# Self-Hosted DNS Upstream Behavior Comparison Matrix: Pi-hole vs AdGuard Home vs Technitium DNS vs Control D

**TechRounder PDF Edition**

Live article:
https://www.techrounder.com/dns-networking/self-hosted-dns-upstream-behavior-comparison-matrix-pi-hole-vs-adguard-home-vs-technitium-dns-vs-control-d/

By Vipin PG | Published April 7, 2026 | Updated April 7, 2026 | Format: Comparison | 7 min read

## Bottom line

Selecting the right DNS tool depends less on ad-blocking features and more on how the resolver manages upstream behavior, such as latency-based racing, strict failover, and domain-specific routing. While AdGuard Home and Pi-hole are ideal for simple network-wide filtering, Technitium DNS offers full server capabilities for complex homelabs, and Control D provides a managed service approach for roaming policy control.

### Key points

- Upstream Logic is Key: The most critical distinction between DNS tools is how they handle multiple upstreams-whether they query all simultaneously (fastest wins), follow a strict sequential order, or use domain-specific routing.
- Pi-hole: Best for familiar UI and light local needs; built on ‘dnsmasq‘ , it requires manual tuning for strict ordered failover and is optimized for simple filtering.
- AdGuard Home: A modern DNS proxy that natively supports encrypted upstreams and domain-specific forwarding without complex configuration files.
- Technitium DNS: A robust choice for advanced users needing a full DNS server with support for zones, wildcards, and conditional forwarding before ad-blocking.
- Control D: A managed service ideal for roaming clients and centralized policy management, though it functions differently than self-hosted authoritative DNS stacks.
- Failure Mode Planning: Users should choose a tool based on their specific requirement for resilience: whether they prefer the fastest response possible or a leak-proof, strict failover chain.

The part that trips people up is not ad blocking. It is upstream behavior. You add two resolvers, assume one is primary and the other is backup, then discover your DNS stack is racing them, preferring the lowest-latency answer, or leaking to some default resolver when the preferred path breaks.

I keep seeing the same pattern in homelabs: Pi-hole or AdGuard Home handles blocking, another box handles local zones, and then somebody tries to turn the whole thing into a predictable failover chain. That is where product marketing stops being useful. The question is no longer "which DNS tool is best?" It is "what exactly will this thing do when upstream number one is slow, down, filtered, or returns a result you did not want?"

## What matters in upstream behavior

Most people only need three answers. Does the resolver query one upstream at a time or several at once? Can it do real ordered failover instead of fastest-response wins? And can it keep local DNS responsibilities separate from filtering and external recursion without turning your network into a science project?

If your network is small and you just want clean device-wide blocking, the answer can be simple. If you run multiple VLANs, conditional forwarding, local zones, split DNS, Active Directory, or a second resolver like Control D upstream, upstream behavior becomes the feature. That is the line where a simple blocker stops being enough and a real DNS server starts making sense.

## The comparison matrix that actually matters

The matrix below focuses on resolver behavior, not generic feature bragging. That is the layer that decides whether your setup feels stable or weird.

Data last verified: April 2026

Product: Pi-hole | What it really is: Network-wide blocker built on FTL/dnsmasq | Default multi-upstream behavior: Not true primary/secondary by default; commonly treated as querying all configured upstreams unless dnsmasq behavior is customized | Can do strict ordered failover?: Yes, but usually through dnsmasq-level tuning such as ‘strict-order‘, not a clean first-class workflow | Domain-specific upstream routing: Possible through dnsmasq-style custom lines, but not its strongest path | Best use case: Simple blocker with familiar UI and light local DNS needs

Product: AdGuard Home | What it really is: DNS proxy with built-in filtering and modern encrypted upstream support | Default multi-upstream behavior: Supports multiple upstreams and domain-specific upstreams; can use parallel requests or fastest-IP logic depending on mode | Can do strict ordered failover?: Partly, but the product is more opinionated around smart upstream handling than strict old-school failover chains | Domain-specific upstream routing: Yes, cleanly exposed in config | Best use case: Users who want blocking plus modern upstream handling without extra sidecars

Product: Technitium DNS | What it really is: Full DNS server with filtering, zones, recursion, and forwarders | Default multi-upstream behavior: Latency-aware forwarding and concurrency are central to how it works | Can do strict ordered failover?: Yes, and newer builds explicitly added the option to disable concurrent forwarding for sequential support | Domain-specific upstream routing: Yes, especially strong for conditional forwarding and broader DNS design | Best use case: Homelabs that need real DNS features first and blocking second

Product: Control D | What it really is: Managed DNS service; ‘ctrld‘ adds local proxy capability | Default multi-upstream behavior: Primarily service-led policy routing rather than local authoritative DNS design | Can do strict ordered failover?: Possible in proxy-driven designs, but not the same thing as owning your whole DNS stack | Domain-specific upstream routing: Yes, through policy and upstream rules in ‘ctrld‘ setups | Best use case: Roaming clients, managed policy, and people who want cloud control with minimal maintenance

## Pi-hole: still good, but upstream logic is not the reason to choose it

Pi-hole remains solid when what you actually want is network-wide filtering with a clean dashboard. Under the hood, though, Pi-hole FTL is still built around dnsmasq. That matters because upstream behavior inherits a lot of dnsmasq logic and dnsmasq-style tuning rather than exposing everything as an obvious UI choice. If you want the broader blocker comparison, the earlier Pi-hole vs AdGuard guide is still useful background.

