The Hidden Lock-In of Encrypted DNS: Your Resolver Is Portable, Your Security Policy Isn’t

TechRounder PDF Edition

Live article:
https://www.techrounder.com/technology/the-hidden-lock-in-of-encrypted-dns-your-resolver-is-portable-your-security-policy-isnt/

By Vipin PG | Published September 9, 2026 | Updated September 9, 2026 | Format: Deep Dive | 8 min read

In brief

Changing DNS providers looks simple on paper. Replace a resolver address, switch a DNS-over-HTTPS endpoint, update a DNS-over-TLS hostname, and traffic starts going somewhere else.

Changing DNS providers looks simple on paper. Replace a resolver address, switch a DNS-over-HTTPS endpoint, update a DNS-over-TLS hostname, and traffic starts going somewhere else.

That describes the resolver connection. It says little about everything configured around it.

Modern DNS services can block malware, enforce content restrictions, identify individual devices, apply different rules to different users, rewrite responses, store analytics in selected regions, handle local DNS names, schedule access restrictions, and decide which rule wins when several rules match the same domain.

Those settings can take years to accumulate.

Moving them to another provider can require a manual reconstruction of the original policy, even when both services support the same encrypted DNS standards.

The DNS industry has made good progress on resolver interoperability. Policy portability has received far less attention.

Encrypted DNS already has common standards

The transport side of encrypted DNS is well defined.

DNS-over-TLS was standardized through RFC 7858. DNS-over-HTTPS followed with RFC 8484, which defines how DNS queries and responses are carried through HTTPS. DNS-over-QUIC arrived later through RFC 9250.

RFC 8484 focuses on the exchange between a DNS client and a DoH server. A DNS query-response pair is mapped to an HTTP exchange, giving clients a standard way to communicate with compatible resolvers over HTTPS.

The standards have continued beyond transport.

RFC 9463 defines DHCP and IPv6 Router Advertisement options that allow devices to discover encrypted DNS resolvers supporting protocols such as DoH, DoT and DoQ.

There is an important boundary in that RFC. Resolver selection policy and other policies are explicitly outside its scope.

That boundary makes sense for an Internet protocol specification. The standards need to describe how systems communicate, while commercial services remain free to build different security products around DNS.

The result is a split that becomes obvious during migration.

A client can understand how to reach another encrypted resolver. It has no standard way to tell that resolver what the previous provider’s security configuration meant.

DNS services now carry much more than resolver settings

The word "resolver" can understate what these services do.

NIST’s Secure Domain Name System Deployment Guide, finalized in March 2026, describes DNS as part of an organization’s security architecture. The guidance notes that DNS can function as a policy enforcement point and can provide information used in security decisions.

Consumer and business DNS platforms have moved in the same direction.

NextDNS exposes multiple security controls, including Google Safe Browsing, cryptojacking protection, DNS rebinding protection, IDN homograph protection, typosquatting detection, Domain Generation Algorithm detection, Newly Registered Domains and Parked Domains. It also provides parental controls, allow and deny lists, rewrites, logging options and separate configurations.

Control D uses a different model. A Profile contains Filters, Services, Custom Rules and Profile Options. That Profile is then applied to an Endpoint. Its documentation also defines an order of precedence: Custom Rules are evaluated above Services, followed by Filters and then the Default Rule.

Cloudflare Gateway expresses DNS policy through logical expressions. Administrators choose selectors, operators and values, then attach actions such as Allow, Block or Override. The policy can use information such as domains, source IPs and user identity.

DNSFilter has its own hierarchy. Its documentation says allow lists take precedence, followed by block lists and filtering categories, with additional priority rules depending on where policies are applied.

AdGuard DNS supports blocklists, user rules, security controls and parental controls. Its filtering syntax can also express more advanced rules than a plain list of domains.

Each provider is describing DNS policy in its own language.

That difference becomes significant when a user wants to leave.

A domain list is only one part of the configuration

The easiest DNS settings to move are usually explicit domains.

A list containing domains that should always be blocked or allowed can often be copied, cleaned up and imported elsewhere. Several DNS platforms provide some form of custom domain rule or list support.

This creates the impression that DNS configuration is already portable.

A mature setup contains more than domains.

Consider a network with separate policies for adults and children. One device has an exception. Social media is restricted during certain hours. Internal hostnames need local resolution. A few DNS responses are rewritten. Logs have a defined retention period. Analytics need to stay within an approved region. Several security protections have been enabled individually.

The domain list captures almost none of that.

Privacy settings provide a simple example.

NextDNS allows users to control how long logs are retained or disable logging, and its current service page lists the United States, European Union, United Kingdom and Switzerland as storage choices.

Those choices are part of the user's DNS policy even though they have nothing to do with whether 'example.com' is allowed.

A useful migration therefore has to preserve more than domains. It needs to preserve intent.

Control D's NextDNS migration guide makes the problem visible

Control D publishes a guide specifically for users moving from NextDNS.

The guide describes most migrations as a "concept-mapping exercise."

Users are told to recreate NextDNS configurations as Control D Profiles, map broader controls to Filters and Services, transfer allow and deny entries into Custom Rules, rebuild rewrites and local-domain behavior, assign Profiles to the correct Endpoints, and validate the results before moving everyone to the new resolver.

The guide also warns users not to assume that similarly named categories or third-party blocklists behave identically across providers.

That is a useful warning because configuration names alone cannot establish equivalence.

