایگاه داده (`x-ui.db`، یا `x-ui.dump` برای PostgreSQL) و فایل پیکربندی Xray `config.json` را ارسال می‌کند.

نام فایل پشتیبانی که ربات ارسال می‌کند بر اساس آدرس سرور تشکیل می‌شود: از مقدار **webDomain** استفاده می‌شود، و اگر تنظیم نشده باشد — از IP عمومی سرور. این کمک می‌کند بفهمید فایل از کدام سرور آمده، زمانی که پشتیبان‌ها از چندین پنل جمع‌آوری می‌شوند. اگر تعیین آدرس ممکن نباشد، از نام کلی استفاده می‌شود.

- **لاگ مسدودسازی** (دکمه «لاگ مسدودسازی»): فایل‌های لاگ فعلی و قبلی آدرس‌های IP مسدودشده به دلیل تجاوز از محدودیت IP را ارسال می‌کند. فایل‌های خالی ارسال نمی‌شوند.

14.7. ویژگی‌های عملکرد

- **پیام‌های طولانی** به بخش‌هایی تقسیم می‌شوند (آستانه ~2000 کاراکتر)، صفحه‌کلید inline به آخرین بخش پیوست می‌شود.
- **موازی‌سازی**: دستورات و ضربات دکمه به‌صورت همزمان پردازش می‌شوند (تا 10 پردازشگر همزمان).
- **قابلیت اطمینان ارسال**: در صورت خطاهای اتصال، پیام‌ها با تأخیر نمایی (1s/2s/4s، تا 3 تلاش) مجدداً ارسال می‌شوند.
- **حافظه پنهان**: داده‌های «وضعیت سرور» ذخیره می‌شوند تا فشار زدن مکرر «بروزرسانی» سیستم را تحت فشار نگذارد.
- **راه‌اندازی مجدد ربات**: هنگام ذخیره تنظیماتی که ربات را تحت تأثیر قرار می‌دهند (پرچم فعال‌سازی، توکن، User IDهای مدیران یا آدرس سرور API)، پنل خودش حلقه نظرسنجی قبلی را متوقف کرده و یک حلقه جدید با پارامترهای به‌روز راه‌اندازی می‌کند — برای این کار نیازی به بارگذاری مجدد پنل نیست. تنها یک نمونه دریافت به‌روزرسانی در یک زمان کار می‌کند.

15. پایگاه‌های جغرافیایی (geoip / geosite و سفارشی)

پایگاه‌های جغرافیایی فایل‌های دودویی `.dat` هستند که Xray-core برای مسیریابی و فیلتر کردن ترافیک بر اساس تعلق به یک کشور (محدوده‌های IP) یا بر اساس دسته‌بندی دامنه‌ها از آن‌ها استفاده می‌کند. پنل می‌تواند هم مجموعهٔ استاندارد فایل‌های جغرافیایی و هم منابع دلخواه کاربر را که با URL مشخص شده‌اند بارگیری و به‌روزرسانی کند. همهٔ فایل‌ها در پوشهٔ `bin` در کنار باینری Xray ذخیره می‌شوند (مسیر پیش‌فرض `bin`، با متغیر محیطی `XUI_BIN_FOLDER` قابل بازنویسی است).

15.1. geoip.dat و geosite.dat چه هستند

- **geoip.dat** — پایگاه تطبیق «نشانی IP ← کد کشور/منطقه». در قوانین مسیریابی به‌شکل `geoip:<код>` استفاده می‌شود، برای مثال `geoip:ru`، `geoip:cn`، و همچنین برای برچسب‌های ویژه `geoip:private` (شبکه‌های خصوصی/محلی). از نظر معنایی، این پاسخ به پرسش «این IP در کدام کشور قرار دارد» است.
- **geosite.dat** — پایگاه تطبیق «دامنه ← دسته/فهرست». به‌شکل `geosite:<категория>` استفاده می‌شود، برای مثال `geosite:category-ads-all` (دامنه‌های تبلیغاتی)، `geosite:google`، `geosite:ru`. از نظر معنایی، این فهرست‌های گروه‌بندی‌شدهٔ دامنه‌ها هستند.

این فایل‌ها برای ساختن قوانینی از نوع «تمام ترافیک به IPها/دامنه‌های روسی — مستقیم، بقیه — از طریق outbound» و مانند آن لازم‌اند. خود قوانین در بخش مسیریابی Xray تعریف می‌شوند؛ پایگاه‌های جغرافیایی تنها داده‌ها را برای آن‌ها تأمین می‌کنند. بدون فایل‌های جغرافیایی به‌روز، قوانینی که به `geosite:` / `geoip:` ارجاع می‌دهند کار نخواهند کرد یا به فهرست‌های قدیمی متکی خواهند بود.

مثال: قانون «دامنه‌ها و IPهای روسی — مستقیم». چنین قانونی در بخش مسیریابی تمام ترافیک به منابع روسی را به outbound با تگ `direct` هدایت می‌کند:

```
{
  "type": "field",
  "domain": ["geosite:category-ru"],
  "ip": ["geoip:ru"],
  "outboundTag": "direct"
}
```

15.2. فایل‌های جغرافیایی استاندارد و به‌روزرسانی آن‌ها

پنل شامل یک «فهرست مجاز» (allowlist) ثابت از شش فایل استاندارد با منابع بارگیری به‌صورت سفت‌وسخت تعریف‌شده است. به‌روزرسانی از طریق `POST /panel/api/server/updateGeofile/:fileName` انجام می‌شود (یا بدون نام فایل — برای به‌روزرسانی همه به‌صورت یکجا).

مثال: به‌روزرسانی یک فایل و همهٔ فایل‌ها به‌صورت یکجا از طریق API. به‌روزرسانی تنها `geoip_RU.dat`:

```
curl -X POST 'https://panel.example.com:2053/panel/api/server/updateGeofile/
geoip_RU.dat' \
  -H 'Cookie: 3x-ui=<session-cookie>'
```

به‌روزرسانی هر شش فایل استاندارد با یک درخواست (نام فایل مشخص نشده است):

```
curl -X POST 'https://panel.example.com:2053/panel/api/server/updateGeofile' \
  -H 'Cookie: 3x-ui=<session-cookie>'
```

پاسخ موفق:

```
{ "success": true, "msg": "Geofile updated successfully", "obj": null }
```

نام فایل	منبع (مخزن انتشارها)
`geoip.dat`	`github.com/Loyalsoldier/v2ray-rules-dat` (geoip.dat)
`geosite.dat`	`github.com/Loyalsoldier/v2ray-rules-dat` (geosite.dat)
`geoip_IR.dat`	`github.com/chocolate4u/Iran-v2ray-rules` (geoip.dat)
`geosite_IR.dat`	`github.com/chocolate4u/Iran-v2ray-rules` (geosite.dat)
`geoip_RU.dat`	`github.com/runetfreedom/russia-v2ray-rules-dat` (geoip.dat)
`geosite_RU.dat`	`github.com/runetfreedom/russia-v2ray-rules-dat` (geosite.dat)

ویژگی‌های به‌روزرسانی فایل‌های استاندارد:

- **دکمهٔ به‌روزرسانی یک فایل.** پیش از بارگیری یک تأیید نمایش داده می‌شود: «آیا واقعاً می‌خواهید فایل جغرافیایی را به‌روزرسانی کنید؟» با توضیح «این فایل #filename# را به‌روزرسانی می‌کند.» (انگلیسی: *Do you really want to update the geofile? This will update the #filename# file.*). در صورت موفقیت اعلانی ظاهر می‌شود: «فایل‌های جغرافیایی با موفقیت به‌روزرسانی شدند» (انگلیسی: *Geofile updated successfully*).
- **دکمهٔ «Обновить все»** (انگلیسی: *Update all*) هر شش فایل را بارگیری می‌کند. تأیید: «این تمام فایل‌های جغرافیایی را به‌روزرسانی می‌کند.» (انگلیسی: *This will update all geofiles*.).
- **بارگیری مشروط.** اگر فایل محلی از پیش موجود باشد، در درخواست سرآیند `If-Modified-Since` با زمان تغییر فایل قرار داده می‌شود. پاسخ سرور `304 Not Modified` به این معناست که فایل تغییری نکرده است — دوباره دانلود نمی‌شود و تنها برچسب زمانی فایل به‌روزرسانی می‌شود.
- **ایمنی نام فایل.** تنها نام‌های موجود در allowlist پذیرفته می‌شوند؛ نام از نظر نبود `..`، جداکننده‌های مسیر `/` و `\`، مسیرهای مطلق بررسی می‌شود و باید با الگوی `^[a-zA-Z0-9._-]+\.dat$` مطابقت داشته باشد. هر نامی خارج از فهرست با خطای «Invalid geofile name» رد می‌شود.
- **راه‌اندازی مجدد Xray.** پس از بارگیری فایل‌های جغرافیایی، Xray-core دوباره راه‌اندازی می‌شود تا پایگاه‌های به‌روزشده را دوباره بخواند. اگر راه‌اندازی مجدد ناموفق باشد، خط مربوطه به پیام خطا افزوده می‌شود.

به‌روزرسانی پایگاه‌های جغرافیایی از خط فرمان (x-ui)

پایگاه‌های جغرافیایی را می‌توان بدون پنل نیز به‌روزرسانی کرد — از طریق منوی تعاملی `x-ui` (گزینهٔ به‌روزرسانی فایل‌های جغرافیایی) یا با فرمان غیرتعاملی `x-ui update-all-geofiles`. برای هر فایل از مجموعه (geoip/geosite، شامل مجموعه‌های IR و RU) وضعیت جداگانه نمایش داده می‌شود: «به‌روزرسانی شد»، «از پیش به‌روز است» یا «خطای بارگیری». در صورت بارگیری ناموفق، پیام موفقیت کاذب چاپ نمی‌شود. راه‌اندازی مجدد Xray (و در نتیجه قطع اتصال‌های فعال) تنها در صورتی رخ می‌دهد که دست‌کم یک فایل واقعاً به‌روزرسانی شده باشد؛ اگر هیچ فایلی تغییر نکرده باشد (همه `304 Not Modified` برگردانده باشند)، پنل و Xray دوباره راه‌اندازی نمی‌شوند.

15.3. به‌روزرسانی خودکار داده‌های جغرافیایی با ابزارهای Xray (Geodata Auto-Update)

منابع `.dat` اضافی با URL دلخواه نه با ابزارهای پنل، بلکه از طریق بخش بومی `geodata` در Xray-core افزوده می‌شوند. بخش مربوطه در پنجرهٔ مودال به‌روزرسانی‌های Xray قرار گرفته است (داشبورد ← به‌روزرسانی‌های Xray، `xrayUpdates`) — این برگهٔ «به‌روزرسانی خودکار Geodata» (انگلیسی: *Geodata Auto-Update*) است. پنل در این‌جا تنها کلید `geodata` را در قالب پیکربندی Xray ویرایش می‌کند؛ بارگیری، بررسی و بارگذاری مجدد داغ فایل‌ها را خودِ هستهٔ Xray انجام می‌دهد.

