Access Your Home Server From Anywhere Without a Public IP Address

FREE • SECURE • NO STATIC IP REQUIRED

Version 2.0 | March 2026

A step-by-step guide for home server enthusiasts

Table of Contents

📋 Before You Begin

Prerequisites Checklist

Ensure you have the following items ready before starting:

- ✅ A domain name (purchased from any registrar - Namecheap, GoDaddy, Google Domains, etc.)
- ✅ A local server or computer (Windows, Linux, or macOS) running your service
- ✅ Your local service already running (e.g., web server on http://localhost:8080)
- ✅ Administrator/root access to your local machine
- ✅ Stable internet connection on your home server
- ✅ Ability to modify nameserver settings at your domain registrar

⏱️ **TIME ESTIMATE:** Complete setup takes 15-30 minutes. DNS propagation may take up to 48 hours but usually completes within 1 hour.

⚠️ **IMPORTANT:** This guide assumes your local service is already running and accessible via localhost or local network IP. If you haven't set up your service yet, do that first.

Step 1: Create and Configure Cloudflare Account

1.1 Sign Up for Cloudflare

1. Open your browser and go to: **https://dash.cloudflare.com/sign-up**
2. Enter your email address and create a strong password
3. Alternatively, sign up using Google or Apple account
4. Check your email and verify your email address by clicking the verification link

1.2 Add Your Domain to Cloudflare

1. After logging in, click the **"Add a site"** button (blue button, top right)
2. Enter your domain name in the field (e.g., `example.com` or `yourdomain.com`)
3. Click **"Add site"**
4. On the next screen, select the **"Free"** plan (shows $0/month)
5. Click **"Continue"**
6. Cloudflare will scan your existing DNS records - you can review them later
7. Click **"Continue"** again

> 📝 **NOTE:** The free plan includes unlimited DNS management, DDoS protection, and SSL certificates - perfect for home use.

1.3 Update Your Domain's Nameservers (CRITICAL STEP)

This step tells the internet to use Cloudflare for your domain:

1. After adding your site, Cloudflare will display two nameserver addresses `Example: dns1.ns.cloudflare.com` and `dns2.ns.cloudflare.com`
2. Log in to your domain registrar's website (where you purchased the domain)
3. Find the DNS management or Nameservers section (often under Domain Settings)
4. Replace the existing nameservers with the two Cloudflare nameservers

5. Save your changes (may take 5-30 minutes to confirm)

⚠️ **CRITICAL:** DNS changes can take from a few minutes to 48 hours to propagate worldwide. During this time, your domain might be temporarily unavailable. This is normal.

✅ **VERIFICATION:** You can check if nameservers have updated using online tools like https://www.whatsmydns.net/ or by running:

```
nslookup -type=NS yourdomain.com
```

Step 2: Install cloudflared on Your Local Server

The `cloudflared` daemon creates the tunnel from your server to Cloudflare's network. Choose your operating system below:

Windows Installation

1. Download the latest Windows version from:
   https://github.com/cloudflare/cloudflared/releases
2. Look for: `cloudflared-windows-amd64.exe`
3. Create folder: `C:\cloudflared`
4. Move the downloaded file to this folder
5. Rename it to: `cloudflared.exe`
6. Open Command Prompt as Administrator
7. Navigate to folder: `cd C:\cloudflared`

Linux Installation (Ubuntu/Debian)

Run these commands in terminal:

TERMINAL

```
curl -L https://pkg.cloudflare.com/cloudflare-main.gpg | sudo tee
/usr/share/keyrings/cloudflare-archive-keyring.gpg >/dev/null

echo "deb [signed-by=/usr/share/keyrings/cloudflare-archive-
keyring.gpg] https://pkg.cloudflare.com/cloudflared $(lsb_release -
cs) main" | sudo tee /etc/apt/sources.list.d/cloudflared.list

sudo apt update
sudo apt install cloudflared
```

macOS Installation

Using Homebrew:

💻 TERMINAL

```
brew install cloudflared
```

Alternative using direct download:

💻 TERMINAL

```
curl -L
https://github.com/cloudflare/cloudflared/releases/latest/download/cloudflared-
darwin-amd64.tgz | tar xz
sudo mv cloudflared /usr/local/bin/
```

