Finding the Origin IP Behind Cloudflare

Finding the Origin IP Behind Cloudflare This is a well-known problem (sometimes called **"origin IP discovery"** or **"de-cloudflaring"**). Cloudflare's whole point is to hide it, so there's no single reliable method — but you can combine several.

---

1. Passive DNS / DNS History (most effective)

Before the site added Cloudflare, its A record pointed directly to the origin. DNS history services logged that.

Service	How
ViewDNS.info (dns view history)	Type the domain → "DNS History" tab
SecurityTrails	`securitytrails.com` → domain → "DNS Records" (has a history timeline)
RapidDNS (rapiddns.io)	Enter domain, shows all past A/CNAME records
Censys	Search the domain, look at historical IP associations
DNSTwist / whoisds.com	Similar passive-DNS lookups

> Look for an A record that is **not** a Cloudflare IP range (104.16–104.31.x, 172.64–172.71.x, 188.114.x, 190.93.x, 198.41.x, 198.97.x, 162.158.x, 131.0.72.x, 141.101.x, 173.245.x, etc.).

That non-Cloudflare IP from, say, 2019 is very likely still the origin (or was very recently).

---

2. Non-Proxied Subdomains

Site owners often only put the **main** domain (`example.com`, `www`) on the orange cloud. Other subdomains stay grey (direct):

```
# Quick manual check
for sub in mail ftp dev staging test api blog shop cdn assets media files docs app admin panel; do
  echo -n "${sub}.example.com → "
  dig +short A ${sub}.example.com
done
```

Or use a subdomain enumerator first (`subfinder`, `amass`, `assetfinder`) and then check each A record for a non-Cloudflare IP.

Common culprits: `mail.`, `ftp.`, `dev.`, `staging.`, `old.`, `legacy.`, `internal.`, `api.`, `ws.` (websockets)

If any of those resolve to a real IP (not a Cloudflare range), that's your origin.

---

3. SSL Certificate Transparency Logs

```
crt.sh?q=%25example.com
```

or search on **Censys** / **SSL Labs**.

Sometimes:

- A cert was issued to the **origin IP** directly.
- A cert is issued to a subdomain like `origin.example.com` that resolves to the real server.
- The cert's `subjectAltName` reveals a domain you can `dig` for a non-CF IP.

---

4. Shodan / Censys / FOFA

- **Shodan.io** → search `"example.com"` (it indexes Host headers, banners, certs).
- **Censys.io** → search by domain or by the cert hash.

- **FOFA.so** → same idea.

They often show the IP that *was* or *still is* answering with that domain's Host header, including before or outside Cloudflare.

---

5. Email & DNS Records That Bypass Cloudflare

Mail records are almost never proxied through Cloudflare:

```
dig +short MX example.com
dig +short TXT example.com    # SPF, DKIM, DMARC
dig +short NS example.com
```

The MX server's IP is frequently the **same server** (or same VPS) that hosts the website.

```
dig +short A mail.example.com   # or the MX hostname
```

Also check:

- **SPF record** (`v=spf1 include:...`) → the included domain's IP.
- **DKIM key** → sometimes references a hostname on the origin.
- **SRV records** (for XMPP, SIP, etc.) → often point to the origin.

---

6. Traceroute (long shot)

```
traceroute example.com
# or
mtr example.com
```

You'll see the path to Cloudflare's edge. After the CF hop, the next hop *should* be the origin's upstream ISP — and if you can identify that ASN/IP, you've narrowed it down. Not super reliable, but occasionally the last hop before "timeout" gives a hint.

---

7. Leaks in the Site Itself

Look at the page source / network tab in DevTools:

- **Hardcoded IPs** in JS, CSS, or HTML (`https://192.0.2.1/api/...`)
- **WebSocket URLs** pointing to a raw IP or non-CF domain
- **`X-Forwarded-For`**, `X-Real-IP`, or custom headers in API responses
- **CORS errors** in the console that reveal the origin
- **Redirect chains**: `example.com/old` → `http://192.0.2.1/...`
- **`Server` header** mismatch (e.g., Cloudflare says `cloudflare` but a sub-resource says `nginx/1.18` on a different IP)
- **Error pages** that leak the origin's hostname (Apache/Nginx default 502/503 pages)

---

8. WHOIS / RDAP (historical)

Check if the domain's registrar history or name-server history reveals a prior hosting provider. Less useful now, but sometimes the old NS pointed to a server whose IP you can trace.

---

9. Brute-Force / Ping Sweep (last resort)

If you've narrowed it to a /24 or a small range (e.g., you know it's an AWS `54.200.x.x` block):

```
for i in $(seq 1 254); do
  ping -c1 -W1 54.200.1.$i &
done
wait
```

Then `curl -H "Host: example.com" http://<ip>` on each live IP to see if it responds with the site's content.

---

10. If You Own / Control the Site

Open the **Cloudflare dashboard** → **Network** (or **DNS**) tab. The origin IP is right there, and you can also see it under **Overview → your zone's "origin" info**.

Practical "Try In This Order" List

1. **Passive DNS history** (ViewDNS / SecurityTrails) ← fastest, most likely to work
2. **Subdomain enumeration +** `dig` for non-CF IPs
3. **MX / SPF / DKIM** → resolve those hostnames
4. **crt.sh / Censys** for cert-based clues
5. **Shodan / FOFA** search by domain
6. **Inspect the site's JS / network traffic** for hardcoded IPs
7. **Traceroute** as a sanity check
8. **Ping sweep** if you've narrowed the range

Quick-Start Script (Python-ish)

```
DOMAIN="example.com"

echo "=== MX / SPF ==="
dig +short MX $DOMAIN
dig +short TXT $DOMAIN | grep spf

echo "=== Common subdomains ==="
for sub in mail ftp dev api blog shop ws cdn; do
  IP=$(dig +short A ${sub}.$DOMAIN)
  [[ -n "$IP" ]] && echo "${sub}.${DOMAIN} → $IP"
done

echo "=== Check if any IP is NOT Cloudflare ==="
# Cloudflare ranges (simplified): 104.16-31, 172.64-71, 188.114, 190.93, 198.41, 198.97, 162.158, 131.0.72, 141.101, 173.245
# If the IP doesn't match, it's likely the origin.
```

Want me to run through this for a **specific domain**? Give me the URL and I can walk through the steps.

Revision #1
Created 11 September 2026 15:23:06 by Togoboi
Updated 11 September 2026 15:24:49 by Togoboi