در بالای این بخش راهنمایی نمایش داده می‌شود: «Xray این فایل‌ها را طبق زمان‌بندی بارگیری می‌کند و بدون راه‌اندازی مجدد آن‌ها را داغ بارگذاری می‌کند. URLها باید HTTPS باشند. فایل باید پیش از آنکه Xray بتواند آن را به‌روزرسانی کند، از قبل در پوشهٔ bin وجود داشته باشد.» (انگلیسی: *Xray downloads these files on schedule and hot-reloads them without a restart. URLs must be HTTPS. Each file must already exist in the bin folder once before Xray can update it*.).

فیلدهای بخش

- **زمان‌بندی (cron)** (انگلیسی: *Schedule (cron)*) — رشتهٔ cron از 5 فیلد؛ مقدار پیش‌فرض `0 4 * * *` (هر روز ساعت 04:00). هنگام ذخیره بررسی می‌شود که رشته دقیقاً 5 فیلد داشته باشد، در غیر این صورت خطای «Cron باید 5 فیلد داشته باشد، مثلاً 0 4 * * *» نمایش داده می‌شود.
- **بارگیری از طریق outbound (اختیاری)** (انگلیسی: *Download through outbound (optional)*) — فهرست کشویی با تگ‌های outboundهای موجود (به‌اضافهٔ outboundهای اشتراک‌ها) که Xray فایل‌ها را از طریق آن بارگیری خواهد کرد؛ outboundهایی با پروتکل `blackhole` در فهرست قرار نمی‌گیرند. این فیلد را می‌توان خالی گذاشت — در این صورت از اتصال مستقیم استفاده می‌شود. این انتخاب مستقل از outbound مربوط به درخواست‌های خود پنل است (به §11 نگاه کنید): به‌روزرسانی خودکار geodata outbound جداگانهٔ خود را برای بارگیری دارد.
- **فهرست فایل‌ها** — هر خط یک جفت «URL + نام فایل» (انگلیسی: *File name*) را تعریف می‌کند. URL باید با `https://` آغاز شود (در غیر این صورت «هر فایل به یک HTTPS URL نیاز دارد.»). نام فایل به‌صورت ساده، بدون مسیر و جداکننده مشخص می‌شود — تنها نویسه‌های `^[A-Za-z0-9._-]+$` (در غیر این صورت «نام فایل باید ساده باشد، برای مثال geosite_custom.dat (بدون مسیر).»). هنگام وارد کردن URL، پنل تلاش می‌کند نام فایل را به‌صورت خودکار از آخرین بخش مسیر قرار دهد. دکمهٔ «افزودن فایل» (انگلیسی: *Add file*) یک خط می‌افزاید و دکمهٔ سطل زباله آن را حذف می‌کند.

اگر فهرست خالی باشد، راهنمایی نمایش داده می‌شود: «هیچ فایلی پیکربندی نشده است. در قوانین مسیریابی فایل‌ها به‌صورت ext:geosite_custom.dat:category مشخص می‌شوند.» (انگلیسی: *No files configured. Reference files in routing rules as ext:geosite_custom.dat:category*.).

ذخیره‌سازی

دکمهٔ «ذخیره و راه‌اندازی مجدد Xray» (انگلیسی: *Save & Restart Xray*) تأیید «تنظیمات geodata ذخیره شود؟» را با توضیح «قالب پیکربندی Xray به‌روزرسانی و Xray دوباره راه‌اندازی خواهد شد.» نمایش می‌دهد (انگلیسی: *Save geodata settings? This updates the Xray config template and restarts Xray*.). پس از ذخیره، کلید `geodata` در قالب پیکربندی نوشته می‌شود (`POST /panel/api/xray/update`) و Xray دوباره راه‌اندازی می‌شود (`POST /panel/api/server/restartXrayService`). اگر فهرست فایل‌ها خالی باشد، کلید `geodata` از قالب حذف می‌شود.

ویژگی‌های مهم:

- **فایل باید از قبل در `bin` وجود داشته باشد.** Xray تنها آن دسته از فایل‌های `.dat` را به‌روزرسانی می‌کند که در زمان اجرا از قبل در پوشهٔ `bin` موجود باشند. بنابراین فایل سفارشی جدید نخست به‌صورت دستی در `bin` قرار داده می‌شود (یا دست‌کم یک نسخهٔ خالی/قدیمی با نام موردنظر در آن‌جا ساخته می‌شود)، و تنها پس از آن Xray شروع به نگه‌داشتن آن در حالت به‌روز طبق زمان‌بندی می‌کند.
- **بارگذاری مجدد داغ.** پس از بارگیری زمان‌بندی‌شده، Xray پایگاه‌های به‌روزشده را بدون راه‌اندازی مجدد کامل فرایند دوباره می‌خواند.
- **سازگاری.** فایل‌های جغرافیایی پیش‌تر بارگیری‌شده (چه استاندارد و چه سفارشی) در قوانین مسیریابی با نحو `ext:` بدون تغییر همچنان کار می‌کنند.

اگر فهرست خالی باشد، راهنمایی نمایش داده می‌شود: «هنوز هیچ منبع جغرافیایی سفارشی وجود ندارد — برای ساختن یکی روی «افزودن» کلیک کنید» (انگلیسی: *No custom geo sources yet — click Add to create one*).

ستون‌های جدول و فیلدهای منبع

فیلد (UI)	JSON	مقدار پیش‌فرض	توضیح
نوع (*Type*)	`type`	– (الزامی)	نوع منبع: تنها `geosite` یا `geoip`. نام فایل نهایی را تعیین می‌کند.
نام مستعار (*Alias*)	`alias`	– (الزامی)	شناسهٔ کوتاه منبع. نام فایل از آن و نوع ساخته می‌شود.
URL (*URL*)	`url`	– (الزامی)	پیوند مستقیم به فایل `.dat` (http/https).
فعال (*Enabled*)	–	–	نشانهٔ فعال‌بودن منبع در فهرست.
به‌روزرسانی‌شده (*Last updated*)	`lastUpdatedAt`	`0`	زمان آخرین به‌روزرسانی موفق (زمان یونیکس؛ `0` — هنوز به‌روزرسانی نشده).
مسیریابی (*Routing*) (...:ext) (*...:ext*)	–	–	رشتهٔ آمادهٔ قوانین مسیریابی: `ext:<файл.dat>:tag`.
اقدامات (*Actions*)	–	–	دکمه‌های «ویرایش»، «حذف»، «به‌روزرسانی اکنون».

افزون بر این، در پایگاه‌داده فیلدهای خدماتی نگه‌داری می‌شوند: `localPath` (مسیر واقعی فایل در پوشهٔ `bin`)، `lastModified` (مقدار سرآیند `Last-Modified` از سرور، که برای بارگیری مشروط استفاده می‌شود)، `createdAt` و `updatedAt`.

نام‌گذاری فایل‌ها

نام فایل نهایی به‌صورت خودکار از نوع و نام مستعار ساخته می‌شود:

- نوع `geoip` ← `geoip_<alias>.dat`؛
- نوع `geosite` ← `geosite_<alias>.dat`.

برای مثال، منبعی با نوع `geosite` و نام مستعار `myads` فایل `geosite_myads.dat` را می‌سازد.

مثال: افزودن منبع از طریق API. افزودن فهرست دامنه‌های تبلیغاتی خودتان به‌عنوان منبع `geosite` با نام مستعار `myads`:

```
curl -X POST 'https://panel.example.com:2053/panel/api/server/customGeo/add' \
  -H 'Cookie: 3x-ui=<session-cookie>' \
  -H 'Content-Type: application/json' \
  -d '{
    "type": "geosite",
    "alias": "myads",
    "url": "https://example.com/lists/myads.dat"
  }'
```

پنل فایل را در پوشهٔ `bin` با نام `geosite_myads.dat` بارگیری می‌کند، رکورد را ذخیره می‌کند و Xray را دوباره راه‌اندازی می‌کند.

دکمه‌ها و اقدامات

- **افزودن** (انگلیسی: *Add*) — فرم «افزودن منبع» (انگلیسی: *Add custom geo*) را باز می‌کند. دکمهٔ ذخیره — «ذخیره» (انگلیسی: *Save*). API: `POST /add`.

- **ویرایش** (انگلیسی: *Edit*) — فرم «ویرایش منبع» (انگلیسی: *Edit custom geo*). API: `POST /update/:id`. هنگام تغییر نوع یا نام مستعار، فایل قدیمی حذف و فایل جدید دوباره بارگیری می‌شود.
- **حذف** (انگلیسی: *Delete*) — تأیید «این منبع جغرافیایی سفارشی حذف شود؟» (انگلیسی: *Delete this custom geo source?*). رکورد را از پایگاه‌داده و خودِ فایل `.dat` را حذف می‌کند. API: `POST /delete/:id`. در صورت موفقیت: «فایل جغرافیایی سفارشی «<имя>» حذف شد».
- **به‌روزرسانی اکنون** (انگلیسی: *Update now*) — منبع مشخصی را دوباره بارگیری و برچسب زمانی را به‌روزرسانی می‌کند. API: `POST /download/:id`. در صورت موفقیت: «Geofile «<имя>» به‌روزرسانی شد».
- **به‌روزرسانی همه** — تمام منابع سفارشی کاربر را به‌صورت یکجا به‌روزرسانی می‌کند. API: `POST /update-all`. در صورت موفقیت کامل: «همهٔ منابع جغرافیایی سفارشی به‌روزرسانی شدند» (انگلیسی: *All custom geo sources updated*). اگر دست‌کم یک منبع به‌روزرسانی نشود، عملیات نسبتاً ناموفق در نظر گرفته می‌شود با پیام «به‌روزرسانی یک یا چند منبع جغرافیایی سفارشی ناموفق بود» (انگلیسی: *One or more custom geo sources failed to update*)، و در پاسخ منابع موفق و ناموفق فهرست می‌شوند.

پس از هر یک از این اقدامات (افزودن، ویرایش، حذف، به‌روزرسانی، به‌روزرسانی همه در صورت وجود موفقیت‌ها) Xray-core دوباره راه‌اندازی می‌شود.

