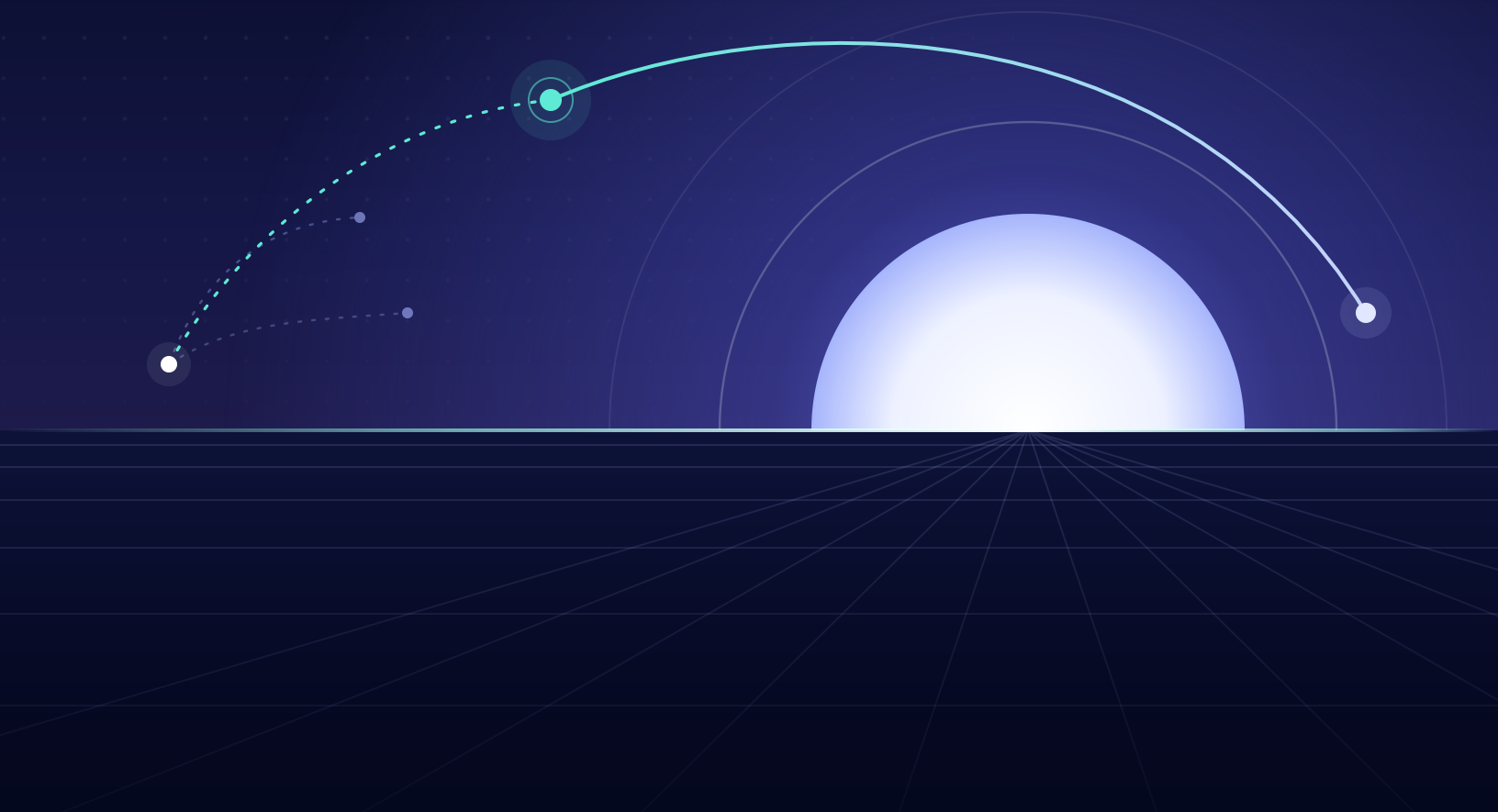
ProxyHorizon

2026 EDITION

A PROXYHORIZON FIELD GUIDE

The Ultimate Proxy Handbook

How proxies actually work, how to choose the right one, and how to run them at scale without getting blocked.



- From first request to production pipelines
- Copy-paste code for cURL, Python, Node and Playwright
- Checklists, glossary and quick reference

BEFORE YOU BEGIN

About This Handbook

Proxies sit underneath price trackers, SEO tools, ad verification, research pipelines and, more and more, AI agents. Yet most guides either sell you a product or stop at a definition. This one explains how proxies work, how to choose them and how to run them well, in plain English and with working code.

Who It's For

Developers and data engineers

You build scrapers and pipelines and you want fewer blocks per dollar.

START WITH chapters 3, 4, 7 and 8

Growth, SEO and e-commerce teams

You need reliable data from many locations and want to buy the right plan the first time.

START WITH chapters 2, 5 and 6

AI and automation builders

Your agents browse the web and need identities that hold up under scrutiny.

START WITH chapters 2, 7, 8 and 10

How to Read It

Read it front to back once, then keep it open as a reference. Every chapter ends with key takeaways, and the appendix holds the checklists, glossary and quick reference you'll reach for when something breaks at 2 a.m.

Signposts you'll see

Pro tip
A shortcut that saves time or money.

Note
Context worth knowing before you move on.

Watch out
A common mistake and how to avoid it.

Risk
Security, legal or ethical exposure.

Pro move
What experienced teams do differently.

Go deeper
A ProxyHorizon guide or free tool to explore next.

2026 Edition. Provider figures (proxy types, pool sizes, country coverage and entry prices) come from the ProxyHorizon directory as of October 2026. Prices move often; proxyhorizon.com/proxies has live numbers. Example IPs use ranges reserved for documentation (RFC 5737) and `gate.example.net` stands in for your provider's gateway. Nothing here is legal advice.

© 2026 ProxyHorizon. Share this PDF freely, unchanged and with credit.

YOUR MAP

Contents

	Why This Handbook Exists The problem proxies solve, and the mistakes this guide helps you skip.	04
01	**How Proxies Work** What a proxy changes, what it can't, and what websites see.	05
02	**The Four Proxy Types** Datacenter, ISP, residential and mobile, compared honestly.	10
03	**Protocols and Authentication** HTTP, HTTPS and SOCKS5, plus the anatomy of a proxy URL.	17
04	**Rotation, Sessions and Geo-Targeting** Gateways, sticky sessions and location targeting done right.	21
05	**Use Cases and Playbooks** Proven setups for scraping, SEO, ads, social and AI agents.	25
06	**Choosing a Provider** A 10-point framework, true-cost math and nine networks to test.	30
07	**Setup and Integration** Working setups for cURL, Python, Node.js, Playwright and more.	36
08	**Staying Unblocked** How anti-bot systems score you, and how to troubleshoot blocks.	42
09	**Scaling and Cost Control** Measure success per dollar and cut bandwidth without losing data.	48
10	**Security, Ethics and What's Next** Safe setups, responsible collection and where proxies are heading.	52
A	**Appendix** Glossary, checklists, quick reference and free tools.	58

INTRODUCTION

Why This Handbook Exists

Your scraper ran clean on Tuesday. On Wednesday every request comes back `403 Forbidden`, and nothing in your code changed. What changed is how the website sees you: same IP, same TLS handshake, same request rhythm, now flagged.

That's the moment most people start shopping for proxies, usually in a hurry and usually asking the wrong question. "Which provider is best?" has no answer until you know which proxy type your targets tolerate, how your traffic should rotate, and what one successful request is actually worth to you.

> **A proxy doesn't make you invisible. It changes who the website thinks you are.**

ProxyHorizon tracks more than 120 proxy providers in its directory, and the same handful of mistakes come up again and again. Residential bandwidth gets burned on images nobody needed. Sticky sessions get used where rotation was the answer. Teams scrape through a VPN and wonder why one shared IP keeps getting blocked. Someone grabs a free proxy list and hands every request, and sometimes every password, to a stranger.

This handbook exists so you can skip those mistakes. It's opinionated where the evidence is clear and honest about trade-offs where it isn't.

By the End, You'll Be Able To

- Explain exactly what a proxy changes, and what it leaves exposed.
- Pick the right proxy type for a target before you spend a dollar.
- Set up rotation, sticky sessions and geo-targeting with confidence.
- Read any provider's pricing page and work out your true cost per successful request.
- Wire proxies into cURL, Python, Node.js, Playwright and antidetect browsers.
- Diagnose a block from its symptoms and fix the cause, not just the symptom.
- Collect data securely and responsibly, with the legal basics in mind.

Start at Chapter 1, even if you've used proxies before

The mental model it builds, network identity versus client identity, is the lens for every decision in the chapters that follow.

01



How Proxies Work

A proxy changes one thing about your request: the address it comes from. Get a precise picture of that one thing and every later decision in this handbook gets easier, from the proxy type you buy to the way you debug a block.

IN THIS CHAPTER

1.1 The journey of a proxied request

1.2 Forward, reverse, VPN and Tor

1.3 What a proxy hides, and what it doesn't

1.4 The IP address, decoded

1.1 The Journey of a Proxied Request

Type a URL and press Enter. Without a proxy, your device opens a connection straight to the website's server. That server sees your public IP address, the one your internet provider gave your router, and it logs it, geolocates it and scores it before it sends back a single byte.

Put a proxy in the middle and the trip changes. Your client sends the request to the proxy server instead. The proxy opens its own connection to the website, forwards your request and relays the answer back. As far as the website can tell, the visitor *is* the proxy's **exit IP**. Your own address never reaches it.

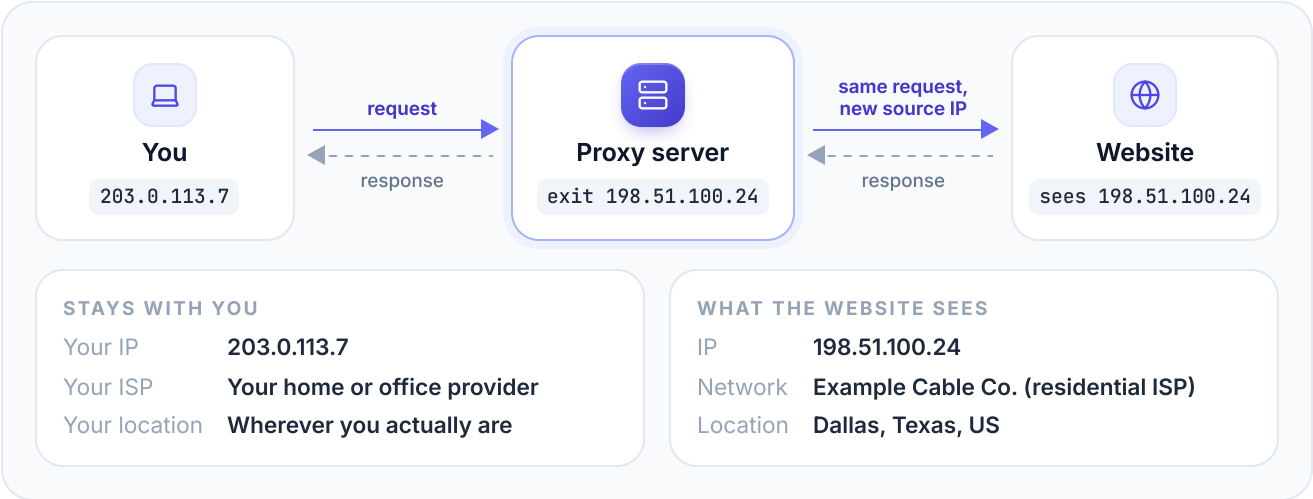


Figure 1.1 The website only ever talks to the proxy. Your real IP stops at the first hop, and the site judges the exit IP instead.

Why people put a proxy in the path

Appear somewhere else

See the prices, search results and ads that users in another city or country see.

Spread the load

Split a big job across thousands of IPs so no single address trips a site's per-IP limits.

Control traffic

Companies route staff traffic through proxies to filter, cache and audit it.

Proxies work per app. VPNs work per device.

A proxy only carries traffic from software you point at it: a scraper, one browser profile, a single tool. A VPN captures everything your device sends. That's why one machine can run a thousand proxy identities at once but only one VPN connection.

1.2 Forward Proxies, Reverse Proxies and Their Cousins

Everything you buy from a proxy provider is a **forward proxy**: it works for the client and sends your requests out under its own address. A **reverse proxy** works for the server. It sits in front of a website, receives every visitor's request and decides what happens next: serve it from cache, spread it across servers, or block it. Cloudflare, Akamai and Nginx load balancers are reverse proxies, and the anti-bot checks in Chapter 8 run inside them.

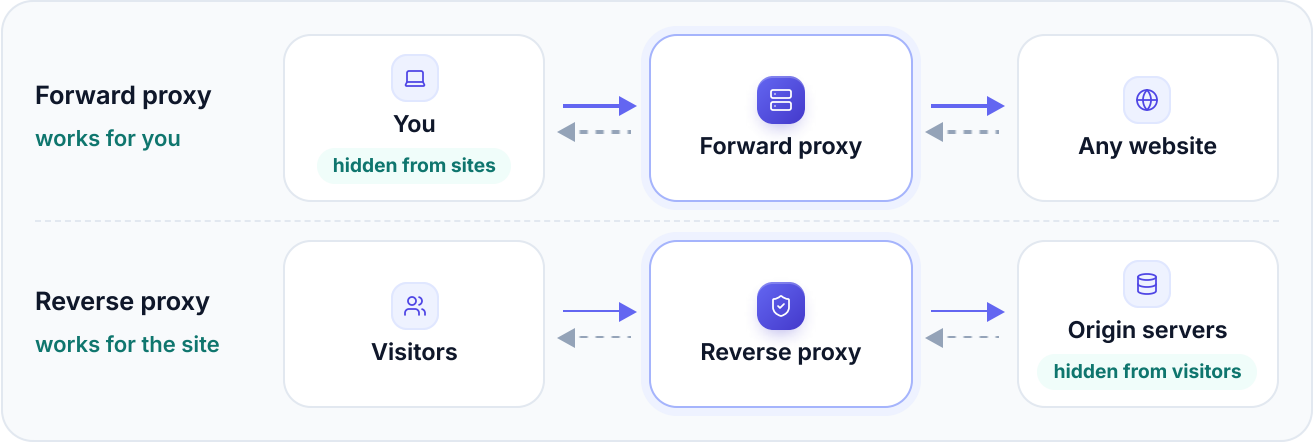


Figure 1.2 Same middle box, opposite loyalties. A forward proxy hides the client from the site; a reverse proxy hides the servers from visitors and inspects everyone who knocks.