One service might expose typosquatting as its own switch. Another may include related detection inside a broader phishing system. A category called "Social" can have a different definition elsewhere. An allow rule may override a security filter in one system while another system applies a different precedence model.

A migration can therefore reproduce the visible list of settings and still change the effective policy.

This is where DNS portability becomes a semantic problem.

Security controls are especially difficult to translate

NextDNS currently presents several protection mechanisms as individually named settings, including cryptojacking, DNS rebinding, IDN homographs, typosquatting, DGAs and Newly Registered Domains.

Another provider can cover some of the same threats without presenting them as separate switches.

That creates a migration problem with several possible outcomes.

A setting may have a close equivalent. It may be included inside a broader security category. The destination service may approach the threat differently. The feature may also have no direct counterpart.

An automated importer would need to know the difference.

Copying a Boolean value such as ‘typosquatting=true‘ is useful only when both platforms agree on what that feature means and how it is enforced.

Security products rarely provide that level of semantic equivalence.

Threat intelligence sources differ. Detection methods differ. Category definitions change. Some providers use third-party lists alongside internal detection. Others combine several mechanisms behind one security control.

A technically successful migration can therefore leave an administrator with a different security posture than expected.

Rule priority can change the result without changing the rule

Policy precedence adds another complication.

Suppose the same domain appears in an allow rule, a category block and a security filter.

DNSFilter documents an explicit hierarchy in which allow lists take priority over other filtering decisions.

Control D defines a different structure around Custom Rules, Services, Filters and the Default Rule.

Cloudflare Gateway uses ordered policies and logical expressions, with precedence influencing which rule is applied.

The domain can appear in every migrated rule and still produce a different result if the destination evaluates those rules differently.

This is one reason a policy export needs more information than a collection of domains.

It needs the relationships between those rules.

DNS policy could use a portable description

A practical portability format would not require every DNS provider to build identical features.

It could describe what the administrator expects and allow the destination service to report how closely it can reproduce that behavior.

A portable DNS policy could include:

- explicit allow and block rules;
- security protections that the administrator expects to remain active;
- content categories and application restrictions;
- schedules;
- device and user groups;
- DNS rewrites and local namespaces;
- upstream-routing requirements;
- DNSSEC expectations;
- caching or TTL preferences where relevant;
- logging and analytics settings;
- retention requirements;
- data-residency preferences;
- rule priority and exceptions.

The import process could then classify each item.

Some settings would be exact matches. Others would be approximate matches. A few might require manual configuration or have no equivalent.

That distinction would make migrations easier to audit.

An administrator could see that 80 domain rules imported correctly while two security controls were absorbed into a broader destination filter and one privacy setting could not be reproduced.

Current migration guides already perform part of this work for humans. The missing piece is a common machine-readable representation.

Existing rule formats solve only part of the problem

DNS already has examples of policy-like formats.

AdGuard’s filtering syntax, for example, can express simple domains, exceptions, pattern matching and DNS-specific modifiers.

Those formats are useful for filtering rules. They do not describe an entire managed DNS account.

A full account can contain device identity, policy assignment, schedules, analytics preferences, category choices, threat protections, routing behavior and privacy requirements.

Moving the rule syntax preserves one layer.

Moving the policy requires considerably more context.

APIs do not automatically make policy portable

Modern DNS platforms increasingly expose APIs, structured rules and infrastructure automation.

Cloudflare Gateway policies, for example, use their own expression syntax based on fields, operators and values.

That makes configuration reproducible inside Cloudflare.

Reproducibility and cross-provider portability are separate problems.

A configuration file can describe a provider’s objects perfectly while remaining unusable by another provider whose concepts are different.

The same limitation applies to scripts, API clients and infrastructure-as-code definitions. Automation removes repeated manual work within a known system. Translation still has to happen when the policy model changes.

Better migration tooling can arrive before a standard

DNS providers do not need to wait for an industry-wide specification to improve portability.

Migration tools could start with a capability report.

After importing a configuration, the destination service could identify settings that transferred directly, controls that were mapped to broader categories, rules whose precedence changed, and features that need manual review.

Staged migration is another useful approach.

Control D’s current NextDNS migration documentation recommends testing with a smaller group before switching production DNS and keeping the previous configuration available for rollback. Its ‘ctrld‘ NextDNS mode can also keep forwarding requests to the existing NextDNS profile while the local deployment is tested.

A more advanced system could compare decisions from two policies before cutover.

Known domains could be tested against both configurations. Differences could then be presented to the administrator before the resolver changes for everyone.

For a security service, "import completed" is a weak success criterion.

The useful question is whether the new policy produces the expected decisions.

DNS portability has moved beyond the resolver address

Encrypted DNS standards have made it easier for clients and resolvers to communicate securely. Newer standards can even help devices discover encrypted resolvers automatically.

The configuration sitting above that connection has become much larger.

A user’s DNS setup can now contain security controls, device identities, privacy settings, application restrictions, category policies, exceptions and local-network behavior.

Current migration documentation shows how much of that configuration still has to be interpreted manually.

That is the hidden lock-in.

It does not require a closed protocol or a resolver that refuses to let users leave. It appears when years of security decisions have been expressed in one provider’s vocabulary and the next provider speaks a different one.

If you manage a filtered DNS setup, I would document that policy separately from the resolver configuration. Record the protections you expect, the important exceptions, rule priority, device assignments and privacy requirements. Until DNS services gain a common way to describe those settings, that record may be the most portable version of your DNS policy.