گام‌به‌گام: افزودن منبع

1. روی «افزودن» کلیک کنید.
2. در فیلد «نوع» گزینهٔ `geosite` یا `geoip` را انتخاب کنید.
3. در فیلد «نام مستعار» شناسه‌ای وارد کنید (تنها حروف کوچک لاتین، ارقام، `-` و `_`؛ راهنمای جای‌نما: `a-z 0-9 _ -`).
4. در فیلد «URL» پیوند مستقیم به فایل `.dat` را مشخص کنید (باید با `http://` یا `https://` آغاز شود).
5. روی «ذخیره» کلیک کنید. پنل بی‌درنگ فایل را در پوشهٔ `bin` بارگیری می‌کند، رکورد را ذخیره می‌کند و Xray را دوباره راه‌اندازی می‌کند.

15.4. اعتبارسنجی و محدودیت‌ها

هنگام ساختن و ویرایش منبع، بررسی‌های سخت‌گیرانه انجام می‌شود. پیام‌های خطا:

شرط	پیام (RU)	پیام (EN)
نوع `geosite` / `geoip` نیست	Тип должен быть geosite или geoip	*Type must be geosite or geoip*
نام مستعار خالی	Укажите псевдоним	*Alias is required*
نویسه‌های نامجاز در نام مستعار (نه `^[a-z0-9_-]+$`)	Псевдоним содержит недопустимые символы	*Alias must match allowed characters*
نام مستعار رزرو شده است	Этот псевдоним зарезервирован	*This alias is reserved*
URL خالی	Укажите URL	*URL is required*
URL تجزیه نمی‌شود	Некорректный URL	*URL is invalid*
شِما http/https نیست	URL должен использовать http или https	*URL must use http or https*
میزبان خالی/نادرست، یا توسط محافظت SSRF مسدود شده	Некорректный хост URL	*URL host is invalid*
تکراری بودن «نوع + نام مستعار»	Такой псевдоним уже используется для этого типа	*This alias is already used for this type*
منبع یافت نشد	Источник не найден	*Custom geo source not found*

شرط	پیام (RU)	پیام (EN)
خطای بارگیری	Ошибка загрузки	*Download failed*

راهنمایی‌ها در فرم (اعتبارسنجی سمت کلاینت): «نام مستعار: تنها a-z، ارقام، - و _» (*Alias may only contain lowercase letters, digits, - and _*) و «URL باید با http:// یا https:// آغاز شود» (*URL must start with http:// or https://*).

محدودیت‌های فنی اضافی:

- **نام‌های مستعار رزرو شده.** نمی‌توان از نام‌های مستعاری که با فایل‌های استاندارد در تعارض‌اند استفاده کرد. رزرو شده‌اند (مقایسه بدون حساسیت به بزرگی و کوچکی حروف، خط تیره معادل زیرخط در نظر گرفته می‌شود): `geoip`، `geosite`، `geoip_ir`، `geosite_ir`، `geoip_ru`، `geosite_ru`. برای مثال، `geosite-ru` به‌عنوان `geosite_ru` رد خواهد شد.
- **محافظت SSRF.** میزبان URL به IP تبدیل می‌شود، و اگر به نشانی خصوصی/داخلی اشاره کند، بارگیری مسدود می‌شود (کاربر «میزبان URL نادرست است» را می‌بیند). این مانع از آن می‌شود که از پنل برای دسترسی به سرویس‌های داخلی استفاده شود.
- **محافظت در برابر path traversal.** مسیر نهایی فایل باید درون پوشهٔ `bin` قرار داشته باشد (با درنظر گرفتن symlinkها)؛ تلاش برای خروج از مرزهای آن رد می‌شود.
- **حداقل اندازهٔ فایل.** فایل بارگیری‌شده تنها در صورتی معتبر در نظر گرفته می‌شود که کمتر از 64 بایت نباشد؛ فایل بیش از حد کوچک با خطای بارگیری رد می‌شود.
- **پروکسی و بارگیری مشروط.** اگر در تنظیمات پنل پروکسی تعریف شده باشد، بارگیری از طریق آن انجام می‌شود؛ در سایر موارد از اتصال مستقیم با ترابری ایمن در برابر SSRF استفاده می‌شود. مانند فایل‌های استاندارد، `If-Modified-Since` / `304 Not Modified` اعمال می‌شود (فایل تغییرنکردهٔ تکراری دوباره بارگیری نمی‌شود). مهلت بارگیری — 10 دقیقه، آزمون در دسترس بودن URL (HEAD)، در صورت نیاز — GET جزئی) — 12 ثانیه.

15.5. بررسی خودکار هنگام راه‌اندازی پنل

هنگام راه‌اندازی، پنل تمام منابع سفارشی کاربر را پیمایش می‌کند و برای هر یک وجود و سلامت فایل محلی را بررسی می‌کند (فایل وجود ندارد، یک پوشه است یا کمتر از 64 بایت است). اگر فایل وجود نداشته باشد یا خراب باشد، آزمون منبع و تلاش برای بارگیری مجدد انجام می‌شود. این تضمین می‌کند که پس از نصب مجدد یا از دست رفتن پوشهٔ `bin`، فایل‌های جغرافیایی سفارشی کاربر به‌صورت خودکار بازیابی شوند.

15.6. استفاده از پایگاه‌های جغرافیایی در قوانین مسیریابی

در قوانین مسیریابی Xray، پایگاه‌های جغرافیایی در فیلدهایی مانند `domain` / `ip` از طریق پیشوندها استفاده می‌شوند:

- **geoip:** برای پایگاه‌های IP — `geoip:<код>`. نمونه‌ها: `geoip:ru`، `geoip:cn`، `geoip:private`. از `geoip.dat` (یا `geoip_RU.dat` و مانند آن، اگر قانون به فایل مشخصی اشاره کند) گرفته می‌شود.
- **geosite:** برای پایگاه‌های دامنه‌ای — `geosite:<категория>`. نمونه‌ها: `geosite:category-ads-all`، `geosite:google`، `geosite:ru`. از `geosite.dat` گرفته می‌شود.

مثال: مسدودسازی تبلیغات از طریق geosite. قانونی که تمام دامنه‌های تبلیغاتی را به «سیاه‌چاله» می‌فرستد (فرض بر این است که outbound با تگ `blocked` و پروتکل `blackhole` وجود دارد):

```
{
  "type": "field",
  "domain": ["geosite:category-ads-all"],
  "outboundTag": "blocked"
}
```

برای فایل‌های **سفارشی** از نحو فایل خارجی `ext:` استفاده می‌شود. راهنمایی در UI: «در قوانین مسیریابی، مقدار را به‌صورت ext:файл.dat:тег استفاده کنید (تگ را جایگزین کنید).» (انگلیسی: *In routing rules use the value column as ext:file.dat:tag (replace tag).*). قالب:

```
ext:<имя_файла.dat>:<тег>
```

که در آن `<имя_файла.dat>` همان `geoip_<alias>.dat` یا `geosite_<alias>.dat` است، و `<тег>` فهرست/دستهٔ مشخصی درون فایل. پنل در ستون «مسیریابی (ext:...)» یک الگوی آماده از نوع `ext:geosite_myads.dat:tag` پیشنهاد می‌دهد — تنها کافی است `tag` را با تگ موردنظر جایگزین کنید. نام چنین فایلی در بخش «به‌روزرسانی خودکار Geodata» (به §15.3 نگاه کنید) در فیلد «نام فایل» تعیین می‌شود — برای مثال `geosite_custom.dat`؛ در قوانین به‌صورت `ext:geosite_custom.dat:category` به آن ارجاع داده می‌شود.

مثال: قانون مبتنی بر فایل سفارشی. اگر منبعی با نوع `geosite` و نام مستعار `myads` افزوده شده باشد و درون فایل `.dat` فهرستی با تگ `ads` نشانه‌گذاری شده باشد، قانون مسیریابی به این شکل است:

```
{
  "type": "field",
  "domain": ["ext:geosite_myads.dat:ads"],
  "outboundTag": "blocked"
}
```

برای منبع IP (نوع `geoip`، نام مستعار `mycorp`، تگ `office`)، فیلد به‌صورت `"ip":` `["ext:geoip_mycorp.dat:office"]` خواهد بود.

16. بهره‌برداری: پشتیبان‌گیری، لاگ‌ها، به‌روزرسانی، CLI

این بخش نگهداری روزانه پنل را پوشش می‌دهد: ایجاد و بازیابی نسخه‌های پشتیبان پایگاه داده، مشاهده لاگ‌های پنل و Xray، راه‌اندازی مجدد و توقف سرویس‌ها، به‌روزرسانی Xray و خود پنل، وظایف دوره‌ای (cron) و حذف پنل. بخشی از عملیات از رابط وب انجام می‌شود (برگه‌های صفحه «داشبورد» و «تنظیمات پنل»)، و بخشی از منوی کنسول `x-ui` روی سرور.

16.1. پشتیبان‌گیری و بازیابی پایگاه داده

تمام داده‌های پنل (inboundها، کلاینت‌ها، گروه‌ها، نودها، تنظیمات) در یک پایگاه داده ذخیره می‌شوند. مدیریت پشتیبان‌گیری در صفحه **«داشبورد»** در برگه **«نسخه پشتیبان»** در دسترس است؛ عنوان بلوک **«پشتیبان‌گیری و بازیابی»** است.

پنل از دو موتور پایگاه داده پشتیبانی می‌کند و رفتار پشتیبان‌گیری به آن‌ها بستگی دارد:

- **SQLite** (پیش‌فرض) — داده‌ها در فایل `x-ui.db` ذخیره می‌شوند.
- **PostgreSQL** — اگر پنل روی PostgreSQL پیکربندی شده باشد، راهنمای زیر در بلوک نمایش داده می‌شود:

> «این پنل روی PostgreSQL اجرا می‌شود. «نسخه پشتیبان» یک آرشیو pg_dump (با پسوند .dump) را دانلود می‌کند، و «بازیابی» آن را از طریق pg_restore بارگذاری می‌کند. ابزارهای کلاینت PostgreSQL (pg_dump و pg_restore) باید روی سرور نصب شده باشند.»

خروجی (ایجاد نسخه پشتیبان)

دکمه **«خروجی پایگاه داده»** (به انگلیسی `Back Up`) فایل نسخه پشتیبان را روی دستگاه شما دانلود می‌کند.

موتور پایگاه داده	نام فایل	آنچه روی سرور اتفاق می‌افتد
SQLite	`{host}_YYYY-MM-DD_HHMMSS.db`	ابتدا checkpoint WAL اجرا می‌شود تا فایل حاوی آخرین رکوردها باشد، سپس فایل به طور کامل خوانده شده و برای دانلود ارسال می‌شود
PostgreSQL	`{host}_YYYY-MM-DD_HHMMSS.dump`	`pg_dump` اجرا می‌شود و آرشیو برای دانلود ارسال می‌گردد

از نسخه 3.4.2 نام فایل پشتیبان دانلودشده شامل آدرسی است که با آن پنل را باز کرده‌اید و **تاریخ-زمان**: `{host}_YYYY-MM-DD_HHMMSS` با پسوند `.db` (SQLite) یا `.dump` (PostgreSQL) — مثلاً `panel.example.com_2026-06-27_000000.db`. به این ترتیب فایل‌ها بر اساس سرور گروه‌بندی و بر اساس زمان مرتب می‌شوند، و چند پشتیبان در یک روز یکدیگر را بازنویسی نمی‌کنند. همین نام در پشتیبان‌هایی که ربات تلگرام می‌فرستد نیز استفاده می‌شود (`host` از دامنه/IP پنل گرفته می‌شود، گزینه جایگزین — `x-ui`).