Proxy, VPN or Tor? Same idea, different jobs

	Proxy	VPN	Tor
Carries	Traffic from apps you configure	Everything the device sends	Tor Browser traffic
Encryption	None of its own; HTTPS stays end-to-end	All traffic, up to the VPN server	Layered, across three relays
IP control	Type, country, city; rotate per request	Pick a server location	Random exit relay
Identities	Thousands in parallel	One per connection	A handful, slowly
How sites see it	Depends on type; residential looks like home users	Known VPN ranges, often challenged	Public exit list, often blocked
Best for	Data collection, geo testing, multi-account work	Privacy on untrusted networks, streaming	Strong personal anonymity

Scraping through a VPN? Stop.

You get one IP at a time, shared with thousands of other subscribers, on address ranges every anti-bot vendor already lists. A modest pool of datacenter proxies will outperform it for a fraction of the hassle.

1.3 What a Proxy Hides, and What It Doesn't

A proxy replaces your **network identity**. It does nothing for your **client identity**, and that second layer is where most people get caught. Keep the two apart in your head and half of all blocking problems explain themselves.

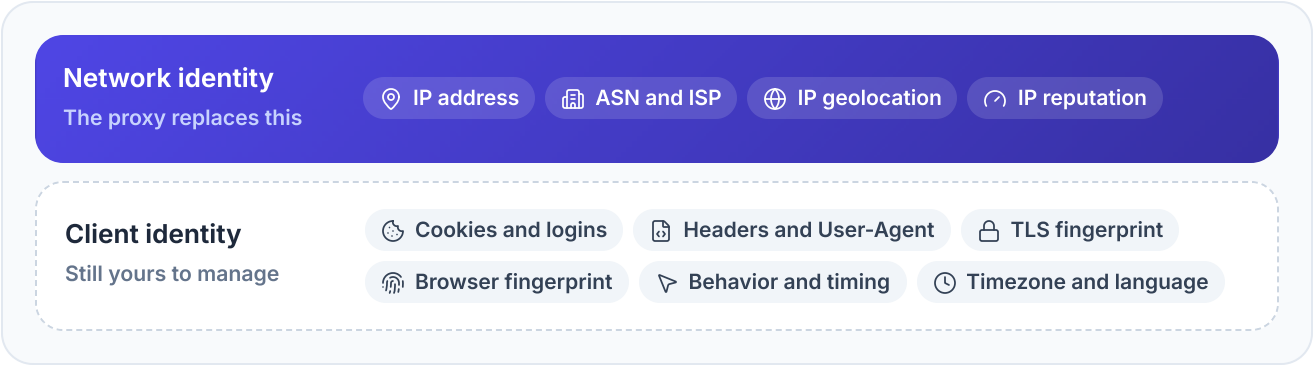


Figure 1.3 Two layers of identity. Swapping the top layer while the bottom one still says "Python script in Frankfurt" is how a perfectly good residential IP gets flagged.

Transparent, anonymous and elite proxies

Older proxies announced themselves with HTTP headers. The terms survive on provider sites, so it's worth knowing what they mean.

Level	What the website learns	Typical headers	Where you meet it
Transparent	Your real IP, plus the fact you're proxied	`X-Forwarded-For: 203.0.113.7`	Corporate and caching proxies
Anonymous	That a proxy is in use, but not your IP	`Via: 1.1 proxy`	Some free and legacy proxies
Elite	Nothing: it looks like a direct visit	None added	Every reputable commercial network

> ⚠ **Elite doesn't mean undetectable**
>
> Modern anti-bot systems barely look at proxy headers. They check who owns the IP, how your TLS handshake is built and how your browser behaves. Elite is the minimum you should accept, not a guarantee.

Leaks that skip the proxy entirely

Some traffic can bypass a proxy without you noticing. DNS lookups resolved on your own machine reveal the sites you visit, and browsers can expose your real IP through WebRTC. Chapter 10 shows how to test for both. In short: let the proxy resolve hostnames, and lock WebRTC down in any browser you automate.

1.4 The IP Address, Decoded

Within milliseconds of your first packet, a website knows more about your IP than most people expect. IP intelligence databases map every routable address to an owner, a network type, a location and a history.

Owner (the ASN)

Every IP belongs to an Autonomous System run by one organization: Amazon, Comcast, Vodafone. The ASN alone tells a site whether you're a server or a household.

Connection type

Hosting, residential, mobile or business. Datacenter ranges get the strictest treatment because real shoppers rarely browse from a cloud server.

Location

Country and region are usually right. City-level geolocation is an estimate, and the big databases often disagree with each other.

Reputation

Spam lists, abuse reports and the site's own logs. An IP that hammered the same site yesterday arrives with baggage today.

IPv4 or IPv6?

IPv4's 4.3 billion addresses ran out years ago, which is why clean IPv4 costs real money. IPv6 is so plentiful that providers hand out millions cheaply, but it only helps on sites that accept IPv6, and many anti-abuse systems treat a whole `/64` block (one customer's standard allocation) as a single visitor. Confirm your targets support it before you buy.

Subnets matter more than single IPs

Datacenter addresses come in tidy blocks. Ban one address in `192.0.2.0/24` and a site can ban all 256 with one rule, so ask for static IPs spread across many subnets and several ASNs. To see any IP the way websites do, paste it into ProxyHorizon's free IP Lookup.

KEY TAKEAWAYS

- A proxy swaps your network identity: IP, ASN, location and reputation. Nothing else.
- Forward proxies work for you. Reverse proxies like Cloudflare work for the site and host most anti-bot checks.
- VPNs and Tor solve personal privacy, not scale. For data collection, use proxies.
- Sites judge an IP by its owner, type, location and history within milliseconds of the first packet.

Go deeper: How Do Proxies Work? · What Is an ASN? · How Proxy IP Reputation Works

02



The Four Proxy Types

Every proxy you'll ever buy belongs to one of four families. They differ in the one thing that matters more than any feature list: whose IP address you borrow, and how much websites trust it.

IN THIS CHAPTER

2.1 The trust ladder

2.2 Datacenter proxies

2.3 Residential proxies

2.4 ISP proxies

2.5 Mobile proxies

2.6 Choosing your starting point

2.1 The Trust Ladder

Websites decide how much to trust a visitor long before they read a single header, and the biggest input is the IP's owner. An address registered to a cloud host says "server." One registered to a cable company or a mobile carrier says "person." Proxy types are really just rungs on that ladder of trust, and every rung up costs more.

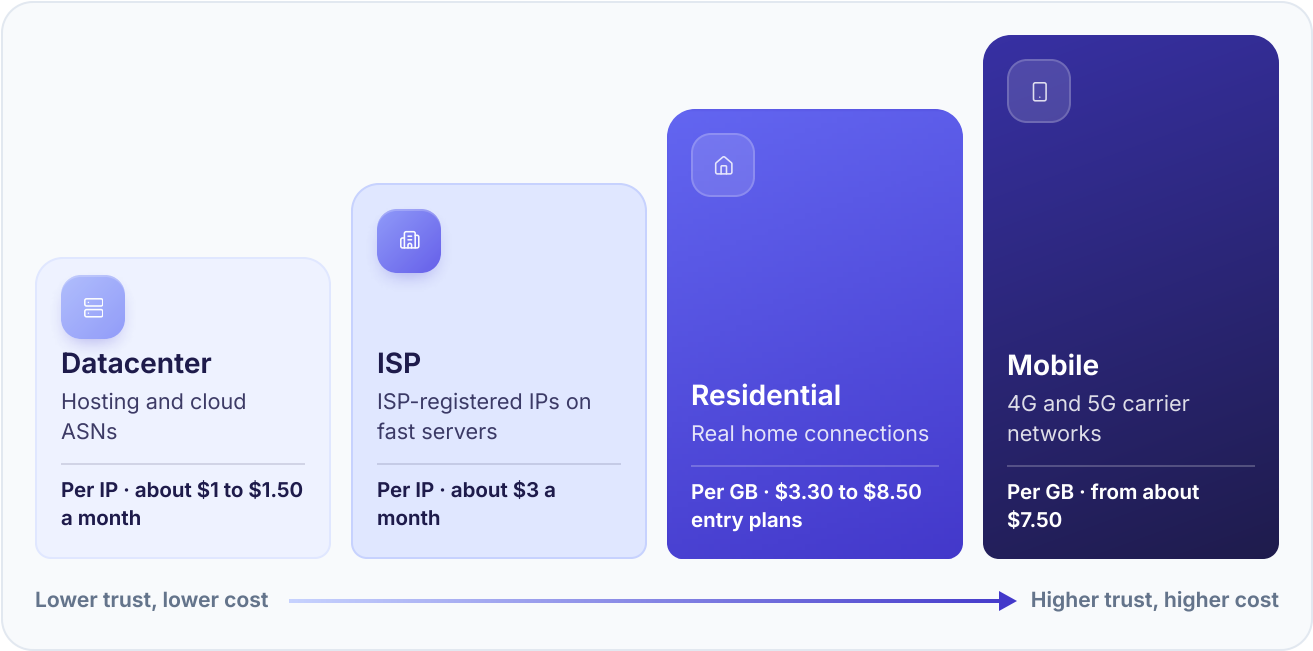


Figure 2.1 Each rung borrows an IP that websites trust more, and charges more for it. Entry prices are from the ProxyHorizon directory, October 2026; volume plans go lower.

	Datacenter	ISP	Residential	Mobile
Speed	●●●●●	●●●●●	●●●○○	●●○○○
Block resistance	●●○○○	●●●●○	●●●●○	●●●●●
Pool size	Large	Small	Huge	Large
IP behavior	Static or rotating	Static	Rotating or sticky	Rotating or sticky
Billed by	IP (or GB)	IP	GB	GB or port
Reach for it when	The target doesn't fight back	You need one stable identity	You need scale on protected sites	Nothing else gets through

The rungs blur at the edges

The dots are ProxyHorizon's relative editorial scores, not benchmarks, and a clean datacenter range can beat a tired residential pool on some targets. Treat the ladder as a starting assumption and let your success rate on real targets decide (Chapter 6 shows how to test).

2.2 Datacenter Proxies

Fast, cheap and easy to spot
IP addresses owned by hosting and cloud companies

Speed ●●●●●
Block resistance ●●○○○
Billing **Per IP, sometimes per GB**

Datacenter proxies run on servers in commercial data centers, using IP ranges registered to hosting providers. They're the original proxy and still the workhorse for anything that doesn't fight back. Connections are fast and stable because the servers sit on business-grade links, so latency mostly comes down to the distance between the proxy and the target.

You'll see them sold two ways. **Shared** IPs serve several customers at once, which makes them cheaper but means you inherit whatever the last user did. **Dedicated** (or private) IPs are yours alone. Some providers also sell rotating datacenter pools behind a single gateway, billed by bandwidth.

Strengths

- The fastest, most stable connections you can buy
- The lowest cost per request by a wide margin
- Per-IP plans often include unmetered bandwidth
- Easy to scale to thousands of static IPs

Weaknesses

- Hosting ASNs are flagged instantly by strict sites
- One bad neighbor can get a whole subnet banned
- Shared IPs carry other customers' history
- City-level targeting is limited or missing

Where datacenter proxies shine

Public APIs, documentation sites, bulk file downloads, uptime and SEO monitoring of your own properties, price checks on retailers without serious bot protection, and the first test against any new target. Steer clear of major marketplaces, social networks, search engines and ticketing sites, where hosting IPs are often challenged on the first request.

Always try datacenter first
Send a few hundred requests to a new target through datacenter IPs. If almost all of them come back with the content you expect, stay there. You'll get the same data for a fraction of residential prices, and you can always move up the ladder later.

Spread your subnets
A hundred IPs from one `/24` block are one IP as far as a strict site's rules are concerned. Ask your provider for addresses across many subnets, and rotate across them evenly.

2.3 Residential Proxies

Real home connections, real trust
IPs that consumer internet providers assigned to households

Speed ●●●○○
Block resistance ●●●●○
Billing **Per GB of traffic**

Residential proxies route your requests through real devices on home connections, like a laptop in Ohio or a desktop in Lyon, so the website sees an IP owned by a consumer ISP. Providers recruit those devices through apps that ask people to share idle bandwidth, usually for payment, rewards or an ad-free product.

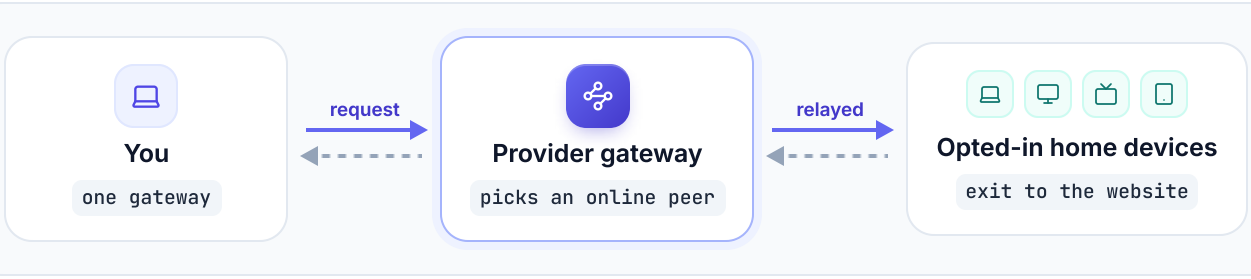


Figure 2.3 Your request exits through whichever opted-in device the gateway picks. Reputable networks reward those people and let them leave anytime.

Strengths

- Consumer ISP ownership passes ASN checks
- Huge pools: 30M to 155M+ IPs per network
- Country, state, city and often ISP targeting
- Built for rotation at scale

Weaknesses

- Per-GB billing punishes heavy pages
- Speed varies with each household's connection
- Sticky sessions end when a device goes offline
- Sourcing ethics vary a lot between providers

Best for: protected sites, search and e-commerce data, ad verification, and anything that needs to look like real people in specific places.

Images eat your gigabytes
Per-GB plans count every byte through the proxy, and a headless browser downloads every image, font and video it sees. Block what you don't need (Chapter 9) and residential gets far cheaper.

Ask where the IPs come from
A reputable provider can show you its consent flow and explain how peers are paid or rewarded. If a provider can't tell you where its residential IPs come from, walk away: more than one residential network has turned out to be a botnet of infected devices. Chapter 10 lists the questions to ask.

2.4 ISP Proxies

Residential trust at server speed

Static IPs registered to internet providers, hosted in data centers

Speed ●●●●●
Block resistance ●●●●○
Billing **Per IP, per month**

ISP proxies split the difference between the first two families. The IP ranges are registered to consumer or business internet providers, so websites see an ISP's ASN, but the addresses are announced from fast servers in data centers. You get residential-grade ownership with datacenter speed and uptime. You'll also see them sold as **static residential proxies**; in practice the two names describe the same product.

The defining feature is that the IP doesn't change. Keep paying and you keep the same address for weeks or months, which is exactly what logged-in work needs. An account that signs in from the same residential-looking IP every day looks like a person who never moved house.