Verify Installation

Run this command to confirm cloudflared is installed correctly:

💻 TERMINAL

```
cloudflared --version
```

Expected output: You should see version information (e.g., `cloudflared version 2024.5.0`)

Step 3: Authenticate with Cloudflare

This links your local machine to your Cloudflare account:

1. In your terminal/command prompt, run:

```
💻 TERMINAL

cloudflared tunnel login
```

2. A browser window will open automatically (if it doesn't, copy the URL from the terminal)
3. Log in to your Cloudflare account if prompted
4. You'll see a list of your domains - select the domain you added in Step 1
5. Click **"Authorize"** to grant permission

✅ **SUCCESS:** Cloudflare will download a certificate file (cert.pem) to:
• Windows: `C:\Users\YourUsername\.cloudflared\cert.pem`
• Linux/Mac: `~/.cloudflared/cert.pem`

🔐 **SECURITY NOTE:** This certificate file is sensitive - it proves your identity to Cloudflare. Don't share it or commit it to version control.

Step 4: Create a Tunnel

Now create the actual tunnel that will connect your server to Cloudflare:

1. In your terminal, run (replace "my-home-server" with your preferred name):

💻 TERMINAL

```
cloudflared tunnel create my-home-server
```

After running, you'll see output similar to:

💻 TERMINAL

```
Created tunnel my-home-server with id: 12345678-aaaa-bbbb-cccc-
123456abcdef
```

> 🔑 **SAVE THIS INFORMATION:**
> • Tunnel ID: **12345678-aaaa-bbbb-cccc-123456abcdef**
> • Tunnel Name: **my-home-server**
> • A credentials JSON file was created in your .cloudflared folder

The credentials file location:

- Windows: `C:\Users\YourUsername\.cloudflared\12345678-aaaa-bbbb-cccc-123456abcdef.json`
- Linux/Mac: `~/.cloudflared/12345678-aaaa-bbbb-cccc-123456abcdef.json`

Step 5: Create a Configuration File

The configuration file tells cloudflared how to route traffic from your domain to your local service.

5.1 Create config.yml

Windows Path:

`C:\cloudflared\config.yml`

Linux/Mac Path:

`~/.cloudflared/config.yml`

Copy the following configuration, replacing the placeholder values:

CONFIG.YML

```
tunnel: my-home-server credentials-file: /full/path/to/12345678-aaaa-bbbb-cccc-123456abcdef.json ingress: - hostname: home.example.com service: http://localhost:8080 - service: http_status:404
```

Configuration Parameters Explained:

Parameter	Description	Example
`tunnel`	Your tunnel name or ID	my-home-server
`credentials-file`	Full path to your credentials JSON file	C:\cloudflared\12345678-aaaa-bbbb-cccc-123456abcdef.json
`hostname`	The subdomain you want to use	home.example.com
`service`	Your local service address and port	http://localhost:8080

💡 **TIP:** The last line `service: http_status:404` is required as a catch-all rule for any requests that don't match your hostname. Without it, the tunnel won't start.

Step 6: Route DNS to Your Tunnel

This creates a DNS record that points your subdomain to the tunnel:

💻 TERMINAL

```
cloudflared tunnel route dns my-home-server home.example.com
```

Replace:

- `my-home-server` → your tunnel name
- `home.example.com` → your chosen subdomain

> ✅ **SUCCESS:** Cloudflare automatically adds a CNAME record pointing to your tunnel. You can verify this in the Cloudflare Dashboard under DNS settings.

To verify the DNS record:

1. Log in to Cloudflare Dashboard
2. Click on your domain
3. Go to DNS tab
4. You should see a new CNAME record for your subdomain pointing to `your-tunnel-id.cfargotunnel.com`

Step 7: Test Your Tunnel

Before setting up the tunnel as a permanent service, test it manually:

1. In your terminal, run:

```
💻 TERMINAL

cloudflared tunnel run my-home-server
```

You'll see log output showing the tunnel connecting. Leave this terminal window open.