راهنماهای رابط کاربری:

- SQLite: «برای دانلود فایل .db حاوی نسخه پشتیبان پایگاه داده فعلی روی دستگاهتان کلیک کنید.»
- PostgreSQL: «برای دانلود دامپ PostgreSQL (با پسوند .dump) از پایگاه داده فعلی روی دستگاهتان کلیک کنید.»

از نظر فنی، خروجی یک درخواست `GET /panel/api/server/getDb` است. نام پیوست توسط سرور (`Content-Disposition`) بسته به موتور تشکیل می‌شود.

نام فایل نسخه پشتیبان بر اساس آدرس سرور شکل می‌گیرد، نه به صورت ثابت `x-ui.db` / `x-ui.dump`. هنگام دانلود از مرورگر، نام از آدرس پنل در نوار آدرس (هاست درخواست) گرفته می‌شود؛ در غیر این صورت از دامنه وب پیکربندی شده، و در صورت نبود آن از IP عمومی

سرور (ابتدا IPv4، سپس IPv6) با بازگشت به `x-ui`. این کار تشخیص پشتیبان‌های سرورهای مختلف را آسان می‌کند. پسوند برای SQLite همان `.db` و برای PostgreSQL همان `.dump` باقی می‌ماند؛ پشتیبان‌های ارسال‌شده از طریق Telegram نیز بر اساس همان دامنه/IP نام‌گذاری می‌شوند.

مثال: دانلود نسخه پشتیبان از طریق API. همین خروجی را می‌توان با درخواست از کنسول نیز دریافت کرد — برای مثال برای یک اسکریپت پشتیبان‌گیری خودکار. نیاز به یک نشست معتبر (کوکی ورود) است:

```
# 1) ورود و ذخیره کوکی نشست
curl -s -c cookies.txt \
     -d 'username=admin&password=admin' \
     https://panel.example.com:2053/panel/login

# 2) دانلود فایل پایگاه داده (x-ui.dump یا x-ui.db: نام را سرور تعیین می‌کند)
curl -s -b cookies.txt -OJ \
     https://panel.example.com:2053/panel/api/server/getDb
```

اگر پنل از طریق مسیر پایه (Web Base Path) باز می‌شود، باید آن را به URL اضافه کنید: `…:2053/<base_path>/panel/api/server/getDb`.

ورودی (بازیابی)

دکمه **«ورودی پایگاه داده»** (به انگلیسی `Restore`) انتخاب فایل را باز کرده و آن را برای بازیابی به سرور آپلود می‌کند (`POST /panel/api/server/importDB`، فیلد فرم `db`).

راهنماهای رابط کاربری:

- SQLite: «برای انتخاب و بارگذاری فایل .db از دستگاهتان جهت بازیابی پایگاه داده از نسخه پشتیبان کلیک کنید.»
- PostgreSQL: «برای انتخاب و بارگذاری فایل .dump جهت بازیابی پایگاه داده PostgreSQL کلیک کنید. این عمل تمام داده‌های فعلی را جایگزین می‌کند.»

فرآیند ورودی برای SQLite (مهم است بدانید که اتمی و با قابلیت بازگشت است):

1. فایل آپلودشده از نظر قالب بررسی می‌شود — باید یک پایگاه داده SQLite معتبر باشد؛ در غیر این صورت خطای «Invalid db file format» برگردانده می‌شود.
2. فایل در `x-ui.db.temp` موقت ذخیره شده و از نظر یکپارچگی بررسی می‌شود.
3. **Xray قبل از جایگزینی پایگاه داده متوقف می‌شود.**
4. پایگاه داده فعلی به عنوان `x-ui.db.backup` (نسخه احتیاطی) تغییر نام می‌دهد.
5. فایل موقت به جای پایگاه داده کاری منتقل می‌شود، مقداردهی اولیه و مهاجرت‌های طرح، سپس مهاجرت inbound انجام می‌شود.
6. **اگر هر مرحله‌ای با خطا مواجه شود** — بازگشت انجام می‌شود: پایگاه داده قبلی از `x-ui.db.backup` بازیابی می‌شود و Xray روی داده‌های قدیمی راه‌اندازی مجدد می‌شود.
7. در صورت موفقیت، فایل نسخه احتیاطی حذف شده و **Xray به طور خودکار با داده‌های بازیابی‌شده راه‌اندازی مجدد می‌شود.**

پیام‌های رابط کاربری بر اساس نتیجه:

نتیجه	متن
موفقیت	«پایگاه داده با موفقیت وارد شد»
خطای ورودی	«هنگام وارد کردن پایگاه داده خطایی رخ داد»
خطای خواندن فایل	«هنگام خواندن پایگاه داده خطایی رخ داد»

> بازیابی داده‌های فعلی را به طور کامل جایگزین می‌کند. از آنجایی که Xray در این فرآیند به طور موقت متوقف می‌شود، اتصالات موجود کلاینت‌ها در طول ورودی قطع می‌شوند.

فایل مهاجرت بین موتورها (SQLite ⇄ PostgreSQL)

جدا از پشتیبان‌گیری معمول، عملکرد **«دانلود فایل مهاجرت»** (`Download Migration` ، درخواست `GET /panel/api/server/getMigration`) وجود دارد. این عملکرد یک فایل قابل حمل برای انتقال به موتور پایگاه داده دیگر ایجاد می‌کند:

موتور فعلی	آنچه دانلود می‌شود	نام فایل	هدف
SQLite	دامپ SQL قابل حمل (متن)	`x-ui.dump`	بارگذاری داده‌های شما در PostgreSQL
PostgreSQL	پایگاه داده SQLite ساخته‌شده از داده‌های PostgreSQL	`x-ui.db`	انتقال پنل به SQLite

راهنماها:

- روی SQLite: «برای دانلود خروجی قابل حمل .dump (متن SQL) از پایگاه داده SQLite خود کلیک کنید.»
- روی PostgreSQL: «برای دانلود پایگاه داده SQLite (با پسوند .db) ساخته‌شده از داده‌های PostgreSQL شما که آماده اجرای پنل روی SQLite است کلیک کنید.»

تبدیل `.db ⇄ .dump` برای SQLite را می‌توان از CLI نیز با دستور `x-ui migrateDB [file]` انجام داد (بخش 16.7 را ببینید).

پشتیبان‌گیری از طریق ربات Telegram

اگر ربات Telegram پیکربندی شده باشد (بخش مربوط به اعلان‌ها را ببینید)، می‌تواند نسخه پشتیبان را مستقیماً در چت مدیر ارسال کند. پشتیبان‌گیری از طریق Telegram شامل **دو فایل** است: خود پایگاه داده (`x-ui.db` ، یا `x-ui.dump` در PostgreSQL) و پیکربندی Xray `config.json` . پیشاپیش پیام رشته «❓ زمان پشتیبان‌گیری: ...» می‌آید.

دو روش برای دریافت پشتیبان در Telegram وجود دارد:

1. **بر اساس درخواست.** دکمه **« 📂 پشتیبان‌گیری از پایگاه داده»** در منوی ربات — ربات بلافاصله فایل‌ها را در چت فعلی ارسال می‌کند.
2. **به طور خودکار با گزارش.** در تنظیمات ربات گزینه‌ای با عنوان **«پشتیبان‌گیری از پایگاه داده»** (`Database Backup`) با توضیح «ارسال اعلان با فایل نسخه پشتیبان پایگاه داده» وجود دارد. هنگامی که فعال است، در هر ارسال دوره‌ای گزارش، ربات پس از گزارش، نسخه پشتیبان را برای همه مدیران ارسال می‌کند. دوره ارسال گزارش توسط برنامه زمانی cron ربات تعیین می‌شود (بخش 16.6 را ببینید). ربات بین فایل‌ها و بین مدیران مکث می‌کند تا از محدودیت‌های Telegram تجاوز نشود.

> پشتیبان‌گیری از طریق ربات فقط در صورتی ارسال می‌شود که ربات در حال اجرا باشد؛ در PostgreSQL نیز نیاز به وجود `pg_dump` روی سرور دارد.

16.2. مشاهده لاگ‌ها

پنل دو ابزار مستقل برای مشاهده لاگ دارد که هر دو از برگه **«لاگ‌ها»** در «داشبورد» باز می‌شوند. هر پنجره می‌تواند به‌روزرسانی شود (آیکون «به‌روزرسانی» در عنوان) و محتوای نمایش‌داده‌شده را در فایل `x-ui.log` دانلود کند (دکمه با آیکون دانلود).

لاگ‌های پنل (برنامه / syslog)

پنجره لاگ‌های پنل (`POST /panel/api/server/logs/{count}`). کنترل‌ها:

عنصر	مقدار پیش‌فرض	توضیح
تعداد خطوط	`20`	لیست کشویی: 20 / 50 / 100 / 500 / 1000
سطح	`Info`	حداقل سطح: Debug / Info / Notice / Warning / Error
SysLog (چک‌باکس)	غیرفعال	منبع لاگ‌ها: از بافر برنامه یا از ژورنال سیستم
به‌روزرسانی خودکار (چک‌باکس)	غیرفعال	هر 5 ثانیه لاگ را دوباره بخوان (ادامه را ببینید)

رفتار بسته به چک‌باکس **SysLog** متفاوت است:

- **غیرفعال (پیش‌فرض):** لاگ‌ها از بافر حلقه‌ای داخلی پنل گرفته می‌شوند و بر اساس سطح انتخاب‌شده فیلتر می‌شوند. رکوردها با سطح (DEBUG / INFO / NOTICE / WARNING / ERROR) و منبع نمایش داده می‌شوند: `X-UI:` — پیام‌های خود پنل، `XRAY:` — پیام‌های ارسال‌شده از Xray.

> اعلان‌های ساده بدون مُهر زمانی و سطح (برای مثال پیام سیستمی «Syslog is not supported» در ویندوز) اکنون به طور کامل همانطور که هستند نمایش داده می‌شوند. فقط قالب `YYYY/MM/DD LEVEL - متن` به طور دقیق شناسایی می‌شود؛ هر چیز دیگری بدون تجزیه نمایش داده می‌شود، بنابراین چنین خطوطی دیگر کوتاه نمی‌شوند (قبلاً سه کلمه اول به اشتباه به عنوان تاریخ/زمان/سطح تفسیر می‌شدند).