Strengths

- ISP ownership with datacenter speed and uptime
- Static IPs you can keep for months
- Usually unmetered bandwidth on per-IP plans
- Ideal for long, logged-in sessions

Weaknesses

- Small pools compared with residential networks
- Mostly limited to the US, UK and major EU markets
- Costs more per IP than datacenter
- Advanced checks can spot server-like network traits

Where ISP proxies shine

Social media and marketplace accounts, e-commerce seller tools, dashboards behind a login, checkout and payment-flow testing, and any job where the same identity has to come back day after day. For high-volume, stateless scraping they're usually the wrong fit: a small static pool burns out faster than a rotating residential one.

Four questions before you buy ISP proxies

- Are the IPs dedicated to me, or shared with other customers?
- How many subnets and ASNs will my IPs span?
- Which ISPs are the IPs actually registered to?
- What's the replacement policy if an IP gets flagged?

2.5 Mobile Proxies

The hardest IPs to block

Traffic exits through real 4G and 5G carrier connections

Speed ●●○○○
Block resistance ●●●●●
Billing **Per GB, or per dedicated port**

Mobile proxies send your traffic out through SIM-based connections on carrier networks, either from devices that opted into a peer network or from racks of dedicated modems. They're the hardest proxies to block, and the reason is how carriers run their networks.

Carriers don't have enough IPv4 addresses for every phone, so they use **carrier-grade NAT** (CGNAT). Hundreds or even thousands of subscribers share one public IP at the same time, and the mapping reshuffles constantly. A website that blocks a mobile IP risks locking out real paying customers, so most sites treat mobile traffic gently.

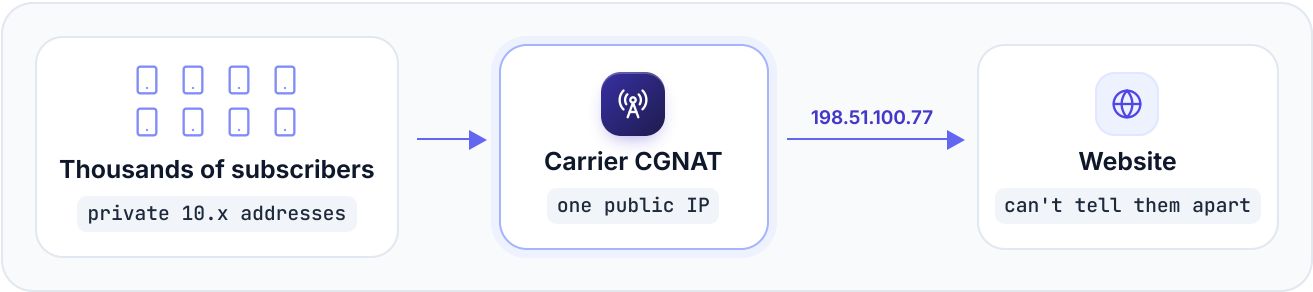


Figure 2.5 Behind carrier-grade NAT, one public IP stands for a crowd of real people. Blocking it would block all of them, which is why mobile IPs earn the most trust.

Strengths

- The highest trust of any proxy type
- Native fit for mobile-first apps and platforms
- Natural rotation as carriers reassign IPs
- Carrier-level targeting on most networks

Weaknesses

- The most expensive per GB or per port
- Slower, with more variable latency
- Thinner pools in some countries
- Overkill for anything a residential IP can handle

Two ways to buy them. Rotating pools bill per GB and hand you a fresh carrier IP per request or session. Dedicated ports give you one modem for a monthly fee, usually with unmetered traffic and an API call that forces a new IP.

Match the device to the network

A mobile IP paired with desktop Chrome on Windows tells an inconsistent story. Use a mobile user agent, viewport and touch support, or an antidetect browser profile built for mobile, so every signal agrees.

2.6 Choosing Your Starting Point

Start at the bottom of the ladder and climb only when your data says so. Three questions settle it for most teams.

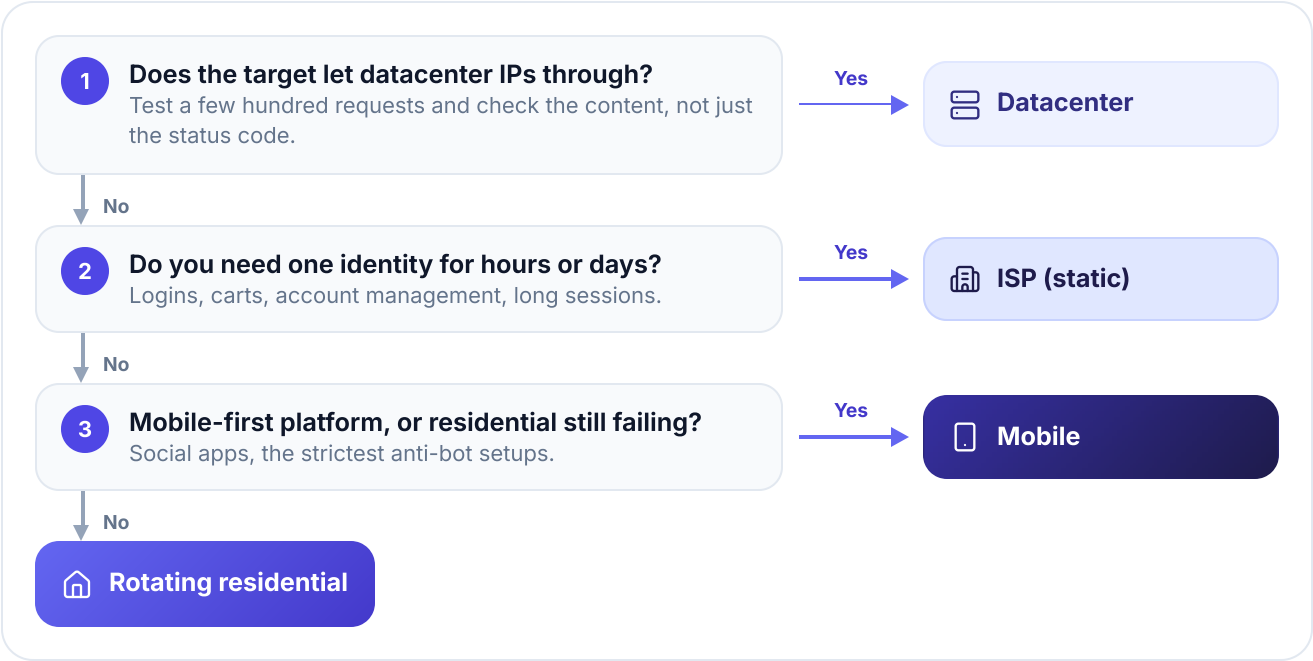


Figure 2.6 Climb the ladder one rung at a time. Most scraping workloads end at rotating residential; most account workloads end at ISP or mobile.

Two options outside the ladder

IPv6 proxies

Enormous pools for very little money, but only useful on sites that accept IPv6, and many of those rate-limit by whole `/64` or `/48` blocks. Test before you commit.

Unblockers and scraping APIs

Managed services that handle proxies, browser fingerprints, retries and CAPTCHAs for you, billed per successful request. Pricier per page, often cheaper once you count engineering time.

KEY TAKEAWAYS

- The IP's owner drives trust: hosting company, ISP, household or mobile carrier.
- Datacenter is fastest and cheapest; try it first on every new target.
- Residential is the default for scraping protected sites; ISP is the default for accounts.
- Mobile buys the most trust at the highest price. Use it where nothing else works.

Go deeper: What Is a Residential Proxy? · Best ISP Proxy Providers · Best Mobile Proxy Providers

03



Protocols and Authentication

Before a proxy can carry a single request, your client and the proxy have to agree on two things: how they'll talk, and who you are. Get both right and most mysterious connection errors simply never happen.

IN THIS CHAPTER

3.1 HTTP, HTTPS and SOCKS5

3.2 Authentication methods

3.3 Anatomy of a proxy endpoint

3.1 HTTP, HTTPS and SOCKS5

HTTP proxies understand web traffic; **SOCKS proxies** sit a layer lower and forward any connection you hand them. Most providers offer both on one gateway, on different ports. For a plain `http://` URL, an HTTP proxy reads your whole request and can even change your headers. For `https://` URLs, which is nearly everything today, your client sends a `CONNECT` request instead: the proxy opens a raw tunnel to the site and steps aside.

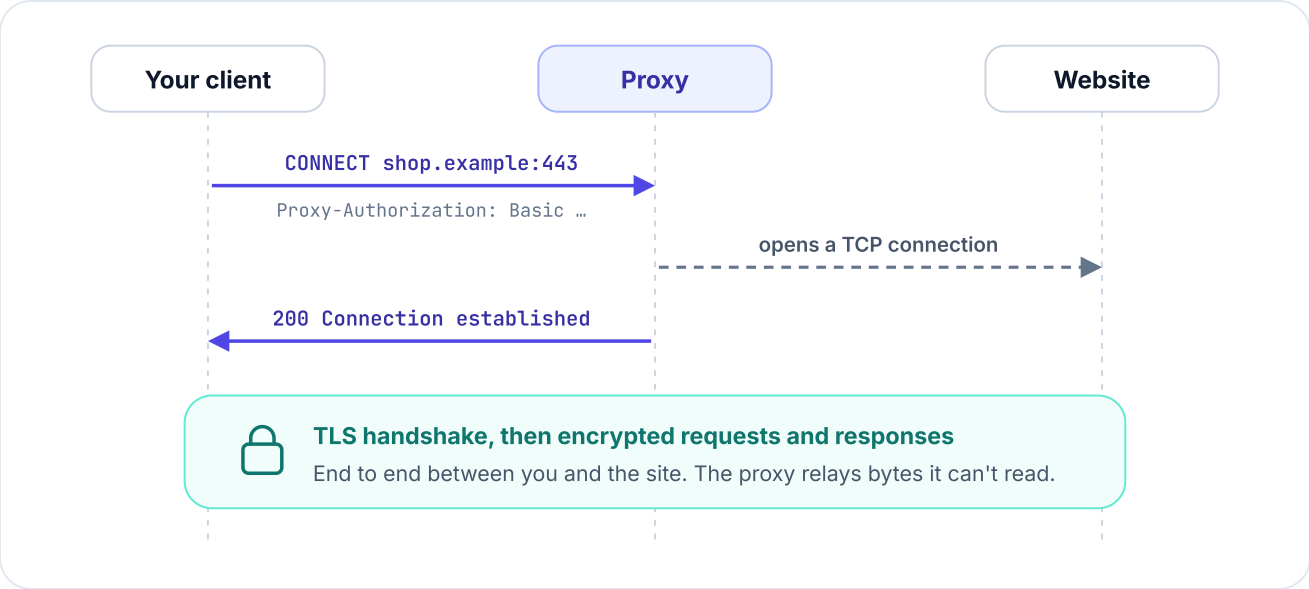


Figure 3.1 An HTTPS request through an HTTP proxy. The proxy learns the hostname, port, timing and byte counts, but never sees URLs, headers, cookies or page content.

SOCKS5 forwards any TCP stream, and UDP where the provider supports it, without parsing what's inside. Its most useful trick: it can let the *proxy* resolve hostnames, so your DNS lookups never leave your machine and the site's IP is picked from the exit's location.

	HTTP / HTTPS proxy	SOCKS5 proxy
Works at	Application layer (HTTP)	Session layer, below HTTP
Carries	Web traffic: HTTP plus HTTPS via CONNECT	Any TCP connection; UDP where supported
DNS resolution	Done by the proxy	Your choice: local or remote (`socks5h`)
Choose it for	Scraping, browsers, APIs: the default	Non-HTTP tools, UDP, strict DNS control

Mind the h in socks5h

In cURL and Python's requests, `socks5://` resolves hostnames on your machine and `socks5h://` lets the proxy do it. Use the h unless you have a reason not to.

3.2 Authentication Methods

Commercial proxies are paid services, so every request has to prove it belongs to your account. Providers offer two ways to do that, and most let you use both.

Username and password

Your client sends credentials with each connection, in a `Proxy-Authorization` header for HTTP proxies or during the SOCKS5 handshake. It works from anywhere: your laptop, a CI runner, a fleet of cloud workers whose IPs change daily.

WATCH FOR

HTTP proxy credentials are Base64-encoded, not encrypted, on the hop between you and the proxy. Keep that hop on networks you trust.

IP allowlisting

You register your server's public IP in the provider dashboard, and any connection from that address is accepted without credentials. Nothing secret lives in your code or config.

WATCH FOR

Anyone behind the same IP can spend your plan, which matters in shared offices and on NAT gateways. It also breaks the moment your server's IP changes.

	Username and password	IP allowlisting
Works from	Anywhere	Only registered IPs
Secrets in code	Yes, keep them in a vault or env vars	None
Per-request targeting	Yes, through username flags	Usually through ports or the dashboard
Best for	Laptops, CI, autoscaling workers	Fixed servers, legacy tools without auth support

When authentication fails

A `407 Proxy Authentication Required` response comes from the proxy, never from the website. It means wrong credentials, an unregistered IP, an expired plan, or special characters in your password that broke the URL. Section 3.3 covers that last one, because it fools experienced engineers every week.

Credential hygiene that takes five minutes

- [ ] Keep proxy credentials in environment variables or a secrets manager, never in Git.
- [ ] Create a **sub-user** per project or client, with its own traffic limit, so one runaway job can't drain the account.
- [ ] Rotate passwords when someone leaves the team or a laptop goes missing.
- [ ] Prefer allowlisting for long-lived production servers with fixed IPs.
- [ ] Review the usage dashboard weekly. A sudden spike is often the first sign of a leaked key.

3.3 Anatomy of a Proxy Endpoint

Most providers encode targeting and session options in the username, so one host and port can serve a rotating US request, a sticky session in Dallas or a mobile exit in Berlin. The syntax varies; the parts don't.



1. **Scheme.** How your client talks to the proxy: `http`, `https` or `socks5h`.
2. **Account.** Your username, or a sub-user for one project.
3. **Targeting.** Country, state, city or ASN flags.
4. **Session.** A sticky-session ID and how many minutes it should hold.
5. **Password.** Encode `@ : / #` as `%40 %3A %2F %23`, or expect a baffling 407.
6. **Gateway.** The provider's entry point, not your exit IP.
7. **Port.** Often picks the protocol or the session type.

Figure 3.3 One endpoint, seven parts. Change the flags in part 3 or 4 and the same gateway hands you a different country or a new sticky IP. Flag names vary by provider, so copy them from your provider's docs.

The five formats you'll run into

Format	Example	Common in
host:port:user:pass	`gate.example.net:7000:acme:S3cr3t`	Dashboard exports, bulk lists
user:pass@host:port	`acme:S3cr3t@gate.example.net:7000`	Browser extensions, many tools
URL	`http://acme:S3cr3t@gate.example.net:7000`	cURL, Python, Node, env vars
SOCKS URL	`socks5h://acme:S3cr3t@gate.example.net:7001`	cURL, requests with PySocks
host:port only	`gate.example.net:7000`	Allowlisted servers

KEY TAKEAWAYS

- HTTP proxies are the default; HTTPS traffic travels through a CONNECT tunnel the proxy can't read.
- Choose SOCKS5 for non-HTTP traffic or strict DNS control, and write it as `socks5h`.
- A 407 always comes from the proxy: check credentials, allowlist, plan status and encoding.
- Targeting and sessions usually live in the username, so one gateway serves every use case.