2. Open a web browser on any device (phone, tablet, another computer)
3. Navigate to: `https://home.example.com` (your subdomain)

Troubleshooting Test Results:

Result	Meaning	Action
✅ See your local service	Tunnel works perfectly!	Proceed to Step 8
❌ 404 Not Found	Hostname mismatch	Check hostname in URL matches config.yml
❌ 502 Bad Gateway	Local service not running	Start your local service
❌ ERR_CONNECTION_REFUSED	Wrong port or service not listening	Verify port in config.yml
❌ 403 Forbidden	DNS routing not set up	Run Step 6 again

Step 8: Run Tunnel as a Service (Recommended)

To keep your tunnel running automatically (even after reboots), install it as a system service.

Windows Service Installation

1. Open Command Prompt as Administrator
2. Navigate to your cloudflared folder:

```
TERMINAL

cd C:\cloudflared
```

3. Install the service:

```
TERMINAL

cloudflared.exe service install --config C:\cloudflared\config.yml
```

4. The service will start automatically. Manage it with:

```
TERMINAL

net start cloudflared
net stop cloudflared
```

Linux Service Installation (systemd)

1. Move configuration files to system location:

```
TERMINAL

sudo mkdir -p /etc/cloudflared
sudo cp ~/.cloudflared/config.yml /etc/cloudflared/
sudo cp ~/.cloudflared/12345678-aaaa-bbbb-cccc-123456abcdef.json
/etc/cloudflared/
```

2. Update the config file with new paths:

```
💻 TERMINAL

sudo nano /etc/cloudflared/config.yml
```

Change credentials-file path to:

```
💻 TERMINAL

credentials-file: /etc/cloudflared/12345678-aaaa-bbbb-cccc-
123456abcdef.json
```

3. Install the service:

```
💻 TERMINAL

sudo cloudflared service install
```

4. Start and enable the service:

```
💻 TERMINAL

sudo systemctl enable --now cloudflared
```

5. Check service status:

```
💻 TERMINAL

sudo systemctl status cloudflared
```

🍎 macOS Service Installation

Create a LaunchDaemon configuration:

TERMINAL

```
sudo nano /Library/LaunchDaemons/com.cloudflare.cloudflared.plist
```

Add the following (adjust paths as needed):

CONFIG.YML

```
<?xml version="1.0" encoding="UTF-8"?> <!DOCTYPE plist PUBLIC
"-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd"> <plist
version="1.0"> <dict> <key>Label</key>
<string>com.cloudflare.cloudflared</string>
<key>ProgramArguments</key> <array>
<string>/usr/local/bin/cloudflared</string> <string>tunnel</string>
<string>run</string> <string>my-home-server</string> </array>
<key>RunAtLoad</key> <true/> <key>StandardOutPath</key>
<string>/usr/local/var/log/cloudflared.log</string>
<key>StandardErrorPath</key>
<string>/usr/local/var/log/cloudflared.err</string> </dict> </plist>
```

Load the service:

TERMINAL

```
sudo launchctl load
/Library/LaunchDaemons/com.cloudflare.cloudflared.plist
```

➕ Advanced: Managing Multiple Services

You can expose multiple local services through the same tunnel. Here's an example configuration:

⚙️ CONFIG.YML

```
ingress: - hostname: photos.example.com service: http://localhost:3000 - hostname: nas.example.com service: http://localhost:5000 - hostname: rdp.example.com service: tcp://localhost:3389 - hostname: game.example.com service: http://localhost:25565 - hostname: ssh.example.com service: ssh://localhost:22 - hostname: git.example.com service: http://localhost:8080 - service: http_status:404
```

After updating config.yml, run DNS routing for each hostname:

💻 TERMINAL

```
cloudflared tunnel route dns my-home-server photos.example.com
cloudflared tunnel route dns my-home-server nas.example.com
cloudflared tunnel route dns my-home-server rdp.example.com
cloudflared tunnel route dns my-home-server game.example.com
cloudflared tunnel route dns my-home-server ssh.example.com
cloudflared tunnel route dns my-home-server git.example.com
```

📌 **PROTOCOL SUPPORT:**

- `http://` or `https://` - For web services (works in any browser)
- `tcp://` - For TCP services like RDP, database connections
- `ssh://` - For SSH connections
- `unix://` - For Unix domain sockets

Note: TCP services require cloudflared on the client side too. For web services (HTTP/HTTPS), any browser works.