- **فعال:** پنل روی سرور `journalctl -u x-ui --no-pager -n <count> -p <level>` اجرا می‌کند، یعنی ژورنال سیستم سرویس `x-ui` را نشان می‌دهد. تعداد خطوط مجاز از 1 تا 10000 است؛ سطح مقادیر syslog را می‌پذیرد (`emerg/0` ، `alert/1` ، `crit/2` ، `err/3` ، `warning/4` ، `notice/5` ، `info/6` ، `debug/7`). در ویندوز حالت SysLog پشتیبانی نمی‌شود — هشداری نمایش داده می‌شود که باید چک‌باکس را بردارید و از لاگ‌های برنامه استفاده کنید. اگر `systemd` /سرویس در دسترس نباشد، پیام خطای راه‌اندازی `journalctl` ظاهر می‌شود.

مثال: همان ژورنال از کنسول سرور. هنگامی که پنل در دسترس نیست (برای مثال راه‌اندازی نمی‌شود)، ژورنال سیستم را می‌توان مستقیماً خواند — این دقیقاً همان دستوری است که پنل در حالت SysLog اجرا می‌کند:

```
# خط با سطح warning و بالاتر آخرین 100
journalctl -u x-ui --no-pager -n 100 -p warning

# دنبال کردن ژورنال در زمان واقعی
journalctl -u x-ui -f
```

> سطح در این پنجره **خروجی** را فیلتر می‌کند. اینکه کدام حداقل سطح اصلاً در کنسول/syslog نوشته می‌شود توسط سطح لاگ‌گیری پنل تعیین می‌شود (متغیر محیطی، به طور پیش‌فرض `Info` ؛ پنل همیشه در فایل با سطح `DEBUG` می‌نویسد).

لاگ‌های دسترسی Xray (ژورنال دسترسی)

پنجره جداگانه‌ای برای access-log ایکسری (`POST /panel/api/server/xraylogs/{count}`). خطوط ژورنال دسترسی Xray را تجزیه کرده و آن‌ها را به صورت جدول نمایش می‌دهد: **Email**، **Outbound**، **Inbound**، **To**، **From**، **Date**.

از نسخه 3.4.1 این پنجره و دکمه فراخوانی آن در کارت وضعیت Xray به نام **«لاگ‌های دسترسی»** (`Access Logs`) نامیده می‌شوند — قبلاً به سادگی «لاگ‌ها» نام داشتند. تغییر نام برای جلوگیری از سردرگمی بین نمایشگر access-log ایکسری و نمایشگر لاگ‌های خود پنل انجام شده که قبلاً نام یکسانی داشت.

عنصر	مقدار پیش‌فرض	توضیح
تعداد خطوط	`20`	20 / 50 / 100 / 500 / 1000
فیلتر	خالی	جستجوی متنی بر اساس زیررشته (با فشار Enter اعمال می‌شود)
به‌روزرسانی خودکار (چک‌باکس)	غیرفعال	هر 5 ثانیه لاگ را دوباره بخوان (ادامه را ببینید)
Direct (چک‌باکس)	فعال	نمایش اتصالات مستقیم (ترافیک از طریق freedom-outbound)
Blocked (چک‌باکس)	فعال	نمایش اتصالات مسدودشده (ترافیک در blackhole-outbound)
Proxy (چک‌باکس)	فعال	نمایش ترافیک پروکسی‌شده

نوع رویداد به طور خودکار بر اساس تگ اتصال خروجی در خط لاگ تعیین می‌شود: تطابق با تگ‌های DIRECT» → freedom» (سبز)، BLOCKED» → blackhole» (قرمز)، بقیه موارد → «PROXY» (آبی). خطوط `api -> api` و خطوط خالی نادیده گرفته می‌شوند.

به‌روزرسانی خودکار. در هر دو پنجره لاگ («لاگ‌ها» و «لاگ‌های دسترسی») چک‌باکس **«به‌روزرسانی خودکار»** (`Auto Update`) وجود دارد. اگر آن را فعال کنید، محتوای لاگ هر 5 ثانیه یک‌بار با حفظ تمام تنظیمات فعلی پنجره — تعداد خطوط انتخاب‌شده، سطح/فیلتر و چک‌باکس‌های Direct / Blocked / Proxy — دوباره خوانده می‌شود. نظرسنجی به محض بسته شدن پنجره یا برداشتن چک‌باکس متوقف می‌شود.

> برای اینکه این پنجره رکوردها را نمایش دهد، Xray باید **ژورنال دسترسی** با مسیر فایل (نه `none`) فعال داشته باشد — ادامه را ببینید. اگر access-log غیرفعال باشد یا فایل در دسترس نباشد، پنجره خالی خواهد بود («No Record...»).

16.3. سطح و پیکربندی لاگ‌گیری Xray

پارامترهای لاگ‌گیری خود Xray در صفحه **«پیکربندی‌های Xray»** در بلوک **«لاگ»** (`Log`) با هشدار زیر تنظیم می‌شوند:

> «لاگ‌ها می‌توانند سرور را کند کنند. فقط در صورت نیاز انواع لاگ‌های مورد نظرتان را فعال کنید!»

فیلد	ترجمه	مقدار پیش‌فرض	توضیح
سطح لاگ‌ها (`logLevel`)	Log Level	`warning`	سطح جزئیات لاگ خطای Xray. مقادیر مجاز: `debug` ، `info` ، `notice` ، `warning` ، `error` . راهنما: «سطح ژورنال برای لاگ‌های خطا که اطلاعات مورد نیاز برای ثبت را مشخص می‌کند.»
لاگ‌های دسترسی (`accessLog`)	Access Log	`none`	مسیر فایل ژورنال دسترسی. مقدار ویژه `none` لاگ‌های دسترسی را غیرفعال می‌کند. راهنما: «مسیر فایل ژورنال دسترسی. مقدار ویژه «none» لاگ‌های دسترسی را غیرفعال می‌کند.»
لاگ‌های خطا (`errorLog`)	Error Log	خالی (مسیر پیش‌فرض)	مسیر فایل لاگ‌های خطا؛ `none` آن‌ها را غیرفعال می‌کند. راهنما: «مسیر فایل لاگ‌های خطا. مقدار ویژه «none» لاگ‌های خطا را غیرفعال می‌کند.»
لاگ‌های DNS (`dnsLog`)	DNS Log	`false` (غیرفعال)	فعال‌سازی لاگ‌گیری درخواست‌های DNS. راهنما: «فعال‌سازی لاگ‌های درخواست DNS».

فیلد	ترجمه	مقدار پیش‌فرض	توضیح
پوشش آدرس (`maskAddress`)	Mask Address	خالی (غیرفعال)	هنگام فعال‌سازی، آدرس IP واقعی به طور خودکار در لاگ‌ها با آدرس پوششی جایگزین می‌شود. راهنما: «هنگام فعال‌سازی، آدرس IP واقعی در لاگ‌ها با آدرس پوششی جایگزین می‌شود.»

از آنجایی که به طور پیش‌فرض **«لاگ‌های دسترسی» = `none`** است، پنجره «لاگ‌های Xray» (بخش 16.2) ابتدا خالی است. برای کار کردن آن، مسیر access-log را اینجا تنظیم کنید و Xray را راه‌اندازی مجدد کنید.

توجه داشته باشید: غیرفعال بودن access-log فقط روی این پنجره تأثیر می‌گذارد. لیست کلاینت‌های آنلاین در «داشبورد» و محدودیت تعداد IP در فرم کلاینت **به** access-log **وابسته نیستند** — پنل کلاینت‌های آنلاین را شناسایی کرده و آدرس‌های IP آن‌ها را از طریق API آمار آنلاین هسته Xray (آمار اتصالات) تعیین می‌کند. در نسخه‌های قدیمی هسته که این API ندارند، پنل به طور خودکار به روش قدیمی (خواندن access-log) برمی‌گردد، و در آن صورت مسیر access-log اینجا همچنان برای محدودیت IP مورد نیاز است.

محدودیت تعداد IP و fail2ban. خود محدودیت تعداد IP کلاینت (فیلد «IP Limit» در فرم کلاینت و هنگام افزودن دسته‌ای) فقط در صورت نصب **fail2ban** روی سرور اعمال می‌شود — این ابزار است که آدرس‌های تجاوزکننده از محدودیت را بن می‌کند. پنل وجود fail2ban را بررسی می‌کند (`GET /panel/api/server/fail2banStatus`)؛ اگر وجود نداشته باشد، فیلد «IP Limit» با راهنمای توضیحی غیرقابل دسترس می‌شود (در ویندوز — با پیام جداگانه)، و محدودیت‌های از قبل تنظیم‌شده در چنین سرورهایی به طور خودکار صفر می‌شوند چون اصلاً فعال نبوده‌اند. مسدودسازی fail2ban هم TCP و هم UDP را پوشش می‌دهد. در سرورهای معمولی، fail2ban اکنون در هنگام نصب و به‌روزرسانی پنل به طور خودکار نصب می‌شود (بخش 16.5 را ببینید).

مثال: بلوک `log` که باعث می‌شود پنجره «لاگ‌های Xray» شروع به نمایش رکوردها کند. در پیکربندی JSON ایکسری این چنین است:

```
{
  "log": {
    "loglevel": "warning",
    "access": "./access.log",
    "error": "",
    "dnsLog": false,
    "maskAddress": ""
  }
}
```

نکته اصلی — جایگزینی `"access": "none"` با مسیر فایل (برای مثال `"./access.log"`). پس از ذخیره، Xray را راه‌اندازی مجدد کنید و جدول در پنجره «لاگ‌های Xray» با خطوط پر خواهد شد.

16.4. مدیریت Xray: توقف و راه‌اندازی مجدد

وضعیت Xray از کارت Xray در «داشبورد» مدیریت می‌شود. وضعیت فعلی با یکی از مقادیر نمایش داده می‌شود: **در حال اجرا** (`Running`)، **متوقف** (`Stopped`)، **ناشناخته** (`Unknown`)، **خطا** (`Error`). در صورت خطا، راهنمای «خطا در راه‌اندازی Xray» در دسترس است.

دکمه	ترجمه	نقطه پایانی	عملکرد
توقف	`Stop`	`POST /panel/api/server/stopXrayService`	فرآیند Xray را متوقف می‌کند. در صورت موفقیت — اعلان هشدار «Xray service has been stopped».

دکمه	ترجمه	نقطه پایانی	عملکرد
راه‌اندازی مجدد	`Restart`	`POST /panel/api/server/restartXrayService`	Xray را با اعمال پیکربندی فعلی راه‌اندازی مجدد می‌کند (یا شروع می‌کند). در صورت موفقیت — اعلان «Xray service has been restarted successfully».