Free tool: Proxy Format Converter · **Read:** HTTP vs HTTPS vs SOCKS5 Proxies

04



Rotation, Sessions and Geo-Targeting

Rotation is what turns one proxy into a crowd. Done well, it spreads your traffic so thinly that no site notices. Done badly, it logs you out, breaks checkouts and burns through good IPs. Here's how to choose.

IN THIS CHAPTER

4.1 How backconnect gateways work

4.2 Rotating vs sticky sessions

4.3 Geo-targeting without surprises

4.1 How Backconnect Gateways Work

There are two ways to get a lot of IPs. You can buy a **static list**, one `host:port` per IP, and rotate through it yourself. Or you can connect to a **backconnect gateway**: one hostname that forwards each connection to an exit IP chosen from the provider's pool. Residential and mobile networks are almost always sold the second way, because their exits go online and offline all day and no list could keep up.

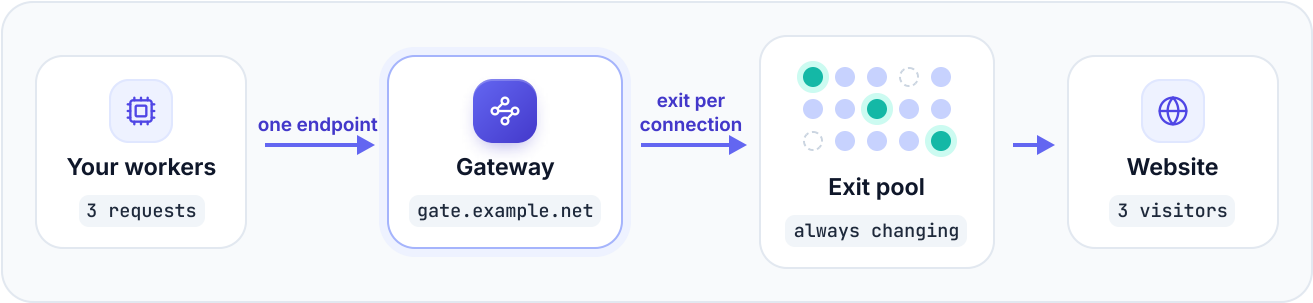


Figure 4.1 You talk to one address; the gateway fans your connections out across the pool. Teal exits carry your three requests right now, grey ones just went offline, and the gateway quietly routes around them.

	Static proxy list	Backconnect gateway
You connect to	A separate endpoint for every IP	One endpoint for the whole pool
Who rotates	You, in your code	The provider, per connection or session
Pool size	Exactly what you bought	Thousands to millions of exits
Dead IPs	You detect and remove them	The gateway skips them for you
Billing	Usually per IP	Usually per GB
Best for	Datacenter and ISP proxies, account work	Residential and mobile, scraping at scale

> ⚠ **Keep-alive can quietly pin your IP**
>
> Most gateways choose the exit when a connection opens. HTTP keep-alive reuses that connection, so ten "rotating" requests from one client session can all leave from the same IP. Need a fresh IP every time? Open a new connection per request. Want stickiness? Ask for it with a session ID instead of relying on keep-alive.

If you run a static list yourself, rotate it with a little discipline: spread requests evenly, give each IP a cool-down after it hits a target, pull an IP out of rotation after a few consecutive failures, and re-test it later instead of deleting it on the spot.

4.2 Rotating vs Sticky Sessions

A gateway can give you a new IP on every connection, or hold the same IP for a while. Holding it is called a **sticky session**, and you request one by adding a session ID to your username. Every request that carries the same ID leaves from the same exit until the session's time runs out or the device behind it disconnects.

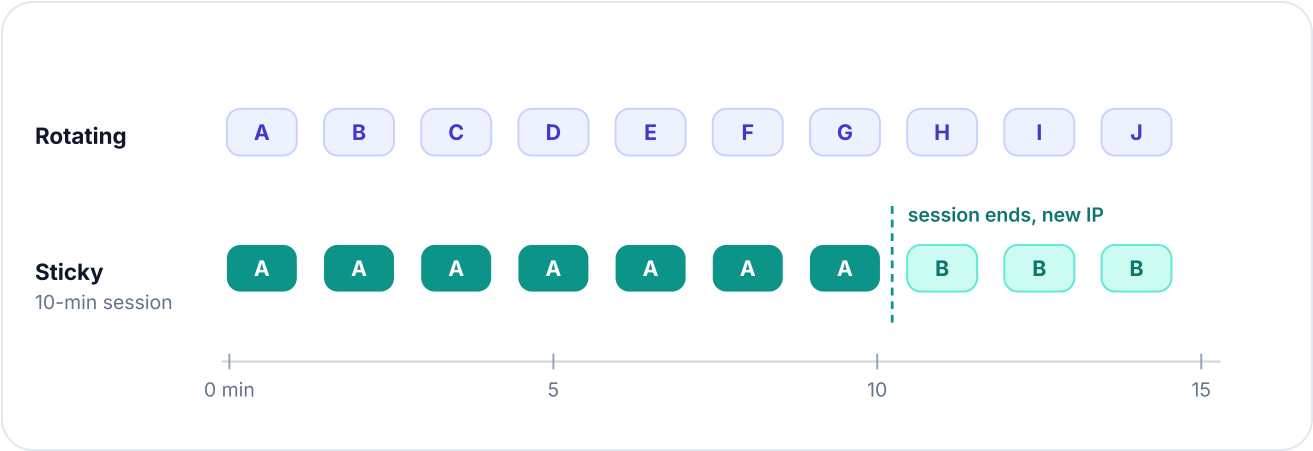


Figure 4.2 Ten requests over 15 minutes. Rotation gives every request a new IP (A to J). A 10-minute sticky session holds IP A for the first seven, then hands you IP B.

Task	Session strategy	Why
Search result tracking	Rotate per request	Each query stands alone; spread the load
Product catalog crawl	Rotate, or 1 to 5 min sticky per category	Pagination is often tied to session cookies
Location-based prices	Sticky for the whole flow	The store or ZIP you pick lives in the session
Logged-in account work	Long sticky, or a static ISP IP	IP changes mid-session trigger security checks
Social multi-account	One dedicated static or mobile IP per account	Each identity needs a consistent home
Checkout and QA flows	Sticky for longer than the flow	Payment risk systems watch for IP changes

Design for the session ending early

On residential networks a sticky session ends the moment the device behind it disconnects, whatever lifetime you asked for. Check your exit IP between critical steps, treat a change as a signal to re-validate the session, and make every step safe to retry.

4.3 Geo-Targeting Without Surprises

Most providers let you target exits by country, and many go down to state, city, ZIP code or a specific ISP. Precision is useful, and it isn't free: every level you narrow shrinks the pool you're drawing from.

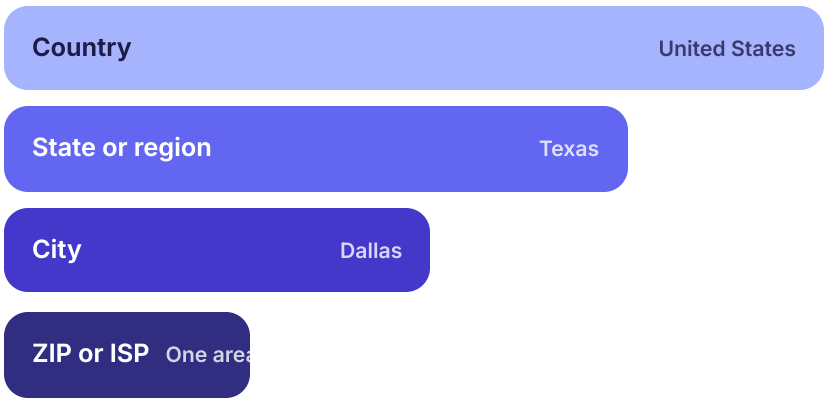


Figure 4.3 Each step down the funnel is more precise and draws on fewer IPs. Use the coarsest level that still returns the right data.

Match everything to the exit's location

An IP in Dallas with a German browser, a Tokyo timezone and a euro currency cookie tells three conflicting stories. When you target a location, set `Accept-Language`, the timezone and any locale or store cookies to match it.

Read pool numbers with care

"100M+ IPs" usually counts every address a network has seen over a month or longer, not the IPs online right now in your city. Ask for country-level numbers, then test: send 50 requests through a city-targeted session and check where each exit lands with the free IP Lookup. Geolocation databases disagree at city level, so expect some spread.

KEY TAKEAWAYS

- Gateways rotate exits for you; static lists put rotation, health checks and cool-downs in your hands.
- Rotate for stateless pages. Stick for logins, carts, checkouts and anything multi-step.
- Keep-alive can pin an IP, and sticky sessions can end early. Design for both.
- Target as coarsely as your data allows, and keep language, timezone and cookies in line with the exit.

Go deeper: What Are Rotating Proxies? · **Free tool:** Proxy Rotator Script Generator

05



Use Cases and Playbooks

Theory only gets you so far. This chapter turns the first four into setups you can copy: which proxy type, how to rotate, where to target and what to watch, for the jobs teams use proxies for most.

IN THIS CHAPTER

5.1 Match the job to the proxy

5.2 Data collection playbooks

5.3 Commerce and advertising playbooks

5.4 Accounts and AI agents

5.1 Match the Job to the Proxy

Almost every proxy project is a variation on one of these ten jobs. Find yours, start with the setup in its row, and adjust from there once you have real success rates.

Use case	Proxy type	Rotation	Targeting	Watch for
Public web scraping	Datacenter first, then residential	Per request	Country	Rate limits, soft blocks
Search and rank tracking	Residential, or a SERP API	Per request	Country, city, language	CAPTCHAs, personalization
Price and stock monitoring	Datacenter, then residential	Per request; sticky for store flows	Country or ZIP	Location cookies, price tests
Ad verification	Residential and mobile	Per request	City, carrier	Cloaking for suspect IPs
Social media management	ISP or mobile, one per account	Static	The account's home country	Fingerprint consistency
Market and review research	Residential	Per request	Country	Pagination, JS rendering
Travel fare aggregation	Residential	Sticky per search	Point-of-sale country	Heavy anti-bot, session pricing
QA and localization	ISP or residential	Sticky	City	Locale, currency, timezone
AI agents	Residential or ISP, real browser	Sticky per task	Country	Cost per task, site terms
Brand protection	Residential and mobile	Per request	Wherever victims are	Isolate risky pages

One project, several proxy types

Mature teams don't pick a single type for a whole project. They route each target, or even each page type, to the cheapest rung that works: sitemaps and category pages over datacenter, protected product pages over residential, account actions over ISP. The savings compound quickly.

Every playbook assumes a legitimate purpose

Collect public data, respect each platform's terms, keep away from personal data you don't need, and never use proxies to commit fraud or evade bans you've earned. Chapter 10 covers the legal and ethical ground in more detail.

5.2 Data Collection Playbooks

Web scraping at scale

Product catalogs, listings, directories and other public pages

PROXY TYPE	ROTATION	TARGETING
Datacenter first, residential where blocked	Per request	Country

1. **Test every target on datacenter IPs first.** Route only the stubborn ones to residential.
2. **Look for the site's own JSON endpoints.** They're lighter and more stable than HTML; render pages only when you must.
3. **Cap concurrency per domain.** Back off on 429s and honor `Retry-After`.
4. **Validate content before you store it.** Check a selector or schema, and retry failures through a fresh IP.

CLASSIC PITFALL

Treating HTTP 200 as success. Soft blocks return a CAPTCHA page with a 200 status.

WATCH THIS NUMBER

Cost per valid record, not cost per request.

Search and rank tracking

Keyword rankings, AI answers, local packs and SERP features

PROXY TYPE	ROTATION	TARGETING
Residential, or a dedicated SERP API	Per request	Country, city, language

1. **Pin the location twice.** Use a geo-targeted exit and the search engine's own language and region parameters.
2. **Start every query with a clean cookie jar.** Personalization quietly skews rankings.
3. **Spread queries across the day.** Bursts of identical requests are what trigger CAPTCHAs.
4. **Parse defensively.** Result layouts change often as engines add AI summaries, ads and local packs.

CLASSIC PITFALL

Checking rankings from your office IP. The results are personalized and localized to you.

WATCH THIS NUMBER

CAPTCHA rate per 1,000 queries.

5.3 Commerce and Advertising Playbooks

Price and stock monitoring

Competitor pricing, availability, promotions and marketplace listings

PROXY TYPE	ROTATION	TARGETING
Datacenter for small shops, residential for marketplaces	Sticky for store selection	ZIP or country

1. **Find out how the site sets location.** It might use your IP, a cookie or a store picker, and often all three.
2. **Replay the store or ZIP choice inside a sticky session,** then crawl products in that same session.
3. **Schedule by volatility.** Check top products hourly and the long tail daily.
4. **Confirm before you alert.** Sample a changed price from two or three IPs to rule out an A/B price test.

CLASSIC PITFALL

Firing an alert on a price that only one bucket of a pricing experiment ever saw.

WATCH THIS NUMBER

Freshness: minutes since the last valid price per product.

Ad verification and brand safety

Placements, creatives, landing pages and ad fraud

PROXY TYPE	ROTATION	TARGETING
Residential and mobile	Per request	City and carrier

1. **Load ads as real users would,** in every geo and device type your campaign targets.
2. **Capture the whole chain:** creative, redirects and final landing page, with screenshots as evidence.
3. **Catch cloaking.** Compare what clean residential IPs see with what datacenter IPs see; bad actors show reviewers something different.
4. **Rotate devices too.** Mobile placements and in-app ads need mobile IPs and mobile fingerprints.

CLASSIC PITFALL

Verifying from datacenter IPs. Fraudulent and legitimate ad systems both treat them differently.

WATCH THIS NUMBER

Coverage: geo and device combinations checked per campaign.

5.4 Accounts and AI Agents

Social media and multi-account management

Agencies, brand pages, community and marketplace accounts

PROXY TYPE	ROTATION	TARGETING
ISP or mobile, one dedicated IP per account	Static	The account's home country

1. **One account, one profile, one proxy.** Pair every account with its own antidetect browser profile and IP, and never share them.
2. **Keep the story consistent** for the account's whole life: same IP, timezone, language and device.
3. **Warm new accounts up slowly.** Real people don't post fifty times on their first day.
4. **Check each platform's rules.** Some allow multiple accounts for agencies and brands; others ban the practice outright.

CLASSIC PITFALL

Rotating IPs on a logged-in account. To the platform it looks like an account takeover.

WATCH THIS NUMBER

Verification prompts and restrictions per account per month.