The practical consequence is simple. A lot of users expect "upstream 1, then upstream 2 if needed." Pi-hole does not naturally feel like that unless you deliberately tune it that way. The long-running Pi-hole community guidance around ‘strict-order‘ exists for a reason. If ordered failover is the core requirement, Pi-hole can be made to behave, but I would not call that its native strength.

That does not make Pi-hole bad. It means Pi-hole is easiest to recommend when the DNS server is mostly there to enforce blocklists and maybe answer a handful of local names. Once you start layering conditional forwarding, multi-site upstream selection, or mixed privacy resolvers, you are already pushing beyond the sweet spot.

## AdGuard Home: better when you want smarter upstream controls without building around dnsmasq

AdGuard Home is much clearer about being a DNS proxy that forwards queries to upstream servers. Its configuration supports multiple upstreams and selected-domain upstream definitions, which already makes it a cleaner fit for split behavior than many Pi-hole installs. That broader DNS-tool context also fits nicely with the DNS solutions overview on TechRounder.

The detail that matters is mode selection. AdGuard Home exposes "Parallel requests" and "Fastest IP address," but they are not additive. They are mutually exclusive. People often miss that and assume they can enable every speed feature at once. They cannot. You need to decide whether you want query racing or downstream IP selection behavior for multi-address answers.

In day-to-day use, AdGuard Home tends to feel more modern than Pi-hole for upstream handling. You get built-in support for encrypted upstream styles and domain-specific forwarding without immediately dropping into dnsmasq syntax. That is why it lands well for users who want one box to do blocking, encrypted upstreams, and enough routing logic to stay practical.

## Technitium DNS: the one to pick when DNS design matters more than ad blocking

Technitium behaves like a real DNS server first. That sounds like a small distinction until you actually need zones, wildcard records, conditional forwarders, recursion choices, or forwarding behavior that matches how you think. In community discussions, that is exactly why people keep moving to it: not because Pi-hole failed, but because their network got more complicated.

Its forwarder model is built around concurrency and latency-based selection. That is great when you want fastest-valid-response behavior, especially across several good upstreams. It is not great if your mental model is old-school "resolver A is primary, resolver B is standby." Technitium eventually added the option to disable concurrent forwarding so sequential support is possible, which tells you the project understands that both models are valid depending on the network.

This is also where local DNS work becomes much easier. Wildcards, richer record handling, and more serious DNS plumbing are just more natural here. If your setup includes reverse proxies, service naming, and internal zones, you will eventually run into the reminder that DNS cannot map ports. Technitium helps on the name-resolution side, but it does not replace proper reverse-proxy design.

## Control D: excellent policy DNS, different job description

Control D should not be forced into the same mental box as the other three. It is a managed DNS platform first. The optional ‘ctrld‘ daemon gives you a local DNS proxy you can run on a router or another device, which is useful when you want LAN clients to use secure DNS, policy routing, or split behavior without configuring every client by hand.

The useful part is convenience. The tradeoff is that you are not building a full self-hosted DNS server in the same sense as Technitium, and not even in the same simple-filter sense as Pi-hole or AdGuard Home. You are consuming a service and optionally placing a smart proxy in front of it. For roaming devices and low-maintenance policy control, that is a very good trade.

The one setting I would not ignore is upstream-failure handling in ‘ctrld‘. Depending on platform and defaults, it can forward to the network’s default resolver when the remote upstream fails. For some users that is a resilience feature. For others it is exactly the kind of privacy leak they were trying to avoid in the first place. If you care about leak-free behavior, read that part twice before rollout.

## What I would deploy in the real world

If the job is "block ads and trackers on a home network with minimal fuss," AdGuard Home is usually the cleanest single-box answer. If the job is "I need a proper DNS server that also blocks," Technitium is the one I would reach for. Pi-hole still makes sense when you value its ecosystem and already know the dnsmasq edges. Control D makes sense when uptime, roaming support, and policy control matter more than owning every part of the resolver stack.

I would not choose between these tools by dashboard screenshots or blocklist counts. I would choose by failure mode. When one upstream is slow, when one resolver returns a filtered answer, when WAN latency spikes, or when a local zone must always win, the product’s upstream model decides whether the network feels boring or brittle.

## Where to look next

Before you switch anything, map your actual requirement in one sentence: fastest-answer routing, strict failover, local zones, roaming clients, or low-maintenance filtering. Once that is clear, the product choice gets easy. After that, apply the change carefully on clients or the router, and verify with query logs instead of assumptions.