پس از هر عملیات، پنل وضعیت جدید را از طریق WebSocket پخش می‌کند، بنابراین وضعیت در «داشبورد» بدون بارگذاری مجدد صفحه به‌روزرسانی می‌شود. اگر عملیات با خطا مواجه شود، وضعیت Xray «خطا» می‌شود و متن خطا در اعلان قرار می‌گیرد.

> علاوه بر راه‌اندازی مجدد دستی، پنل خودش بررسی می‌کند که آیا Xray نیاز به راه‌اندازی مجدد دارد یا نه (وظیفه پس‌زمینه هر 30 ثانیه) و آیا فرآیند از کار افتاده است (بررسی هر ثانیه) — بخش 16.6 را ببینید.

مانیتور سلامت تونل (راه‌اندازی مجدد خودکار Xray)

در نسخه 3.4.1 **مانیتور سلامت تونل** اختیاری معرفی شد. اگر فعال باشد، پنل به صورت دوره‌ای در دسترس بودن یک URL مشخص را بررسی می‌کند و پس از چندین بررسی ناموفق متوالی، هسته Xray را به طور خودکار راه‌اندازی مجدد می‌کند — این کمک می‌کند تونلی که ترافیک را منتقل نمی‌کند بازیابی شود. به طور پیش‌فرض مانیتور **غیرفعال** است و **فقط از طریق متغیرهای محیطی سرویس** پیکربندی می‌شود (تنظیمات آن در رابط وب وجود ندارد — این عمداً توسط توسعه‌دهندگان طراحی شده است).

متغیر `XUI_TUNNEL_HEALTH_MONITOR=true` مانیتور را فعال می‌کند. متغیر `XUI_TUNNEL_HEALTH_PROXY` باید به یک inbound محلی xray اشاره کند (برای مثال `socks5://127.0.0.1:1080`) — در این صورت آزمایش از طریق خود Xray می‌رود و دقیقاً تونل را بررسی می‌کند؛ بدون آن، فقط اتصال‌پذیری هاست بررسی می‌شود و راه‌اندازی مجدد مشکل اتصال شبکه سرور را برطرف نمی‌کند. سایر متغیرها پارامترهای بررسی را تعیین می‌کنند:

متغیر	هدف	پیش‌فرض
`XUI_TUNNEL_HEALTH_MONITOR`	فعال‌سازی مانیتور (روشن/خاموش)	`false`
`XUI_TUNNEL_HEALTH_PROXY`	پروکسی که از آن آزمایش می‌رود (inbound محلی xray را وارد کنید)	خالی
`XUI_TUNNEL_HEALTH_URL`	URL که بررسی می‌شود	`https://www.cloudflare.com/cdn-cgi/trace`
`XUI_TUNNEL_HEALTH_INTERVAL`	فاصله بین بررسی‌ها	`30s`
`XUI_TUNNEL_HEALTH_TIMEOUT`	زمان انتظار یک بررسی	`10s`
`XUI_TUNNEL_HEALTH_FAILURES`	تعداد شکست‌های متوالی تا راه‌اندازی مجدد	`3`
`XUI_TUNNEL_HEALTH_COOLDOWN`	حداقل مکث بین راه‌اندازی‌های مجدد	`5m`

> راه‌اندازی مجدد Xray اتصالات تمام کلاینت‌های متصل را قطع می‌کند، بنابراین منطقی است فاصله و آستانه تعداد شکست‌ها به اندازه کافی بزرگ باشند تا یک خرابی تصادفی در یک آزمایش منجر به راه‌اندازی‌های مجدد غیرضروری نشود.

16.5. راه‌اندازی مجدد و به‌روزرسانی پنل

راه‌اندازی مجدد پنل

در صفحه **«تنظیمات پنل»** عملکرد **«راه‌اندازی مجدد پنل»** (`Restart Panel` ، `POST /panel/api/setting/restartPanel`) وجود دارد. با تأیید، پنل **پس از 3 ثانیه** راه‌اندازی مجدد می‌شود.

پیام‌ها:

- تأییدیه: «آیا مطمئنید که می‌خواهید پنل را راه‌اندازی مجدد کنید؟ تأیید کنید و راه‌اندازی مجدد پس از 3 ثانیه اتفاق می‌افتد. اگر پنل در دسترس نبود، لاگ سرور را بررسی کنید.»
- موفقیت: «پنل با موفقیت راه‌اندازی مجدد شد».

از نظر فنی در لینوکس، راه‌اندازی مجدد با ارسال سیگنال `SIGHUP` به فرآیند پنل (یا از طریق هوک ثبت‌شده) انجام می‌شود. در ویندوز ارسال `SIGHUP` پشتیبانی نمی‌شود.

به‌روزرسانی خودکار پنل (Update Panel)

در «داشبورد» عملکرد **«به‌روزرسانی پنل»** (`Update Panel`) در دسترس است — به‌روزرسانی 3X-UI به آخرین نسخه مستقیماً از رابط وب.

قبل از به‌روزرسانی، پنل نسخه‌ها را مقایسه می‌کند (`GET /panel/api/server/getPanelUpdateInfo`) و آخرین نسخه 3x-ui را از GitHub درخواست می‌کند:

فیلد	ترجمه
نسخه فعلی پنل	Current panel version
آخرین نسخه پنل	Latest panel version
پنل به‌روز است / «به‌روزرسانی شده»	Panel is up to date / Up to date — نمایش داده می‌شود اگر نسخه جدیدی وجود نداشته باشد

شروع به‌روزرسانی — `POST /panel/api/server/updatePanel` . گفتگوی تأییدیه:

- «آیا واقعاً می‌خواهید پنل را به‌روزرسانی کنید؟»
- «این عمل 3X-UI را به نسخه #version# به‌روزرسانی کرده و سرویس پنل را راه‌اندازی مجدد می‌کند.»

پس از شروع — پیام بازشونده «به‌روزرسانی پنل شروع شد» (`Panel update started`)؛ در صورت عدم موفقیت در بررسی نسخه‌ها — «بررسی به‌روزرسانی پنل ناموفق بود» (`Panel update check failed`).

آنچه روی سرور اتفاق می‌افتد: به‌روزرسانی خودکار **فقط در لینوکس** پشتیبانی می‌شود (در سیستم‌عامل‌های دیگر خطای «panel web update is supported only on Linux installations» برگردانده می‌شود). پنل اسکریپت رسمی `update.sh` را از GitHub (`raw.githubusercontent.com/MHSanaei/3x-ui/main/update.sh`) دانلود کرده و آن را در یک فرآیند جداگانه اجرا می‌کند: ترجیحاً از طریق `systemd-run` در یک unit جداگانه (`x-ui-web-update-<timestamp>`)، و در غیاب systemd — به عنوان یک فرآیند جداشده مستقل. پس از اتمام، اسکریپت اجزا را به‌روزرسانی کرده و سرویس پنل را راه‌اندازی مجدد می‌کند. اجرا نیاز به `bash` دارد.

اگر در حین به‌روزرسانی اسکریپت یک مسیر پایه وب جدید تصادفی (Web Base Path) ایجاد کند، سرویس `x-ui` به طور خودکار راه‌اندازی مجدد می‌شود تا مسیر جدید فوراً فعال شود. (بدون راه‌اندازی مجدد، سرور همچنان مسیر قدیمی را ارائه می‌داد، در حالی که رابط مسیر جدید را نشان می‌داد و آدرس جدید تا راه‌اندازی مجدد دستی در دسترس نبود.)

کانال Dev (rolling)، از 3.4.2. دکمه نسخه پنل در «داشبورد» اکنون همیشه پنجره به‌روزرسانی را باز می‌کند (قبلاً روی ساخت پایدار جاری به صفحه انتشارهای GitHub می‌رفت)، و کلید **«کانال Dev»** (`devChannelEnable`) همیشه نمایش داده می‌شود — حتی روی ساخت پایدار. با

فعال کردن آن، در به‌روزرسانی بعدی می‌توان به کانال «غلتان» dev منتقل شد (ساخت‌ها به‌ازای هر commit در `main`). هنگام فعال‌سازی هشداری نمایش داده می‌شود: «ساخت‌های Dev هر commit در main را دنبال می‌کنند و نسخه‌های پایدار نیستند — بازگشت خودکار وجود ندارد». بدون اقدام صریح کاربر، چیزی به‌روزرسانی نمی‌شود.

کانال به‌روزرسانی Dev (ساخت‌های rolling بر اساس commit)

علاوه بر به‌روزرسانی معمول به نسخه پایدار، **«کانال توسعه»** (`Dev`) اختیاری وجود دارد. سوئیچ **فقط در ساخت‌های dev** (ساخت‌های CI که برای commit جداگانه‌ای ساخته شده‌اند) در پنجره به‌روزرسانی پنل ظاهر می‌شود؛ در نسخه‌های پایدار نمایش داده نمی‌شود. هنگام فعال‌سازی، پنل به ساخت rolling `dev-latest` که هر commit از شاخه `main` را دنبال می‌کند به‌روزرسانی خواهد شد و یک نسخه پایدار نیست — هشداری نمایش داده می‌شود مبنی بر اینکه ساخت‌های dev ناپایدار هستند و بازگشت خودکار وجود ندارد. در حالت dev، پنجره «commit فعلی» / «آخرین commit» را به جای شماره‌های نسخه نشان می‌دهد. این ویژگی فقط در لینوکس با systemd در دسترس است.

در ساخت‌های dev، پنل نسخه خود را به صورت `dev+<commit-کوتاه>` به جای شماره پایدار گمراه‌کننده نشان می‌دهد — در نشان نوار کناری، در کارت «داشبورد»، در پنجره به‌روزرسانی، در گزارش وضعیت ربات Telegram و در خروجی دستور `x-ui -v`. در نسخه‌های پایدار، نمایش نسخه تغییر نمی‌کند.

> در نودها، پنل همان 3x-ui به طور متمرکز از طریق `POST /panel/api/nodes/updatePanel` به‌روزرسانی می‌شود — بخش مربوط به نودها را ببینید.

نصب خودکار fail2ban

تا اینکه محدودیت تعداد IP کلاینت‌ها (بخش 16.3) از ابتدا کار کند، هنگام نصب و به‌روزرسانی پنل روی یک سرور معمولی `fail2ban` اکنون به طور خودکار نصب و پیکربندی می‌شود (قبلاً این فقط در تصویر Docker اتفاق می‌افتاد). رفتار با متغیر محیطی `XUI_ENABLE_FAIL2BAN` کنترل می‌شود: پیکربندی اگر متغیر تنظیم نشده باشد یا برابر `true` باشد انجام می‌شود. اجرای دستی با دستور `x-ui setup-fail2ban` در دسترس است. خرابی در پیکربندی fail2ban نصب یا به‌روزرسانی پنل را متوقف نمی‌کند.