AI agents and LLM data pipelines

Browsing agents, research assistants and training-data collection

PROXY TYPE	ROTATION	TARGETING
Residential or ISP, with a real browser	Sticky per task	Country

1. **Give each task one sticky session and one browser profile,** so a multi-step task never changes identity halfway.
2. **Budget every run.** Cap pages, bandwidth and time per task so a confused agent can't spend your month in an hour.
3. **Prefer APIs, sitemaps and feeds** before you browse. They're cheaper, faster and more polite.
4. **Respect robots.txt and rate limits,** and identify your agent wherever a site offers a verified-bot program.

CLASSIC PITFALL

An agent stuck in a CAPTCHA loop or an endless scroll, burning gigabytes with nothing to show.

WATCH THIS NUMBER

Cost per completed task.

06



Choosing a Provider

More than a hundred providers all claim the biggest pool, the best success rate and the fairest price. This chapter gives you a way to check those claims yourself, and the math to see what a plan will really cost.

IN THIS CHAPTER

6.1 A 10-point evaluation framework

6.2 Reading a pricing page

6.3 Working out your true cost

6.4 Running a fair trial

6.5 Nine networks worth testing

6.1 A 10-Point Evaluation Framework

Marketing pages are built to answer the questions providers want asked. These ten are the ones that decide whether a network works for *you*. Score each provider on them during a trial, and weight the first three most heavily.

1. Success rate on your targets

The only number that really matters, measured on the sites you care about, by content rather than status code.

2. Pool depth where you need it

Country and city availability right now, not a lifetime headline figure.

3. Speed and latency

Median and 95th-percentile response times, plus how long connections take to open.

4. Targeting options

Country, state, city, ZIP, ASN or carrier, and how precise each one really is.

5. Session control

Per-request rotation, sticky durations, and how often sessions end before you ask.

6. Protocol support

HTTP, HTTPS and SOCKS5 on every product, plus UDP if your tools need it.

7. Pricing model

Per GB or per IP, minimum commitments, traffic expiry and overage rates.

8. Ethics and compliance

Consent-based sourcing, customer verification (KYC) and a clear acceptable-use policy.

9. Tooling

A usage dashboard, an API, sub-users, IP allowlisting and integrations with your stack.

10. Support and trial

A trial or refund window, real technical support and documentation you can follow.

Turn the list into a scorecard

Rate each provider 1 to 5 on every point, multiply the first three by two, and add them up. The checklist in the appendix is ready to print for exactly this.

6.2 Reading a Pricing Page

Proxy pricing looks chaotic until you notice it comes in just four shapes. Each one wins for a particular kind of workload, and each hides a different trap.

Model	You pay for	Typical for	Wins when	Watch for
Per GB	Bandwidth through the proxy	Residential, mobile	Pages are light or traffic is bursty	Heavy pages; traffic that expires monthly
Per IP	A monthly fee per address	Datacenter, ISP	Pages are heavy or sessions are long	Thread caps; replacement rules
Per port or time	Connections or days of access	Mobile ports, unlimited plans	You push lots of data at steady concurrency	Unpublished thread or speed caps
Per success	Successful requests only	Scraping APIs, unblockers	Targets are hard or the team is small	Higher unit price, less control

What entry plans cost in 2026

$3.30–8.50
per GB, residential entry plans

$7.50+
per GB, mobile entry plans

~$3
per IP per month, ISP

$1.20–1.40
per dedicated datacenter IP

Entry-level prices across the nine providers in section 6.5, from the ProxyHorizon directory in October 2026. Volume tiers, shared datacenter plans and promotions go lower; check proxyhorizon.com/proxies for live numbers.

The costs that hide in the small print

- **Traffic expiry.** Many monthly per-GB plans wipe unused gigabytes at renewal. If your usage is irregular, look for non-expiring traffic or pay-as-you-go.
- **Failed requests still cost bandwidth.** On per-GB plans you pay for blocked responses and retries too, which is why success rate drives real cost.
- **Minimums and commitments.** The headline per-GB price often applies only to a large monthly commitment.
- **Premium features.** Some providers charge extra for city-level targeting, dedicated pools or longer sticky sessions.
- **Overage rates.** Going over your plan can cost far more per GB than the plan itself. Set usage alerts on day one.

6.3 Working Out Your True Cost

Two formulas tell you almost everything about what a proxy plan will cost and whether it's worth it. Keep them in a spreadsheet next to your trial results.

MONTHLY BANDWIDTH

```
requests × average response size × (1 + retry rate)
```

COST PER SUCCESSFUL REQUEST

```
monthly proxy spend ÷ valid responses
```

Worked example: one million product pages a month

Say you track a million product pages a month through residential proxies at $4 per GB, with a 10% retry rate. The only thing that changes between these two setups is what you download.

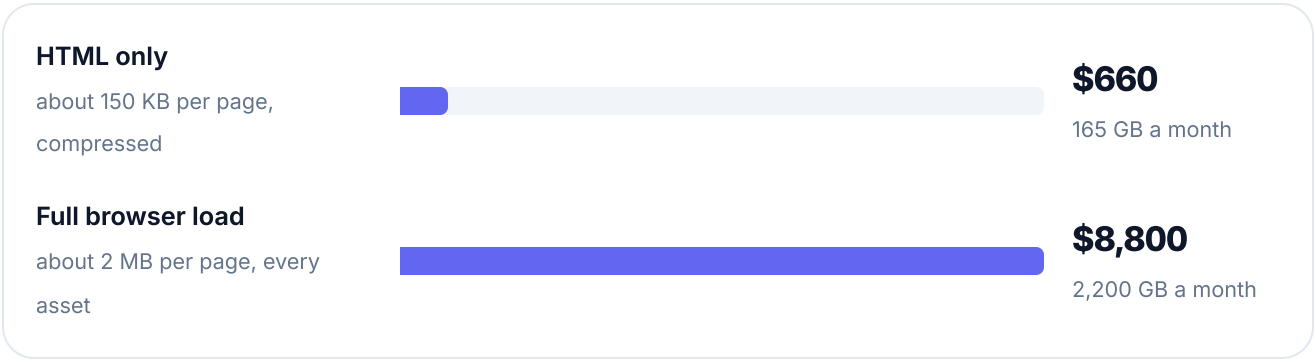


Figure 6.3 Same pages, same data, about 13 times the bill. Page sizes are illustrative assumptions; measure your own targets before you budget.

Cheap per GB is not cheap per result

Success rate matters as much as price. Compare two providers on a target where each attempt moves about 0.2 MB, assuming blocked attempts cost as much bandwidth as good ones:

	Price per GB	Success rate	Cost per 1,000 good pages
Provider A	$3.00	50%	$1.20
Provider B	$5.00	95%	**$1.05**

The "expensive" provider comes out about 12% cheaper per usable page, and that's before you count the extra time, retries and engineering a 50% success rate costs you.

Budget in cost per result

Report proxy spend to your team as cost per valid record or cost per completed task. It's the only number that lets you compare proxy types, providers and scraping APIs on equal terms.

6.4 Running a Fair Trial

Benchmarks published by providers, and frankly by anyone else, are measured on someone else's targets. A two-day trial on your own targets beats them all. Most providers offer a trial, a small pay-as-you-go plan or a refund window, so use it.

1. **Pick three to five representative targets.**
   Include your hardest one, not just the easy ones.
2. **Write one test script and run it everywhere.**
   Same targets, same concurrency, same time window for every provider.
3. **Validate content, not status codes.**
   Count a request as successful only when the data you need is actually in the response.
4. **Record the numbers that matter:**
   success rate, median and 95th-percentile latency, megabytes per success and geo accuracy.
5. **Stress a sticky session.**
   Ask for 30 minutes and measure how long the IP actually holds.
6. **Ask support one real technical question,**
   and note how long a useful answer takes.
7. **Compute cost per successful request**
   for each provider at your expected volume, then decide.

Red flags that end the trial early

- No clear answer on where residential IPs come from
- No trial, refund window or pay-as-you-go option
- No usage dashboard or per-request statistics
- "Unlimited" plans with unpublished thread or speed caps
- Residential IPs that geolocate to data centers
- Prices available only through a sales call at small volumes

KEY TAKEAWAYS

- Judge providers on your targets: success rate, pool depth in your locations and latency come first.
- Pricing comes in four shapes. Match the shape to your workload and read the small print on expiry and overage.
- Budget in cost per successful request. The cheapest gigabyte is rarely the cheapest result.

Free tool: Proxy Checker · **Read:** How to Test Proxy Speed, Latency and Success Rate

6.5 Nine Networks Worth Testing

A neutral starting shortlist from the ProxyHorizon directory, in alphabetical order. Every network here is a credible choice; which one is right depends on your targets, locations and budget, so test two or three of them with the method from section 6.4.

Bright Data

Enterprise web data platform: proxies plus scraper APIs and ready-made datasets.

Residential · ISP · Datacenter · Mobile

RESIDENTIAL POOL	COUNTRIES
72M+	195

Decodo

Formerly Smartproxy. An easy dashboard and broad coverage across every proxy type.

Residential · ISP · Datacenter · Mobile

RESIDENTIAL POOL	COUNTRIES
115M+	195

Geonode

Developer-friendly residential proxies with unlimited-bandwidth plans.

Residential

RESIDENTIAL POOL	COUNTRIES
30M+	190

IPRoyal

Residential traffic that never expires, a good fit for irregular workloads.

Residential · ISP · Datacenter · Mobile

RESIDENTIAL POOL	COUNTRIES
32M+	195

NodeMaven

Quality-filtered residential and mobile IPs, known for long sticky sessions.

Residential · Mobile

RESIDENTIAL POOL	COUNTRIES
30M+	195

Oxylabs

Premium enterprise network built for large-scale scraping, plus scraper APIs.

Residential · ISP · Datacenter · Mobile

RESIDENTIAL POOL	COUNTRIES
100M+	195

PyProxy

Hong Kong-based network with SOCKS5 per-IP pricing and an unlimited daily plan.

Residential · ISP · Datacenter

RESIDENTIAL POOL	COUNTRIES
90M+	190

SOAX

Clean pools with granular targeting down to city and ISP level.

Residential · Mobile

RESIDENTIAL POOL	COUNTRIES
155M+	195

Webshare

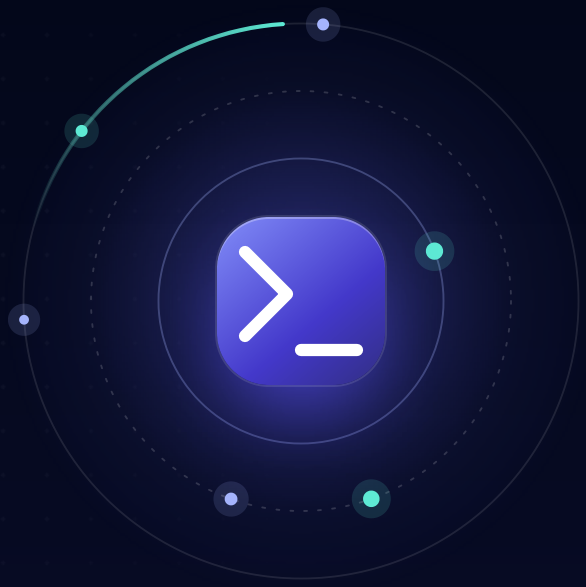
Affordable, self-serve datacenter proxies with highly customizable plans.

Datacenter · Residential

RESIDENTIAL POOL	DATACENTER POOL
80M+	400K+

Product types, pool sizes and coverage as listed in the ProxyHorizon directory, October 2026. Listing order is alphabetical and implies no ranking. Compare any two side by side at proxyhorizon.com/compare.

07



Setup and Integration

Every tool speaks proxy a little differently. These are the patterns for the ones you're most likely to use, ready to paste into a project and adapt, followed by the checks to run before you point any of it at production.

IN THIS CHAPTER

7.1 Your first request in 60 seconds
7.2 Python
7.3 Node.js and Puppeteer
7.4 Playwright, Scrapy and antidetect browsers
7.5 Verify before you scale

7.1 Your First Request in 60 Seconds

Before you touch your codebase, prove the proxy works from the command line. cURL ships with macOS, Linux and Windows, and it shows you exactly what's happening. Replace `USER`, `PASS` and the gateway with the values from your provider's dashboard.

Terminal · cURL

```
# 1. Your real IP, for comparison
curl -s "https://api.ipify.org?format=json"

# 2. The same request through an HTTP proxy
curl -s -x "http://USER:PASS@gate.example.net:7000" "https://api.ipify.org?format=json"

# 3. SOCKS5, letting the proxy resolve DNS
curl -s -x "socks5h://USER:PASS@gate.example.net:7001" "https://api.ipify.org?format=json"

# 4. Where the time goes: proxy connect, TLS handshake, total
curl -s -o /dev/null -x "http://USER:PASS@gate.example.net:7000" \
  -w "connect %{time_connect}s  tls %{time_appconnect}s  total %{time_total}s\n" \
  "https://example.com"
```

If request 2 prints a different IP from request 1, you're in business. If it prints an error, run it again with `-v`: the verbose output shows the `CONNECT` line, the proxy's response and, on failure, whether the problem is reaching the proxy or authenticating with it.

Set it once with environment variables

Most command-line tools, Python's requests and many SDKs pick up proxy settings from the environment, which keeps credentials out of your code.

Shell · environment

```
export HTTPS_PROXY="http://USER:PASS@gate.example.net:7000"
export HTTP_PROXY="$HTTPS_PROXY"
export NO_PROXY="localhost,127.0.0.1,.internal.example"   # never proxy these
```

⚠ **Two traps in the shell**

Quote every URL that contains a `?`, or zsh, the macOS default shell, will try to expand it and fail. And cURL deliberately ignores an uppercase `HTTP_PROXY` for plain-HTTP URLs, so set the lowercase `http_proxy` as well if you need it there.

7.2 Python

Build proxy URLs in one function so rotation, targeting and sessions are just arguments (flag syntax varies by provider). For SOCKS5, add `requests[socks]` and use `socks5h://`.

Python · requests

```python
import os, uuid
from urllib.parse import quote
import requests

USER = os.environ["PROXY_USER"]
PASS = quote(os.environ["PROXY_PASS"], safe="")   # encode @ : / # in passwords
GATEWAY = "gate.example.net:7000"

def proxy_url(country="us", session=None):
    user = f"{USER}-country-{country}"
    if session:
        user += f"-session-{session}"              # same id, same exit IP
    return f"http://{user}:{PASS}@{GATEWAY}"

# Rotating: each call opens a new connection and gets a fresh exit
for _ in range(3):
    url = proxy_url()
    r = requests.get("https://api.ipify.org?format=json",
                     proxies={"http": url, "https": url}, timeout=30)
    print(r.json()["ip"])

# Sticky: one exit IP for a whole multi-step flow
sticky = proxy_url(session=uuid.uuid4().hex[:8])
with requests.Session() as s:
    s.proxies = {"http": sticky, "https": sticky}
    s.get("https://shop.example/store?zip=75201", timeout=30)
    s.get("https://shop.example/product/123", timeout=30)
```

For concurrency, use **httpx**'s async client. Keep-alive applies: ten pooled connections means up to ten exits, not one per request.