⚡ Quick Test Method (No Domain Required)

If you want to quickly test cloudflared without setting up a domain:

💻 TERMINAL

```
cloudflared tunnel --url http://localhost:8080
```

This gives you a random URL like: `https://random-name.trycloudflare.com`

> ⏱️ **LIMITATIONS:**
> • URL is valid for a few hours only
> • No custom domain support
> • Great for testing, demos, or temporary sharing
> • Bandwidth may be limited on free tier

This is perfect for:

- Quickly testing if cloudflared works with your service
- Sharing a local development site with a client temporarily
- Testing before committing to a domain setup

🔧 Troubleshooting Common Issues

Issue	Likely Cause	Solution
403 Forbidden	DNS route not set up or incorrect	Run `cloudflared tunnel route dns` again
404 Not Found	Hostname in URL doesn't match config	Check hostname in config.yml matches exactly
ERR_CONNECTION_REFUSED	Local service not running or wrong port	Verify service is running and port is correct
502 Bad Gateway	Service not responding	Check service logs, restart if needed
Connection timeout	Network issues or protocol problems	Try `--protocol h2mux` or `--protocol http2`
Tunnel shows "Inactive"	cloudflared not running	Check service status, restart service
Certificate errors	Auth token expired or missing	Re-run `cloudflared tunnel login`
DNS not resolving	Propagation delay or wrong nameservers	Wait, check nameservers at registrar
High latency	Geographic distance to Cloudflare edge	Try different protocol, check your internet
Service stops after reboot	Not installed as service	Follow Step 8 to install as service

Diagnostic Commands:

💻 TERMINAL

```
# Check tunnel status
cloudflared tunnel list

# Check tunnel logs (when running manually)
cloudflared tunnel run my-home-server --loglevel debug

# Verify DNS resolution
nslookup home.example.com

# Check local service
curl http://localhost:8080

# Test cloudflared connectivity
cloudflared tunnel ping my-home-server
```

📌 Important Notes and Best Practices

🔒 Security

- No inbound ports need to be open on your router
- Only outbound connections from cloudflared to Cloudflare
- Your origin server remains hidden and protected
- All traffic is encrypted end-to-end
- Keep your credentials file secure

🌐 SSL/HTTPS

- Cloudflare automatically provides SSL certificates
- Free Universal SSL for all domains
- Automatic HTTPS redirection
- HSTS support available
- Certificate auto-renewal

⚡ Performance

- Free tier: Unlimited bandwidth
- Global CDN caching
- Argo Smart Routing (paid)
- Variable speeds based on location
- Good for personal/family use

🔄 Reliability

- Auto-reconnects after internet drops
- Load balancing across multiple tunnels
- Failover support
- 99.9%+ uptime on Cloudflare network
- Regular automatic updates

💾 Backup & Maintenance

- Save credentials.json securely
- Backup config.yml
- Document your tunnel ID

📊 Monitoring

- Cloudflare Dashboard → Zero Trust → Networks → Tunnels
- View tunnel status and metrics
- Access logs available
- Set up alerts (paid)

- Regular updates: `cloudflared update`
- Monitor via Cloudflare Dashboard

- Real-time connection status

Use Cases

- 🏠 **Home Server:** Access your NAS, media server, or home automation from anywhere
- 💻 **Development:** Share local development sites with clients or teammates
- 🎮 **Game Servers:** Host Minecraft, Valheim, or other game servers
- 📁 **File Sharing:** Secure access to personal cloud storage
- 🖥️ **Remote Desktop:** RDP or VNC access without VPN
- 🤖 **IoT/Home Assistant:** Remote access to smart home dashboards

🎉 **CONGRATULATIONS! Your local server is now accessible from anywhere via your custom domain!**

📚 Additional Resources

- Cloudflare Tunnel Documentation: **https://developers.cloudflare.com/cloudflare-one/connections/connect-apps/**
- Cloudflare Dashboard: **https://dash.cloudflare.com/**
- GitHub Repository: **https://github.com/cloudflare/cloudflared**
- Community Forum: **https://community.cloudflare.com/**

📄 How to Save as PDF:

Browser: File → Print → Save as PDF (or Ctrl+P / Cmd+P → Select "Save as PDF")

Chrome/Edge: Print → Destination → "Save as PDF"

Firefox: Print → General → "Save as PDF"

Safari: File → Export as PDF

⚠️ Make sure to enable "Background Graphics" in print settings for colored boxes.