نصب و به‌روزرسانی روی هاست‌های IPv6-only

اسکریپت‌های `install.sh` و `update.sh` اکنون به درستی روی سرورهایی که فقط IPv6 دارند کار می‌کنند: دانلود نسخه، اسکریپت `x-ui.sh` و فایل‌های سرویس دیگر از IPv4 اجباری (`curl -4`) استفاده نمی‌کنند و پروتکل در دسترس را می‌گیرند. بنابراین پنل را می‌توان روی هاستی بدون آدرس IPv4 نیز نصب و به‌روزرسانی کرد.

بازنویسی پورت پنل با متغیر `XUI_PORT`

پورت شنود وب‌پنل را می‌توان با متغیر محیطی `XUI_PORT` بازنویسی کرد — این تنها در طول اجرای فرآیند فعلی تأثیر دارد و مقدار ذخیره‌شده `webPort` در پایگاه داده را **تغییر نمی‌دهد**. مقادیر معتبر از `1` تا `65535` هستند؛ مقدار خالی، نادرست یا خارج از محدوده نادیده گرفته می‌شود (`webPort` استفاده می‌شود) با یک هشدار در لاگ. این در هنگام استقرار مفید است، به ویژه در Docker: هنگام استفاده از شبکه bridge، پورت منتشرشده کانتینر باید با `XUI_PORT` مطابقت داشته باشد — برای مثال، `XUI_PORT=8080` و `ports: "8080:8080"`.

به‌روزرسانی و تغییر نسخه Xray-core

همچنین در «داشبورد» می‌توانید نسخه Xray-core را جداگانه از پنل مدیریت کنید.

- **به‌روزرسانی‌های Xray** (`Xray Updates`) / **انتخاب نسخه** (`Version`) — لیست کشویی نسخه‌های موجود. راهنماها: «نسخه مورد نظر را انتخاب کنید» و هشدار «مهم: نسخه‌های قدیمی ممکن است تنظیمات فعلی را پشتیبانی نکنند».
- نصب/تغییر نسخه — `POST /panel/api/server/installXray/{version}`. گفتگو: «تغییر نسخه Xray» / «آیا واقعاً می‌خواهید نسخه Xray را تغییر دهید؟». در صورت موفقیت — «Xray با موفقیت به‌روزرسانی شد».

مثال: تغییر نسخه Xray-core از طریق درخواست API. نسخه با تگ نسخه از XTLS/Xray-core (با پیشوند `v`) مشخص می‌شود. برای مثال، تغییر به `v1.8.24` :

```
curl -s -b cookies.txt -X POST \
     https://panel.example.com:2053/panel/api/server/installXray/v1.8.24
```

(`cookies.txt` — فایل کوکی از مثال در بخش 16.1.) پس از نصب، Xray به طور خودکار با نسخه انتخاب‌شده راه‌اندازی مجدد می‌شود.

روی سرور هنگام تغییر نسخه، Xray ابتدا متوقف می‌شود، آرشیو نسخه مورد نظر از GitHub (XTLS/Xray-core) دانلود می‌شود، باینری استخراج شده و جایگزین می‌شود، و سپس Xray با بررسی کنترل‌های اندازه آرشیو/باینری راه‌اندازی مجدد می‌شود.

16.6. وظایف دوره‌ای (cron)

پنل هنگام راه‌اندازی تعدادی وظیفه پس‌زمینه ثبت می‌کند. برنامه‌های زمانی آن‌ها ثابت هستند (در UI قابل پیکربندی نیستند، به استثنای برنامه زمانی گزارش Telegram و همگام‌سازی LDAP). در ادامه وظایف مربوط به بهره‌برداری آمده است.

وظیفه	برنامه زمانی	هدف
بررسی عملکرد Xray	هر 1 ثانیه	کنترل اینکه فرآیند Xray در حال اجرا است
بررسی نیاز به راه‌اندازی مجدد Xray	هر 30 ثانیه	راه‌اندازی مجدد اگر پیکربندی به عنوان تغییریافته علامت‌گذاری شده باشد
جمع‌آوری ترافیک Xray	هر 5 ثانیه (شروع 5 ثانیه پس از راه‌اندازی)	حسابداری ترافیک inbound/کلاینت‌ها
بررسی IP کلاینت‌ها	هر 10 ثانیه	کنترل محدودیت IP از طریق لاگ
Heartbeat و همگام‌سازی ترافیک نودها	هر 5 ثانیه	تبادل با نودها
پاکسازی لاگ‌ها	**روزانه** (`@daily`)	لاگ‌های محدودیت IP و access-log دائمی را پاک می‌کند و لاگ فعلی را به `*.prev.log` می‌چرخاند
بازنشینی ترافیک بر اساس دوره	`@hourly` ، `@daily` ، `@weekly` ، `@monthly`	شمارنده‌های ترافیک inboundهایی (و کلاینت‌هایشان) که دوره بازنشینی خودکار مربوطه برایشان تنظیم شده را بازنشینی می‌کند
گزارش Telegram	در تنظیمات ربات تعیین می‌شود (به طور پیش‌فرض `@daily`)	ارسال گزارش به مدیران؛ اگر گزینه فعال باشد — با نسخه پشتیبان ضمیمه‌شده پایگاه داده (بخش 16.1)
بازنشینی مخزن hash Telegram	هر 2 دقیقه	فقط هنگام فعال بودن ربات
کنترل بار CPU برای Telegram	هر 10 ثانیه	فقط اگر آستانه CPU > 0 تنظیم شده باشد

علاوه بر این:

- **بازنشینی دوره‌ای ترافیک** فقط برای inboundهایی که حالت بازنشینی خودکار مربوطه (هر ساعت/روزانه/هفتگی/ماهانه) برایشان انتخاب شده فعال می‌شود. وظیفه ترافیک خود inbound و تمام کلاینت‌هایش را بازنشینی می‌کند.
- **بررسی انقضا و تخلیه.** غیرفعال‌سازی کلاینت‌ها پس از انقضای مدت زمان و تخلیه محدودیت ترافیک در چارچوب حسابداری ترافیک انجام می‌شود: کلاینت‌هایی با `expiry_time` منقضی‌شده یا حجم تخلیه‌شده علامت‌گذاری و غیرفعال می‌شوند، در صورت نیاز مهلت بعدی محاسبه می‌شود (برای محدودیت‌های چرخه‌ای و حالت «شمارش از اولین استفاده»). در «داشبورد» و لیست‌ها این با وضعیت‌های «منقضی‌شده»/«تخلیه‌شده»/«به زودی تمام می‌شود» منعکس می‌شود.
- **پشتیبان‌گیری خودکار در Telegram** — این یک اثر جانبی وظیفه گزارش است، برنامه زمانی cron جداگانه‌ای فقط برای پشتیبان‌گیری وجود ندارد. بنابراین فرکانس پشتیبان‌گیری خودکار برابر فرکانس گزارش ربات است.

16.7. منوی کنسول و CLI (`x-ui`)

روی سرور، پنل با دستور `x-ui` مدیریت می‌شود. بدون آرگومان، منوی تعاملی «3X-UI Panel Management Script» باز می‌شود؛ با آرگومان، زیردستور مشخصی اجرا می‌شود. موارد منو مربوط به بهره‌برداری:

شماره در منو	مورد	عملکرد
1	Install	نصب پنل (دانلود و اجرای `install.sh`)
2	Update	به‌روزرسانی تمام اجزای x-ui به آخرین نسخه بدون از دست دادن داده؛ پس از آن — راه‌اندازی مجدد خودکار
3	Update to Dev Channel (latest commit)	به‌روزرسانی به ساخت rolling `dev-latest` (آخرین commit از شاخه `main`) با تأییدیه (بخش 16.5 را ببینید)
4	Update Menu	به‌روزرسانی فقط اسکریپت منوی `x-ui`
5	Legacy Version	نصب نسخه مشخص‌شده (قدیمی) پنل بر اساس شماره وارده (برای مثال `2.4.0`)
6	Uninstall	حذف کامل پنل و Xray (بخش 16.8 را ببینید)
7	Reset Username & Password	بازنشینی نام کاربری/رمز عبور مدیر
8	Reset Web Base Path	بازنشینی مسیر پایه وب‌پنل
9	Reset Settings	بازنشینی تنظیمات به مقادیر پیش‌فرض
10	Change Port	تغییر پورت پنل
11	View Current Settings	مشاهده تنظیمات فعلی
14–12	Start / Stop / Restart	شروع، توقف، راه‌اندازی مجدد سرویس پنل
15	Restart Xray	راه‌اندازی مجدد فقط Xray
16	Check Status	وضعیت فعلی سرویس
17	Logs Management	مشاهده و پاکسازی لاگ‌ها (ادامه را ببینید)
19–18	Enable / Disable Autostart	فعال/غیرفعال کردن راه‌اندازی خودکار سرویس هنگام شروع سیستم‌عامل
27	Update Geo Files	به‌روزرسانی فایل‌های جغرافیایی (GeoIP/GeoSite)
25	PostgreSQL Management	مدیریت PostgreSQL

> شماره‌گذاری موارد منو در نسخه 3.4.1 تغییر کرد: به دلیل افزوده شدن مورد 3 «Update to Dev Channel»، تمام موارد بعدی یک شماره به جلو رفتند. تعداد کل موارد 28 شد، انتخاب در محدوده `[0–28]` وارد می‌شود.

مدیریت لاگ‌ها در CLI (مورد 16)

زیرمنوی «Logs Management» اکنون با مورد **17** باز می‌شود (قبلاً — 16):

- **Debug Log** — مشاهده جریانی ژورنال سرویس: `journalctl -u x-ui -e --no-pager -f -p debug` (در Alpine — `grep` روی `/var/log/messages`).
- **Clear All logs** — پاکسازی ژورنال سیستم: `journalctl --rotate` + `journalctl --vacuum-time=1s` ، پس از آن سرویس راه‌اندازی مجدد می‌شود. (در Alpine در دسترس نیست.)