Python · httpx (async)

```python
import asyncio, httpx

async def fetch_all(urls, proxy):
    limits = httpx.Limits(max_connections=10)   # 10 pooled connections = up to 10 exits
    async with httpx.AsyncClient(proxy=proxy, limits=limits, timeout=30) as client:
        return await asyncio.gather(*(client.get(u) for u in urls), return_exceptions=True)

results = asyncio.run(fetch_all(urls, proxy_url()))
```

7.3 Node.js and Puppeteer

Node's built-in `http` module and `fetch` ignore proxy environment variables by default, so you pass a proxy agent explicitly. With the **undici** package, import `fetch` and `ProxyAgent` together so their versions always match.

Node.js · undici

```
import { fetch, ProxyAgent } from 'undici';

const agent = new ProxyAgent(process.env.PROXY_URL);   // http://user:pass@host:port
const res = await fetch('https://api.ipify.org?format=json', { dispatcher: agent });
console.log(await res.json());
```

Node.js · axios

```
import axios from 'axios';
import { HttpsProxyAgent } from 'https-proxy-agent';

const agent = new HttpsProxyAgent(process.env.PROXY_URL);
const { data } = await axios.get('https://example.com', {
  httpsAgent: agent,
  proxy: false,   // stop axios applying its own proxy handling on top
});
```

Puppeteer

Chrome doesn't accept credentials inside `--proxy-server`, so pass the address at launch and the credentials per page. A launch-level proxy covers every page in that browser, so run one browser per identity, or use Playwright's per-context proxies (next page).

Node.js · Puppeteer

```
import puppeteer from 'puppeteer';

const browser = await puppeteer.launch({
  args: ['--proxy-server=http://gate.example.net:7000'],   // address only
});
const page = await browser.newPage();
await page.authenticate({
  username: process.env.PROXY_USER,
  password: process.env.PROXY_PASS,
});
await page.goto('https://api.ipify.org?format=json');
await browser.close();
```

7.4 Playwright, Scrapy and Antidetect Browsers

Playwright gives every browser context its own proxy, locale and timezone, so one browser can run several consistent identities, and it makes blocking heavy assets easy.

Node.js · Playwright

```
import { chromium } from 'playwright';

const browser = await chromium.launch();
const context = await browser.newContext({            // one context = one identity
  proxy: { server: 'http://gate.example.net:7000',
           username: `${process.env.PROXY_USER}-session-a1b2`, password: process.env.PROXY_PASS },
  locale: 'en-US',
  timezoneId: 'America/Chicago',                       // match the exit's location
});
await context.route('**/*', (route) =>                 // skip images, media and fonts
  ['image', 'media', 'font'].includes(route.request().resourceType())
    ? route.abort()
    : route.continue());
const page = await context.newPage();
await page.goto('https://example.com');
```

Python · Scrapy

```
import scrapy

class ProductSpider(scrapy.Spider):
    name = "products"
    start_urls = ["https://shop.example/catalog"]

    def start_requests(self):
        for url in self.start_urls:
            # HttpProxyMiddleware is on by default and reads credentials from the URL
            yield scrapy.Request(url, meta={"proxy": PROXY_URL})
```

Antidetect browsers

1. **One profile, one proxy.** Paste the proxy into the profile's settings; most tools accept `host:port:user:pass`.
2. **Run the built-in proxy check before the first login,** and confirm the exit country is the one you expect.
3. **Let the profile follow the IP.** Set timezone, language and WebRTC to match the proxy; most tools can do it automatically.
4. **Never share a proxy between profiles,** and never move a profile to a different country mid-life.

7.5 Verify Before You Scale

Ten minutes of checks now saves days of confusing data later. Run through this list with every new provider, proxy type or browser profile before it touches production traffic.

- [ ] The exit IP differs from yours and sits in the country you targeted.
- [ ] DNS goes through the proxy (`socks5h` or an HTTP proxy), not your local resolver.
- [ ] Timezone, `Accept-Language` and locale match the exit location.
- [ ] Credentials load from environment variables or a vault, never from code.
- [ ] Its ASN and connection type match what you paid for: residential IPs show a consumer ISP.
- [ ] WebRTC can't reveal your real IP in any automated browser.
- [ ] You've recorded baseline latency and success rate on one real target.
- [ ] Usage alerts are set in your provider's dashboard.

Free ProxyHorizon tools for the job

What Is My IP
See the exit IP your browser shows the world.

IP Lookup
ASN, ISP, location and type for any address.

Proxy Checker
Test speed, anonymity and uptime.

My Fingerprint
Check what your browser leaks, WebRTC included.

cURL Builder
Build proxy-ready cURL commands visually.

Random User-Agent
Realistic, current browser user agents.

KEY TAKEAWAYS

- Prove the proxy works with cURL first; `-v` tells you whether the proxy or the target is failing.
- Build proxy URLs in one function so rotation, targeting and sessions are just arguments.
- Browsers take the proxy address and the credentials separately; Playwright can isolate one identity per context.
- Verify IP, ASN, DNS, WebRTC and locale before any new setup touches production.

08



Staying Unblocked

Getting blocked is rarely about one thing. Anti-bot systems score dozens of signals across six layers, and your proxy controls only the first. This chapter covers the other five, and gives you a fast way to find the cause when something breaks.

IN THIS CHAPTER

8.1 How websites spot bots
8.2 Look like the browser you claim to be
8.3 Pace yourself
8.4 Read the status codes
8.5 A diagnostic path for failures

8.1 How Websites Spot Bots

Modern bot management, the kind built into Cloudflare, Akamai, DataDome and their peers, doesn't make one yes-or-no decision. It builds a risk score as your request travels through a series of checks, starting with the first packet and continuing through every click. Lenient sites look at the first two or three layers. Strict ones use all six.

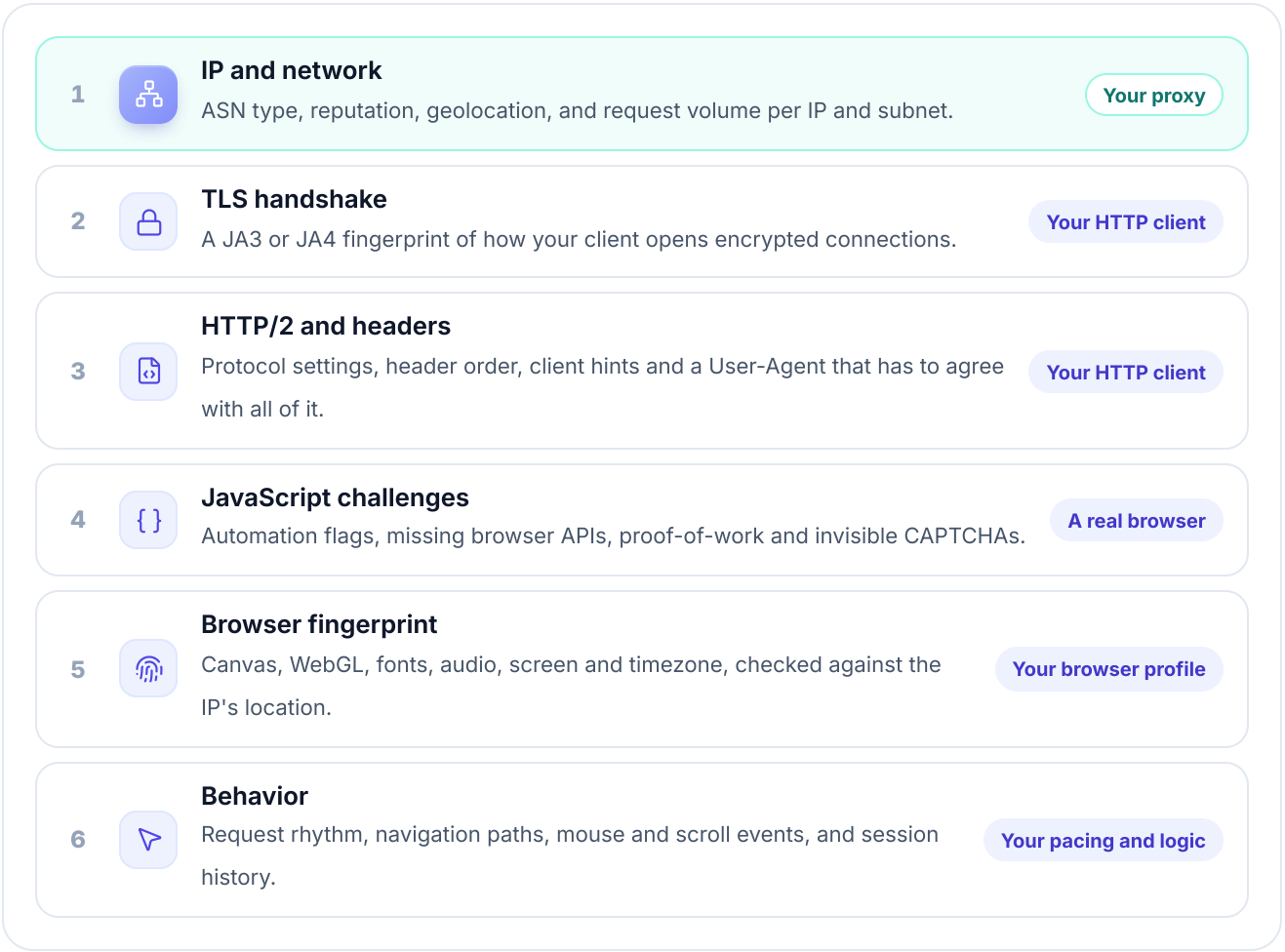


Figure 8.1 Six layers, checked from the first packet to the last click. A proxy only answers layer 1; layers 2 to 6 are decided by the client you run.

This is why a premium residential IP can still be blocked in under a second: if your TLS handshake announces "Python script" at layer 2, the clean IP from layer 1 doesn't save you. It's also why a modest datacenter IP paired with a real, well-configured browser often gets through lenient sites without trouble.

Proxies fix where you come from, not what you are

When a block survives a fresh, high-quality IP, stop buying more IPs. The problem lives in layers 2 to 6, and the next two sections show how to fix it.

8.2 Look Like the Browser You Claim to Be

Anti-bot systems care less about any single value than about whether your values agree with each other. A request that claims to be Chrome on Windows, from a home connection in Texas, at 10 a.m. local time, has to look like exactly that at every layer. These are the mismatches that give automation away most often.

Mismatch	What it tells the site	Fix
User-Agent says Chrome, TLS says Python	An HTTP library pretending to be a browser	Use a real browser or a TLS-impersonating client
German IP, US language and New York timezone	Someone hiding where they are	Match locale and timezone to the exit
`navigator.webdriver` **is true**	An automated browser	Hardened automation or an antidetect browser
Mobile IP, desktop fingerprint	A spoofed setup	Match the device to the network
Requests at perfectly even intervals	A script on a timer	Add jitter and realistic pauses
A brand-new session jumps straight to deep pages	No browsing history at all	Enter through normal pages and keep cookies

Fix the TLS layer without a browser

A full browser solves layers 2 to 5 at once, but it's heavy. When you only need HTML or JSON, a client that impersonates a real browser's TLS and HTTP/2 fingerprint is far cheaper. In Python, **curl_cffi** keeps the familiar requests-style API:

Python · curl_cffi

```python
from curl_cffi import requests

r = requests.get(
    "https://example.com",
    impersonate="chrome",              # Chrome's TLS and HTTP/2 fingerprint
    proxies={"https": PROXY_URL},
    timeout=30,
)
```

> ⚠ **Don't randomize everything**
>
> A fresh random fingerprint on every request is a signal of its own. Real users are boringly consistent: same device, same browser, same timezone, visit after visit. Pick a believable identity per session and keep it.

8.3 Pace Yourself

Sites count requests per IP, subnet, session and endpoint, and block whatever crosses the line first. A big pool helps, but rhythm matters too: a thousand IPs hitting one endpoint in the same second still look like one bot.

A `429 Too Many Requests` often comes with a `Retry-After` header. Honor it. Otherwise, back off exponentially with random jitter so retries don't arrive in synchronized waves.



Figure 8.3 Full-jitter backoff. Before each retry, wait a random time anywhere from zero up to the bar's height. The window doubles each time and is capped at 30 seconds.

Python · retries with backoff

```python
import random, time

RETRYABLE = {429, 500, 502, 503, 504}

def backoff(attempt, base=1.0, cap=30.0):
    return random.uniform(0, min(cap, base * 2 ** attempt))   # full jitter

def get_with_retries(session, url, attempts=5):
    for attempt in range(attempts):
        r = session.get(url, timeout=30)
        if r.status_code not in RETRYABLE:
            return r
        retry_after = r.headers.get("Retry-After", "")
        time.sleep(float(retry_after) if retry_after.isdigit() else backoff(attempt))
    return r
```

Start slow, then ramp

Open with two to five concurrent requests per target domain. Raise concurrency step by step while your success rate holds, and drop back the moment it falls. The ceiling you find is specific to that site, so record it.

8.4 Read the Status Codes

Every failure tells you something, if you know where it came from. Codes in the 4xx range from the *target* point to your identity or your pace; a 407 always comes from the *proxy*; and timeouts usually mean something between the two.

Code or symptom	Usually means	What to do
200, but the wrong page	A soft block: CAPTCHA or interstitial	Validate content; slow down; fix the fingerprint
302 to a CAPTCHA page	Your session has been flagged	Back off; fresh IP and clean cookies
403 Forbidden	IP or fingerprint blocked	Check TLS and headers; climb the ladder
407 Proxy Auth Required	The proxy rejected you	Credentials, allowlist, plan, password encoding
429 Too Many Requests	You hit a rate limit	Honor `Retry-After`; back off; lower concurrency
451 Unavailable	Blocked in that region	Target a different country
502, 503, 504	Trouble at the exit or the target	Retry with a new session; check provider status
Timeouts and resets	Dead exit, slow peer or a firewall	Shorter connect timeout; retry; try another port
Endless CAPTCHAs	Reputation or fingerprint problem	Stop solving; fix the cause; consider an unblocker

Build a block dashboard

Track success rate, CAPTCHA rate and the share of 403 and 429 responses per target and per proxy type. Blocks show up there hours before they show up as gaps in your data, and the trend tells you which layer to look at.

Solve CAPTCHAs, or avoid them?

CAPTCHA-solving services exist, and some workflows legitimately need them. But a CAPTCHA is a symptom: the site already distrusts you. Solving it treats the symptom at a per-page cost and teaches the site nothing good about you. Fixing the layer that triggered it is cheaper at any real volume.

