NEWS · MAY 25, 2026

What's my public IP? Meet ifconfig.me

A wonderfully simple service that answers a question every developer (and especially every Delphi developer wiring up a server) eventually asks — "what does the internet see as my IP right now?"

By Dr. Holger Flick

If you've ever set up a VPS, configured a router, asked a hosting provider to whitelist your office, debugged a webhook that wouldn't fire, or pointed a DNS record at a freshly-provisioned server, you've hit the same question I have a thousand times:

> *What does the rest of the internet think my IP is, right now?*

On Windows, the instinct is to open `cmd` and type `ipconfig`. On Linux, `ip addr` or the old `ifconfig`. On macOS, `ifconfig en0`. And every single one of those tools answers the wrong question. They tell you the address of your **network interface** — which, behind almost any modern setup (NAT, Wi Fi router, corporate firewall, VPN, cloud VPC), is *not* what the public internet sees.

For years I'd open a browser, search "what is my ip", click whichever blue link looked least sketchy, dismiss a cookie banner, and copy the number off a page that thought I needed three banner ads to find it. Recently I finally moved on to a tool that should have been my default a decade ago.

ifconfig.me — one URL, one IP, done

`ifconfig.me` is a tiny web service that does exactly one thing: it tells you the IP address it sees your request coming from.

From any shell with `curl`:

```
curl ifconfig.me
```

That's it. No HTML, no banner, no JavaScript. Just the four octets, terminated by a newline. Pipe it, paste it, script it — it behaves like a Unix command, not a web page.

If you want to force IPv4 because your machine prefers v6 but your A record needs v4:

```
curl -4 ifconfig.me
```

Or the opposite:

```
curl -6 ifconfig.me
```

Want a quiet output for use inside a script? Add `-s` so curl skips its progress bar:

```
PUBLIC_IP=$(curl -4 -s ifconfig.me)
echo "Updating DNS to $PUBLIC_IP"
```

Why this matters for us Delphi folks

A lot of us live in two worlds. We've been writing Delphi software for 20+ years, and the network side of things — the part where the binary actually meets the public internet — has gradually become a bigger chunk of the job.

You ship a desktop client that needs to talk back to a REST API hosted on your own server? You need to know which IP to put in your DNS record.

You're testing a TMS WEB Core app from your home office and the backend on AWS is firewalled to a whitelist? You need to know your current home IP.

You're connecting a piece of Delphi machinery (a Windows service running on a customer site, a kiosk somewhere, a manufacturing controller) and you want to verify the device can reach your cloud and that the outbound NAT mapping is what you expect? You ask the device "what IP do you look like from the outside?"

In every one of those cases, `ipconfig /all` on the Windows box lies — or at best, tells you a 192.168.x or 10.x address that's true *locally* and meaningless *externally*. `ifconfig.me` answers the actual question.

Doing it from Delphi

I needed this in a Delphi service recently and it ended up being four lines:

```
uses
  System.Net.HttpClient;

function GetPublicIP: string;
begin
  with THTTPClient.Create do
  try
    Result := Trim(Get('https://ifconfig.me').ContentAsString);
  finally
    Free;
  end;
end;
```

No JSON parsing, no XML, no API key, no rate-limit headers to interpret. The response *is* the IP. That's it.

If you want more (your user agent, your reported language, what port your connection came from), `ifconfig.me` exposes those as paths:

```
curl ifconfig.me/ua       # your user agent
curl ifconfig.me/port     # remote port number
curl ifconfig.me/host     # reverse DNS of your IP
curl ifconfig.me/all      # all of the above, JSON
```

That `/all` endpoint is genuinely useful when you're chasing a "but it works from my machine" bug — it tells you what you look like to the server, which is often the missing piece of the puzzle.

A note on trust, and alternatives

`ifconfig.me` is run by Andrée Toonk (the same person behind BGPmon). It's been around since 2009 and is operated as a public-good service. But it *is* a third party — every request reveals your IP to them, and if the service is ever down, your script breaks. So:

For mission-critical scripts I keep a backup or two:

- `curl -4 https://api.ipify.org` — same idea, different operator.
- `curl -4 https://icanhazip.com` — Cloudflare runs this one now.
- `dig +short -4 myip.opendns.com @resolver1.opendns.com` — uses **DNS**, not HTTP. Faster, no third-party HTTP service, and OpenDNS has a special record that just returns the asking IP.

The DNS one is my favorite for production scripts because it's blindingly fast and doesn't depend on any web stack being reachable. The HTTP ones are my favorite for interactive use because I can just type them.

The deeper lesson

The reason `ifconfig.me` feels like a small revelation is that it inverts the usual question. Tools like `ipconfig`, `ip addr`, `ifconfig` all assume you're asking *"what's happening on my machine?"* — and they answer truthfully. But for half the work I do, that's not what I want to know. I want to know *"how does my machine appear to everyone else?"* — which is fundamentally a question only the outside can answer.

The genius of `ifconfig.me` is recognizing that the only honest answer to "what's my public IP" lives on a server somewhere else, and shipping the world's simplest service to provide it.

INDEPENDENT · AD-FREE · READER-SUPPORTED

Free to read. Not free to make.

Every tutorial, article, and line of example code on Holger's Code is published without a paywall and without ads. This page is about what keeps it that way.

Professional work, published free

The content here is held to the same standard as professional client work: every tutorial is built as a real project, tested against current versions, documented step by step, and revisited when the frameworks move on. A single in-depth tutorial routinely takes days — and that's before the bills behind it: servers, storage, build infrastructure, licenses, and the hardware everything is verified on.

If an article or tutorial here has saved you an afternoon of debugging, taught you a technique, or helped you ship, it created real value for you. Supporting it isn't charity — it's paying for professional work you found useful, at whatever price you choose. And if you can't, keep reading anyway. That's what the free part means.

Ways to support

KO-FI

Buy me a coffee

A one-time tip — no account, no subscription, any amount.

BOOKS & COURSES

Buy the paid work

The paid tier of everything published free here — and you get something lasting in return.

SPREAD THE WORD

Share an article

Send a tutorial to a colleague, cite an article, subscribe to the RSS feed.

Nothing here goes behind a paywall, whether you chip in or not. But if something on this site earned its keep today, this is where to say so.

Support on Ko-fi —