زیردستورات مستقیم `x-ui`

تمام زیردستورات موجود:

دستور	توضیح
`x-ui`	باز کردن منوی مدیریت
`x-ui start`	شروع پنل
`x-ui stop`	توقف پنل
`x-ui restart`	راه‌اندازی مجدد پنل
`x-ui restart-xray`	راه‌اندازی مجدد Xray
`x-ui status`	وضعیت فعلی
`x-ui settings`	نمایش تنظیمات فعلی
`x-ui enable`	فعال‌سازی راه‌اندازی خودکار هنگام شروع سیستم‌عامل
`x-ui disable`	غیرفعال‌سازی راه‌اندازی خودکار
`x-ui log`	مشاهده لاگ‌ها
`x-ui banlog`	مشاهده لاگ‌های banهای Fail2ban
`x-ui setup-fail2ban`	نصب و پیکربندی fail2ban برای محدودیت IP (بخش 16.5 را ببینید)
`x-ui update`	به‌روزرسانی پنل

| `x-ui update-dev` | به‌روزرسانی پنل به کانال توسعه (ساخت rolling `dev-latest`) || `x-ui update-all-geofiles` | به‌روزرسانی تمام فایل‌های جغرافیایی (با راه‌اندازی مجدد بعدی) || `x-ui migrateDB [file]` | تبدیل پایگاه داده `.db ⇄ .dump` (SQLite) || `x-ui legacy` | نصب نسخه قدیمی || `x-ui install` | نصب پنل || `x-ui uninstall` | حذف پنل |

> دستور `x-ui update` اسکریپت رسمی `update.sh` را دانلود کرده و اجرا می‌کند (همانند به‌روزرسانی وب از بخش 16.5)، با درخواست تأییدیه: «This function will update all x-ui components to the latest version, and the data will not be lost.» پس از اتمام، پنل به طور خودکار راه‌اندازی مجدد می‌شود.

> **پرچم‌های `-webCert` / `-webCertKey` در زیردستور `setting`.** مسیرهای گواهی و کلید خصوصی وب‌پنل را می‌توان مستقیماً در زیردستور `x-ui setting -webCert <مسیر> -webCertKey <مسیر>` تنظیم کرد — مشخص کردن هر یک از این پرچم‌ها مسیر مربوطه را ذخیره می‌کند (مانند زیردستور `cert` جداگانه)، و پنل بلافاصله به HTTPS تغییر می‌کند.

دریافت API-token از طریق CLI

دستور دریافت API-token از طریق CLI (مورد منو/دستور `x-ui`) توکن قبلاً صادرشده را نشان نمی‌دهد. API-tokenها فقط به صورت hash ذخیره می‌شوند، بنابراین توکن موجود را نمی‌توان به صورت متن ساده دریافت کرد. اگر توکن‌ها از قبل پیکربندی شده باشند، دستور تعداد آن‌ها را اعلام می‌کند، توصیه می‌کند توکن‌ها را در پنل مدیریت کنید (**Settings** → **API Tokens**، بخش مربوط به API-tokenها را ببینید) و بلافاصله یک **توکن ذخیره جدید** با نامی به شکل `cli-fallback-<timestamp>` تولید کرده و آن را نمایش می‌دهد تا CLI بدون ورود به رابط کاربری مفید باقی بماند.

16.8. حذف پنل

حذف از CLI انجام می‌شود — مورد منو **5 (Uninstall)** یا دستور `x-ui uninstall`. قبل از حذف، تأییدیه درخواست می‌شود (به طور پیش‌فرض «نه»): «Are you sure you want to uninstall the panel? xray will also uninstalled!».

با تأیید، اسکریپت:

1. سرویس را متوقف کرده و راه‌اندازی خودکار آن را غیرفعال می‌کند (`systemctl stop/disable x-ui`، یا در Alpine — `rc-service/rc-update`)، فایل unit سرویس را حذف کرده و پیکربندی systemd را بارگذاری مجدد می‌کند.
2. دایرکتوری‌های داده و برنامه (`/etc/x-ui/`، دایرکتوری نصب) و فایل env سرویس (`/etc/default/x-ui`، `/etc/conf.d/x-ui` یا `/etc/sysconfig/x-ui` — بسته به توزیع) را حذف می‌کند.
3. خود اسکریپت `x-ui` را حذف کرده و پیام «Uninstalled Successfully.» و همچنین دستور نصب مجدد را نمایش می‌دهد.

اگر پنل از PostgreSQL استفاده می‌کرد (در فایل env `XUI_DB_TYPE=postgres`)، پس از حذف فایل‌های پنل، اسکریپت به علاوه می‌پرسد که آیا سرور PostgreSQL به همراه تمام پایگاه داده‌هایش نیز حذف شود: «Also purge PostgreSQL and delete all of its data?». درخواست نیاز به تأیید صریح دارد (به طور پیش‌فرض — رد) و با هشدار همراه است: حذف **تمام** پایگاه داده‌های PostgreSQL روی دستگاه را تحت تأثیر قرار می‌دهد، از جمله آن‌هایی که به برنامه‌های دیگر تعلق دارند، و برگشت‌ناپذیر است. با رد کردن، PostgreSQL و داده‌هایش دست‌نخورده باقی می‌مانند.

> حذف برگشت‌ناپذیر است: Xray و تمام داده‌ها (از جمله پایگاه داده) همراه با پنل حذف می‌شوند. اگر داده‌ها ممکن است مورد نیاز باشند، از قبل خروجی پایگاه داده را بگیرید (بخش 16.1).

16.9. دستور `x-ui migrateDB`

از نسخه 3.3.0، اسکریپت مدیریت `x-ui.sh` زیردستور `migrateDB` را دریافت کرد — یک wrapper دور باینری داخلی `x-ui` (`x-ui migrate-db`) برای تبدیل پایگاه داده پنل SQLite بین دو قالب: باینری `.db` و دامپ متنی قابل حمل `.dump` (متن SQL معمولی).

آنچه دستور انجام می‌دهد

دستور در دو جهت کار می‌کند و جهت به طور **خودکار** بر اساس فایل ورودی تعیین می‌شود:

جهت	نامیده می‌شود	آنچه اتفاق می‌افتد
`.db → .dump`	dump (استخراج)	پایگاه داده باینری SQLite در یک فایل SQL متنی استخراج می‌شود
`.dump → .db`	restore (بازیابی)	پایگاه داده باینری SQLite از فایل SQL متنی دوباره ساخته می‌شود

زیر کاپوت، اسکریپت باینری پنل را فراخوانی می‌کند:

- استخراج: `x-ui migrate-db --src <ورودی> --dump <خروجی>`
- بازیابی: `x-ui migrate-db --restore <ورودی> --out <خروجی>`

نحو فراخوانی

```
x-ui migrateDB [file.db|file.dump] [output]
```

- **`[file.db|file.dump]`** — فایل ورودی (آرگومان اول). اگر مشخص نشود، پایگاه داده نصب‌شده پنل به طور پیش‌فرض گرفته می‌شود: `/etc/x-ui/x-ui.db`.
- **`[output]`** — مسیر به فایل خروجی (آرگومان دوم). اختیاری است: در صورت نبود، نام به طور خودکار در کنار فایل ورودی انتخاب می‌شود (ادامه را ببینید).

مثال‌ها:

```
x-ui migrateDB                          # استخراج /etc/x-ui/x-ui.db -> /etc/x-ui/x-ui.dump
x-ui migrateDB /etc/x-ui/x-ui.db backup.dump
x-ui migrateDB backup.dump restored.db      # ساختن .db از دامپ
```

نحوه تعیین جهت

اسکریپت پسوند فایل ورودی را بررسی می‌کند:

- `*.db`، `*.sqlite`، `*.sqlite3` → حالت **dump** (استخراج به متن)؛
- `*.dump`، `*.sql` → حالت **restore** (ساختن پایگاه داده).

اگر پسوند شناسایی نشود، اسکریپت 16 بایت اول فایل را می‌خواند: امضای `SQLite format 3` به معنای پایگاه داده باینری (حالت dump) است، در غیر این صورت فایل به عنوان دامپ (حالت restore) در نظر گرفته می‌شود.

نام فایل خروجی، اگر آرگومان دوم مشخص نشده باشد:

- هنگام استخراج — همان نام ورودی با پسوند `.dump`؛
- هنگام بازیابی — همان نام با پسوند `.db`.

بررسی‌های حفاظتی و رفتار

- **وجود باینری.** اگر باینری `x-ui` پیدا نشود یا اجرایی نباشد — خطای «x-ui binary not found ... Is the panel installed?» نمایش داده می‌شود.
- **پشتیبانی از ویژگی در ساخت.** اسکریپت بررسی می‌کند که باینری قادر به `migrate-db --dump/--restore` است (از طریق `x-ui migrate-db -h`). اگر نه — پیشنهاد می‌شود ابتدا پنل را با دستور `x-ui update` به‌روزرسانی کنید.

- **وجود فایل ورودی.** در صورت نبود فایل ورودی، خطا و خط نحو فراخوانی چاپ می‌شود.
- **بازنویسی خروجی.** اگر فایل خروجی از قبل وجود داشته باشد، تأییدیه درخواست می‌شود (به طور پیش‌فرض «نه»)؛ بدون تأیید، عملیات لغو می‌شود. هنگام بازیابی، فایل خروجی قدیمی از قبل حذف می‌شود.
- **حفاظت از پایگاه داده «زنده».** هنگام بازیابی در پایگاه داده پیش‌فرض `/etc/x-ui/x-ui.db` ، وقتی پنل در حال اجرا است، عملیات با الزام به توقف اول پنل (`x-ui stop`) یا انتخاب مسیر خروجی دیگری رد می‌شود. این از بازنویسی پایگاه داده کاری سرویس در حال اجرا جلوگیری می‌کند.
- در صورت شکست در ساختن پایگاه داده، فایل خروجی ناقص حذف می‌شود.

چرا این مفید است

- **پشتیبان‌گیری.** دامپ متنی `.dump` قابل خواندن توسط انسان است، برای ذخیره در سیستم‌های کنترل نسخه و مشاهده تفاضلی محتوای پایگاه داده راحت است.
- **انتقال.** دامپ بین دستگاه‌ها قابل حمل است و در برابر تفاوت‌های نسخه قالب فایل SQLite مقاوم است — روی سرور جدید `.db` کاری از آن ساخته می‌شود.
- **تشخیص.** از `.dump` می‌توان ساختار و داده‌های پنل را بدون داشتن ابزارهای SQLite در دسترس با چشم مشاهده کرد.

حالت تعاملی

علاوه بر فراخوانی مستقیم، تبدیل از منوی تعاملی نیز در دسترس است. در زیرمنوی PostgreSQL (`x-ui` → بخش کار با PostgreSQL) مورد **9. Convert SQLite `.db` <-> `.dump`** وجود دارد: مسیر فایل ورودی (به طور پیش‌فرض `/etc/x-ui/x-ui.db`) و خروجی (می‌توان برای نام‌گذاری خودکار خالی گذاشت) را می‌پرسد، و جهت مانند حالت CLI به طور خودکار تعیین می‌شود.

این سند بر اساس کد منبع *3X-UI* تهیه شده است. اگر مورد رابط کاربری در نسخه شما متفاوت است — رفتار پنل و راهنماهای خود *UI* اولویت دارند.