8.5 A Diagnostic Path for Failures

When requests start failing, work down this list in order. Each question rules out a whole class of problems, so you stop guessing and start fixing.

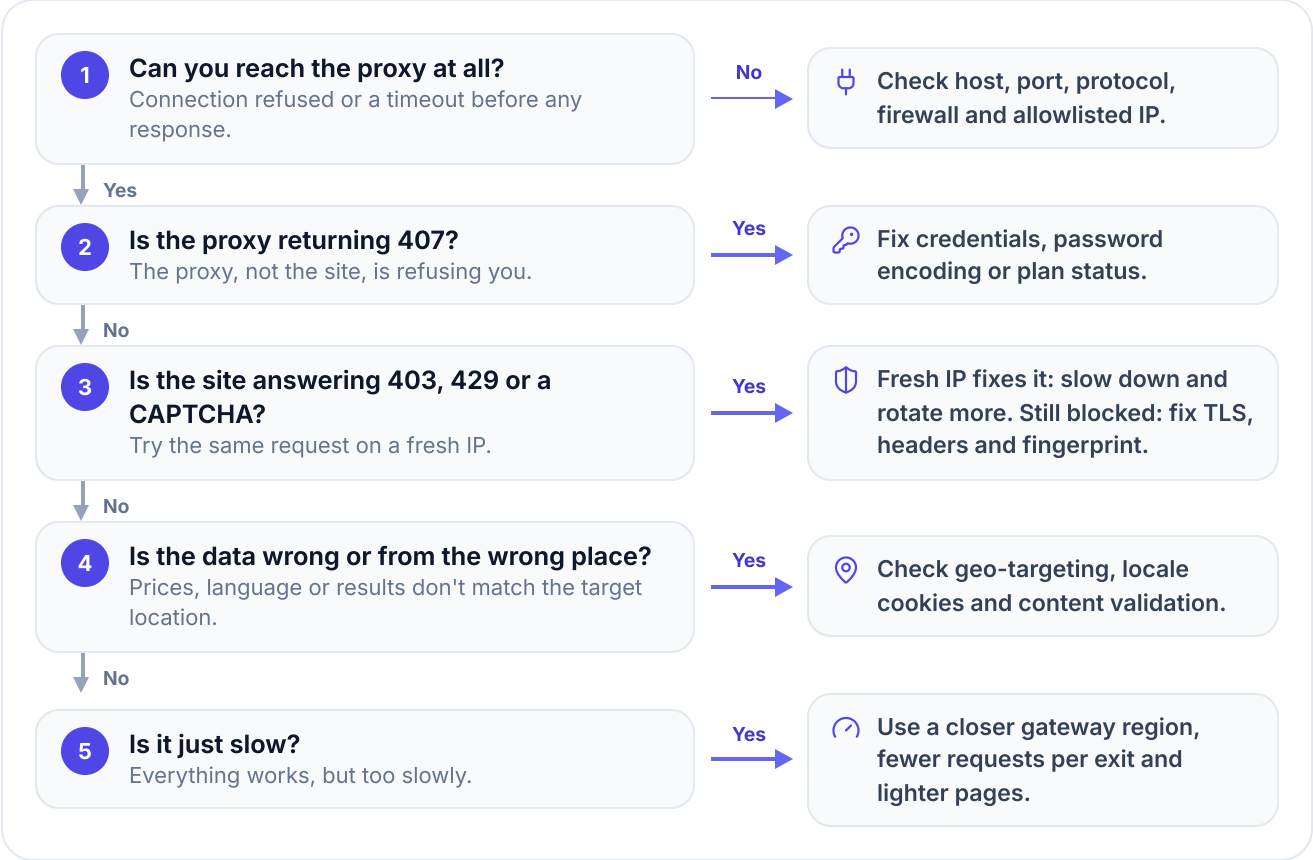


Figure 8.5 Work from the network outward: first the proxy, then the site's verdict, then the data itself, then speed.

KEY TAKEAWAYS

- Anti-bot systems score six layers; a proxy only covers the first one.
- Consistency beats randomness. Every signal should tell the same story as the IP.
- Honor `Retry-After`, back off with jitter, and ramp concurrency per site.
- Diagnose in order: reach the proxy, authenticate, read the site's verdict, check the data.

Go deeper: How Anti-Bot Systems Detect Automated Browsers · Browser Fingerprinting

09



Scaling and Cost Control

Getting a proxy to work is the easy part. Running millions of requests a month without the bill or the block rate creeping up takes discipline: measure the right numbers, stop downloading what you don't need, and let the data route your traffic.

IN THIS CHAPTER

9.1 Measure what matters

9.2 Cut bandwidth, not corners

9.3 Tune for throughput

9.1 Measure What Matters

Teams that run proxies well can answer six questions about any target at any moment. Teams that struggle usually track one number, total spend, and find out about blocks from an angry stakeholder. Instrument these from day one.

Success rate

`valid responses ÷ attempts`

Count by content, not by status code.

Cost per success

`proxy spend ÷ valid responses`

The number to budget and compare on.

Latency

`p50 and p95 response time`

Watch the tail; p95 is where timeouts start.

Megabytes per success

`bytes moved ÷ valid responses`

Your bandwidth efficiency, per target.

Block rate

`(403 + 429 + CAPTCHAs) ÷ attempts`

Rising block rate is the earliest warning.

Session survival

`actual ÷ requested sticky time`

How long sticky IPs really hold.

Log one line per request

Every metric above falls out of a simple log. Write one structured line per request and you can slice success, cost and speed by target, provider, proxy type or hour of the day.

One log line per request

```
{"ts": "2026-10-29T10:14:03Z", "target": "shop.example", "proxy_type": "residential",
 "provider": "provider-a", "session": "k7x2", "exit_ip": "198.51.100.24",
 "status": 200, "valid": true, "bytes": 148213, "latency_ms": 812, "attempt": 1}
```

Put the validator in the log

The `valid` field is the one most teams skip and the one that matters most. Set it from a real content check, such as a CSS selector or a JSON schema, so soft blocks that return HTTP 200 can't inflate your success rate.

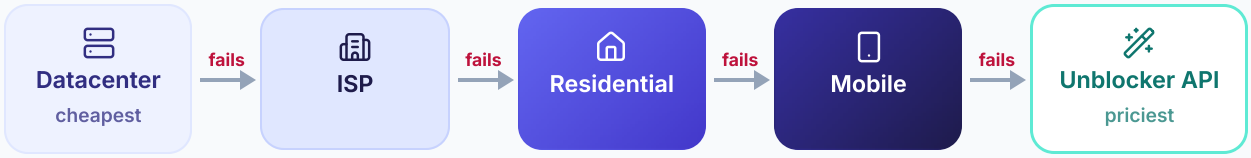
9.2 Cut Bandwidth, Not Corners

On per-GB plans, every byte you don't download is money you keep. These tactics are listed roughly by how much they usually save. None of them costs you data quality.

Tactic	Impact	How
Call the site's JSON endpoints	High	Find them in your browser's network tab. They skip HTML, CSS and images entirely.
Block images, media and fonts	High	Abort those resource types in headless browsers (see the Playwright example in 7.4).
Escape crawling traps	High	Deduplicate URLs, cap pagination and faceted filters, and follow canonical tags.
Route each target to the cheapest rung	High	Sitemaps and category pages over datacenter, protected pages over residential.
Ask for compressed responses	Medium	Send `Accept-Encoding: gzip, br`; most clients already do.
Skip unchanged pages	Medium	Use ETags, `If-Modified-Since` or sitemap dates to re-fetch only what changed.
Cap retries and timeouts	Medium	Retrying a hopeless request forever just burns bandwidth.

The escalation waterfall

Instead of choosing one proxy type per project, let each request start on the cheapest rung and climb only when it fails. Over time, every target settles on the cheapest rung that works for it.



Once a month, re-test each target one rung lower. Sites relax their defenses as often as they tighten them.

Figure 9.2 Each request starts at the cheapest rung and only escalates on failure, so your spend follows each target's real difficulty.

> **Find your heaviest target first**
>
> Sort your targets by megabytes per successful request (section 9.1). A handful of heavy targets usually drives most of the bandwidth bill, so fixing those first pays off fastest and tells you which tactic from the table matters most for your stack.

9.3 Tune for Throughput

At scale, a proxy setup is a small distributed system. The shape below works the same on one server or a fleet of them, because the feedback loop keeps routing honest as targets change.

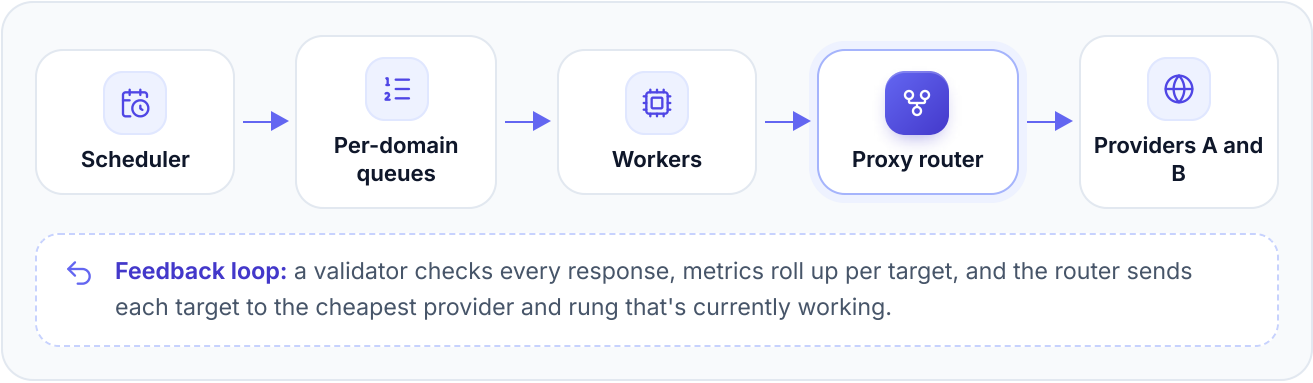


Figure 9.3 A production proxy pipeline. The router is the only part that knows about providers, so switching or adding one never touches your scrapers.

Limit concurrency per domain

Give each target its own limit instead of one global cap, so a slow or strict site can't starve the rest of your jobs.

Set two timeouts

A short connect timeout, 5 to 10 seconds, kills dead exits fast. A longer read timeout fits each target's page weight. Never leave either unset.

Keep a second provider warm

Send 5 to 10% of traffic to a backup so you always have live success data, and fail over automatically when the primary dips.

Schedule around the target

Run heavy crawls in the target's off-peak hours. It's kinder to the site, and quieter hours often mean fewer challenges.

KEY TAKEAWAYS

- Track success rate, cost per success, latency, megabytes per success, block rate and session survival.
- The biggest savings come from not downloading things: APIs over pages, no images, no crawl traps.
- Start every target on the cheapest rung and escalate only on failure; re-test downward monthly.
- Put provider choice behind one router with a warm backup, and let validated metrics steer it.

10



Security, Ethics and What's Next

The last chapter covers what keeps a proxy operation safe to run for years: protecting your own data, collecting other people's responsibly, and knowing where the industry is heading so today's setup still works tomorrow.

IN THIS CHAPTER

10.1 Why free proxies cost the most

10.2 Leaks that undo your proxy

10.3 Ethics and the law

10.4 The road ahead

10.5 Your 30-day proxy plan

10.1 Why Free Proxies Cost the Most

A free proxy list is tempting when you just want to test something quickly. But someone pays for those servers, and if it isn't you, it's worth asking what they get in return. Usually it's one of the things below.

Reads your traffic

Anything sent over plain HTTP, logins and form data included, is visible to the operator.

Rewrites pages

Unencrypted pages can be altered in transit to inject ads, trackers or malware.

Logs everything

Your IP, timing and destinations can be stored, profiled and sold.

Runs on hacked devices

Many free proxies are compromised machines, and some are deliberate honeypots.

Already burned

Public lists are scraped by everyone, so most IPs are blocklisted within days.

Gone tomorrow

No uptime promise, no support, and no one accountable when it breaks.

HTTPS protects you, if you let it

With HTTPS, an ordinary proxy sees hostnames but not the pages, cookies or credentials inside (Chapter 3). That protection vanishes the moment you switch off certificate checks to make an error go away: `verify=False` in Python, `--insecure` in cURL, `rejectUnauthorized: false` in Node. Fix the certificate error instead. It's usually telling you something important.

Never install a proxy's root certificate

A proxy that asks you to trust its root certificate can decrypt and read all of your HTTPS traffic. Only do that for debugging tools you run yourself, such as a local intercepting proxy on your own machine.

Testing on a budget?

Most reputable providers offer a free trial, a refund window or a small pay-as-you-go plan. A few dollars of real residential traffic will teach you more than a week with a free list, and it won't leak your data.

10.2 Leaks That Undo Your Proxy

A leak is anything that reveals the identity your proxy was supposed to replace, or links identities you meant to keep apart. Most are easy to prevent once you know where to look.

Leak	What it exposes	Prevent it	Test it
DNS lookups	Every domain you visit, to your network or ISP	Let the proxy resolve names: `socks5h` or an HTTP proxy	Watch for local DNS queries while proxied
WebRTC	Your real IP, even behind the proxy	Disable it, or force proxy-only routing in automated browsers	My Fingerprint tool
Timezone and language	Your real region	Match both to the exit location	My Fingerprint tool
Cookies and logins	The link between sessions on different IPs	One cookie jar per identity	Inspect cookies per profile
Reused fingerprints	The link between "different" accounts	One browser profile per identity	Run the check in every profile

Guard the keys as well as the traffic

Proxy credentials are spending authority. A leaked password can burn through a month's bandwidth in an afternoon, and the abuse will be traced to your account. The hygiene list in section 3.2 takes five minutes; do it before you go to production, and again whenever someone leaves the team.

Separate what you can't afford to mix

Keep production scraping, account management and personal browsing on separate sub-users, separate machines or containers, and separate browser profiles. When one of them gets flagged or leaks, the blast radius stays small.

Free tool: see what any browser profile reveals, WebRTC included, with My Fingerprint · **Read:** Why Online Privacy Matters

10.3 Ethics and the Law

General information, not legal advice. Laws differ by country and change often; for anything high-stakes, talk to a lawyer.

Public versus personal data. Collecting publicly available, non-personal information, like prices, product details and public listings, is generally the lower-risk end of the spectrum. Personal data, meaning anything that identifies a person, brings obligations under laws such as the EU's GDPR and California's CCPA: a lawful basis for processing, collecting only what you need, keeping it secure and honoring people's requests about it.

Access controls and terms. In the US, courts have generally held that reading public pages isn't "unauthorized access" under the Computer Fraud and Abuse Act. Breaking terms of service you've agreed to, for example by scraping while logged in, can still create contract liability, and getting around technical access controls is riskier still. **robots.txt** isn't law in most places, but it's the site's stated wish, and respecting it shows good faith.

Responsible collection

- [ ] Have a clear, legitimate purpose for the data.
- [ ] Prefer official APIs, feeds and licensed datasets.
- [ ] Respect robots.txt and rate limits; never degrade a site.
- [ ] Skip personal data you don't need; secure what you keep.
- [ ] Don't use proxies to get around logins, paywalls or bans.
- [ ] Honor opt-out and takedown requests promptly.

Ask your provider

- [ ] How do residential and mobile peers consent?
- [ ] How are peers paid or rewarded?
- [ ] Can peers opt out at any time?
- [ ] Do you verify customers (KYC)?
- [ ] How do you enforce your acceptable-use policy?
- [ ] How do you handle abuse reports?

Where ProxyHorizon draws the line

Proxies are infrastructure for legitimate work: research, price intelligence, ad verification, testing and privacy. They are not a tool for fraud, credential stuffing, scalping, harassment or evading bans you've earned, and no setup in this handbook is meant for those.

10.4 The Road Ahead

The proxy market moves fast, but the direction of travel is clear. These are the shifts most likely to change how you build over the next few years.

AI agents everywhere

Automated browsing is growing quickly, and sites are answering with stricter bot management plus ways for good bots to identify themselves, such as proposals for cryptographically signed agent requests.

Paying for access

Some CDNs now let publishers block AI crawlers by default or charge them per crawl. Expect more data to sit behind licensing deals instead of open pages.

Detection moves up the stack

TLS, HTTP/2 and behavioral models carry more weight every year. IP quality still matters, but on its own it buys less than it used to.

IPv6 grows up

More sites accept IPv6, which makes IPs cheaper, and rate limits move from single addresses to whole prefixes.

Managed access

Scraping APIs and unblockers absorb more of the hard work, and ready-made datasets replace some scraping entirely.

Higher sourcing standards

After high-profile botnet takedowns, buyers and regulators are asking harder questions about where residential IPs come from.

Future-proof your setup

- Hide the proxy layer behind one interface, so switching providers is a config change.
- Keep two providers live and route by measured cost per success.
- Invest in browser identity and consistency, not just more IPs.
- Prefer official APIs and licensed data where they exist.
- Review your legal and ethical posture once a year.

10.5 Your 30-Day Proxy Plan

You've read the whole handbook. Here's how to turn it into a working setup in a month, one focused week at a time.

Week 1

Map your targets

List the sites and data you need. Test each one on datacenter IPs, note which pages block you, and sort targets by the rung they seem to need.

CHAPTERS 2 AND 5

Week 2

Run a fair trial

Trial two or three providers on your hardest targets with one script. Compare success rate, latency and cost per success, then pick a primary and a backup.

CHAPTER 6

Week 3

Build the plumbing

One proxy-URL function, sticky sessions where flows need them, retries with backoff, content validation and one log line per request.

CHAPTERS 4, 7 AND 8

Week 4

Optimize, then scale

Block heavy assets, route each target to its cheapest rung, set usage alerts and a block dashboard, then raise volume step by step.

CHAPTER 9

Proxies reward the teams that treat them as engineering rather than magic. Measure honestly, stay consistent, respect the sites you depend on, and the blocks get rarer while the bills get smaller.

Whenever you need to compare providers, test a setup or go deeper on a topic, ProxyHorizon has you covered:

Proxy directory | Compare side by side | Free tools | Guides and tutorials

APPENDIX A

Glossary

Anonymous proxy

Hides your real IP from websites but still reveals that a proxy is in use.

Antidetect browser

Runs many isolated browser profiles, each with its own fingerprint, cookies and proxy, so sites see separate, genuine users.

ASN (Autonomous System Number)

Identifies the network an IP belongs to, and reveals whether it's a home ISP, a data center or a mobile carrier.

Backconnect proxy

One endpoint that rotates you through a large pool. You connect once; the provider picks the exit for each connection.

Bandwidth

The amount of data transferred, and the most common way residential and mobile proxies are billed.

Browser fingerprinting

Identifying a browser by combining details like screen size, fonts, GPU and timezone into a signature that works without cookies.

CAPTCHA

A challenge-response test that separates humans from bots, from ticking a box to identifying images.

CGNAT (Carrier-Grade NAT)

How carriers share one public IP among many subscribers, which is why mobile proxy IPs are so trusted.

Datacenter proxy

An IP owned by a hosting company. Fast, cheap and plentiful, and easy for strict sites to identify.

DNS leak

Domain lookups that bypass your proxy or VPN and reveal the sites you visit to your network or ISP.

Elite proxy

Hides your IP and adds no proxy headers, so the request looks like a direct visit.

Exit IP

The address your request leaves the proxy network from, and the only one the website sees.

Forward proxy

A server that makes requests on your behalf. Every commercial proxy you buy is one.

Geo-targeting

Choosing exit IPs in a specific country, region, city or network.

Headless browser

A real browser running without a window, driven by code instead of a person.

IP reputation

A trust score for an IP based on its history; clean IPs pass, flagged ones get challenged or blocked.

IP rotation

Changing the source IP so no single address exceeds what a target tolerates.

IP allowlisting

Authorizing pre-approved IP addresses to use a service without a username and password.

ISP proxy

A static IP registered to an internet provider but hosted on datacenter hardware. Also sold as static residential.

APPENDIX A, CONTINUED

JA3 / JA4 fingerprint

A hash of how a client sets up a TLS connection, used to recognize tools even when their IPs change.

Mobile proxy

Routes traffic through a real 4G or 5G connection; the hardest proxy type to block.

Proxy pool

The collection of IPs a provider or tool rotates through.

Proxy server

A middleman between you and the internet; the website sees its IP instead of yours.

Rate limiting

A cap on how many requests a client may make in a given time, and one of the most common defenses scrapers meet.

Residential proxy

Routes traffic through a real home connection, so sites see an ordinary ISP customer.

Reverse proxy

Sits in front of web servers and handles requests for them: caching, load balancing, HTTPS and bot defense.

robots.txt

A file at a site's root that tells crawlers what they may access. A convention, not a technical lock.

Rotating proxy

A gateway that hands you a new IP per request or on a timer.

SOCKS5

A proxy protocol that forwards any TCP connection, and UDP where supported, without parsing it.

Sticky session

Keeps the same exit IP for a set time so multi-step flows stay on one address.

Subnet

A block of addresses carved out of a larger range. Sites often ban whole subnets at once.

TLS fingerprinting

Identifying a client from how it negotiates encryption, before any page data is exchanged.

Transparent proxy

Intercepts traffic without hiding your IP, often without you knowing it's there.

User agent

The string a browser sends to name itself, its version and its operating system.

WebRTC leak

Your real IP exposed through the browser's real-time communication feature, even behind a proxy.

Adapted from the ProxyHorizon Glossary, which has longer explanations, examples and FAQs for 80+ terms at proxyhorizon.com/glossary.

APPENDIX B

Checklists

Provider scorecard

Score each provider 1 to 5 during your trial (section 6.4), multiply by the weight and add up the column.

Criterion	Weight	Provider A	Provider B	Provider C
Success rate on your targets	×2			
Pool depth in your locations	×2			
Speed and latency	×2			
Targeting options	×1			
Session control	×1			
Protocol support	×1			
Pricing model fit	×1			
Ethics and compliance	×1			
Tooling	×1			
Support and trial	×1			
Total	**/65**			

Before you launch

- [ ] Each target tested on the cheapest rung first
- [ ] Proxy URLs built in one function
- [ ] Sticky sessions only where flows need them
- [ ] Retries with backoff; `Retry-After` honored
- [ ] Responses validated by content
- [ ] One log line per request
- [ ] Heavy assets blocked in browsers
- [ ] IP, DNS, WebRTC and locale verified
- [ ] Credentials in env vars or a vault
- [ ] Usage alerts set with the provider

Every week

- [ ] Success and block rate per target
- [ ] Cost per successful request
- [ ] p95 latency and timeout rate
- [ ] Bandwidth against plan and alerts
- [ ] Backup provider's live success rate

Every month

- [ ] Re-test each target one rung lower
- [ ] Rotate shared credentials
- [ ] Review terms and robots.txt changes

APPENDIX C

Quick Reference

Proxy URL formats

URL	`http://user:pass@host:port`
SOCKS5	`socks5h://user:pass@host:port`
Colon list	`host:port:user:pass`
At-sign list	`user:pass@host:port`
Allowlisted	`host:port`

Encode these in passwords

`@`	`%40`	`:`	`%3A`	`/`	`%2F`
`#`	`%23`	`?`	`%3F`	`%`	`%25`

cURL flags

`-x URL`	Use this proxy
`-U user:pass`	Proxy credentials, kept out of the URL
`-v`	Show the CONNECT exchange
`--connect-timeout 10`	Give up on dead exits fast
`-m 60`	Cap the whole request
`-w "%{time_total}"`	Print timings

Environment variables

`HTTPS_PROXY`	Proxy for HTTPS URLs
`http_proxy`	Proxy for HTTP URLs (lowercase for cURL)
`NO_PROXY`	Hosts that skip the proxy
`ALL_PROXY`	Fallback for any protocol

Status codes at a glance

200 + wrong page	Soft block. Validate content.	**302 to a CAPTCHA**	Flagged session. Back off, fresh IP.
403	Blocked. Check TLS, headers, proxy type.	**407**	From the proxy. Credentials or allowlist.
429	Rate limited. Honor Retry-After.	**451**	Region blocked. Change country.
502 / 503 / 504	Exit or target trouble. Retry, new session.	**Timeouts**	Dead exit or firewall. Shorter connect timeout.

APPENDIX D

Free Tools and Further Reading

Everything below is free on ProxyHorizon, no account needed for the browser tools.

Free tools

Proxy Checker	Speed, anonymity and uptime	**IP Lookup**	ASN, ISP, location and type
What Is My IP	The IP the world sees	**My Fingerprint**	What your browser leaks
Proxy Format Converter	Switch between list formats	**Proxy Rotator Generator**	Rotation scripts in your language
cURL Builder	Proxy-ready commands	**Random User-Agent**	Current, realistic user agents
User-Agent Parser	Decode any user agent	**Subnet Calculator**	Ranges, masks and sizes

Directories and comparison

Proxy providers	Tested, rated and compared	**Comparison tool**	Any two providers, side by side
Antidetect browsers	Profiles, pricing and features	**VPN services**	For personal privacy

Guides worth reading next

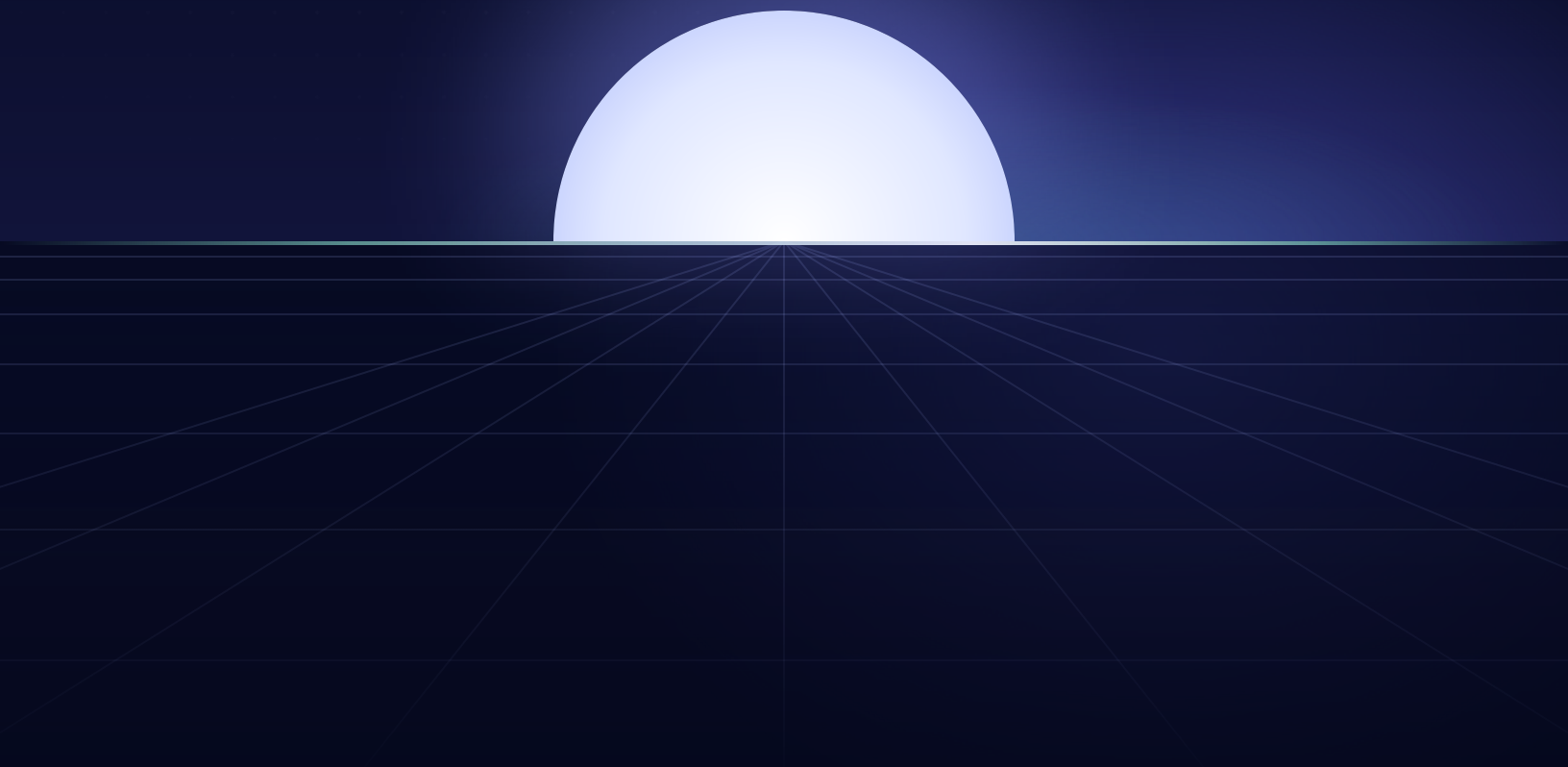
- **How Do Proxies Work?**
- **What Is a Residential Proxy?**
- **What Are Rotating Proxies?**
- **How SOCKS5 Proxies Work**
- **How Websites Detect Proxy Traffic**
- **Handling Proxy Timeouts and Errors**
- **Scraping Cloudflare-Protected Sites**
- **What Is Web Scraping?**

ProxyHorizon

Compare confidently. Implement correctly.

ProxyHorizon is the independent directory and resource hub for proxies, VPNs and antidetect browsers: expert-tested comparisons, free tools and practical guides to choose and use them.

- Compare 120+ proxy providers
- Free proxy and network tools
- In-depth, fact-checked guides



© 2026 PROXYHORIZON · THE ULTIMATE PROXY HANDBOOK