



DNS, Explained

What you are actually editing
when you edit a DNS record

A plain-English guide for people who run businesses, not mail servers.
The foundation for the other five in this series — and the one that
explains why a change you made an hour ago still has not appeared.

01

What DNS actually is

Not a directory. A chain of referrals.

DNS turns names into instructions. Not only addresses — the same system carries where your mail should go, which senders you have authorised, and every other record the rest of this series asks you to publish.

The part worth understanding is that no single machine holds the answers. Nobody has the list. Each step in a lookup only knows enough to say who to ask next.

Nobody holds the whole directory. Each step just says who to ask next.

The root servers

Do not know yourdomain.com — but know who runs .com.

The .com servers

Do not know your records either — but know which nameservers you chose.

Your nameservers

These hold the answer. This is the only step you control.

Doing the asking is a resolver — usually run by your internet provider or by a service like Cloudflare or Google. It remembers what it learns, which is where almost every DNS surprise comes from.

Figure 1 — Three questions, each narrower than the last.

Why this shape matters to you

Because the last step is the only one you control, and because every step in between is allowed to remember what it was told. Those two facts explain almost everything that seems strange about DNS.

Who controls your domain

Three things people routinely mistake for each other.

Three different things, often bought from three different companies.

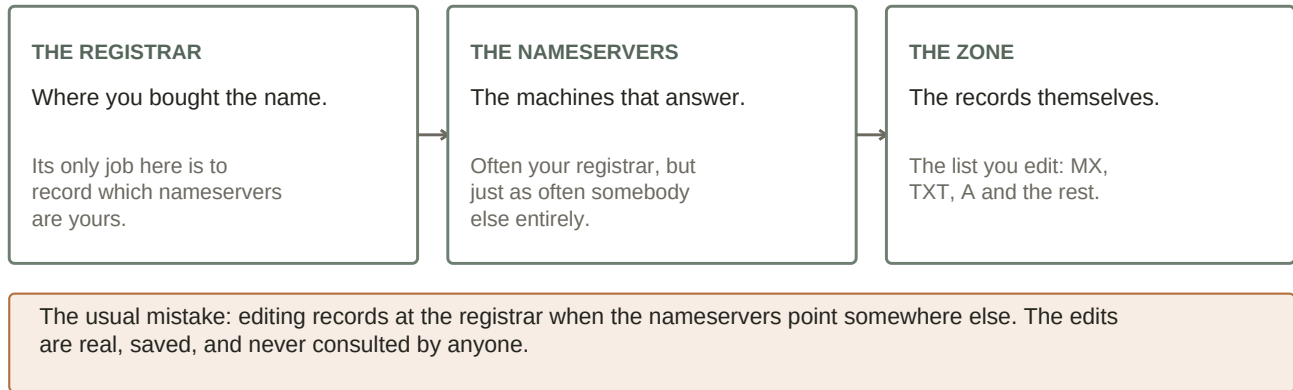


Figure 2 — Registrar, nameservers, zone.

If you take one practical thing from this page: before editing any record, confirm *where* your domain's nameservers actually point. Plenty of people have spent an afternoon editing a perfectly correct record in a control panel nobody was reading.

How to check

Any DNS lookup tool will report your domain's nameservers, including ReaperCloud's. Whatever they say is authoritative is where your records have to live — regardless of who sold you the name.

What a record looks like

A name, a type, a value, and a number.

Every DNS record has the same four parts. Most control panels show them as four boxes, sometimes with different labels.

yourdomain.com.	3600	MX	10 mail.yourdomain.com.
name	TTL	type	value

The handful you will actually meet.

A	Points a name at a server address.
AAAA	The same, for newer addresses.
MX	Where mail for this domain should be delivered.
TXT	Free text. SPF, DKIM and DMARC all live here.
CNAME	Says "this name is really that name". Has rules.
NS	Which nameservers are authoritative.
SOA	Housekeeping for the zone. One per zone.

Every record also carries a name and a TTL. Records sharing a name and type must all carry the same TTL.

Figure 3 — The types you will actually meet.

The one record type with rules

CNAME is useful, and it is fussy.

A CNAME says "this name is really another name — go and look there instead". It saves repeating yourself when something moves. It also carries two restrictions that are easy to trip over because nothing warns you at the moment you save the record.

Two rules, both of which people meet the hard way.

A name with a CNAME can have nothing else

Not an MX record, not a TXT record, nothing. And since the top of your zone must carry NS and SOA records, that is why you cannot put a CNAME on `yourdomain.com` itself — only on names beneath it.

An MX or NS record must not point at a CNAME

The target of an MX record has to be a real name with a real address, not an alias. Mail servers are entitled to refuse to follow the alias, and some do.

Some DNS hosts offer a way around the first rule at the top of the zone, under names like ALIAS or ANAME. Those are the host doing the work for you, not DNS changing its mind.

Figure 4 — Both rules, and where they bite.

The first rule has one consequence worth stating plainly: you cannot point `yourdomain.com` itself at another name with a CNAME. The top of a zone must carry its own housekeeping records, and a CNAME is not allowed to sit alongside them.

The number nobody explains

TTL is a ceiling, not a schedule.

The TTL — time to live — is how long anything that asks for this record is permitted to remember the answer before asking again. It is measured in seconds, so 3600 is an hour and 86400 is a day.

What it is not

It is not how long your change takes, and it is not a promise in either direction. A resolver may throw the answer away sooner if it wants the space, and many refuse to honour very long values at all, quietly substituting a ceiling of their own.

One practical rule follows from this, and it is the single most useful operational habit in this guide: because a lower TTL only takes effect once the *old* one has expired, it has to be lowered in advance.

Lowering a TTL is itself subject to the old TTL. Do it early.

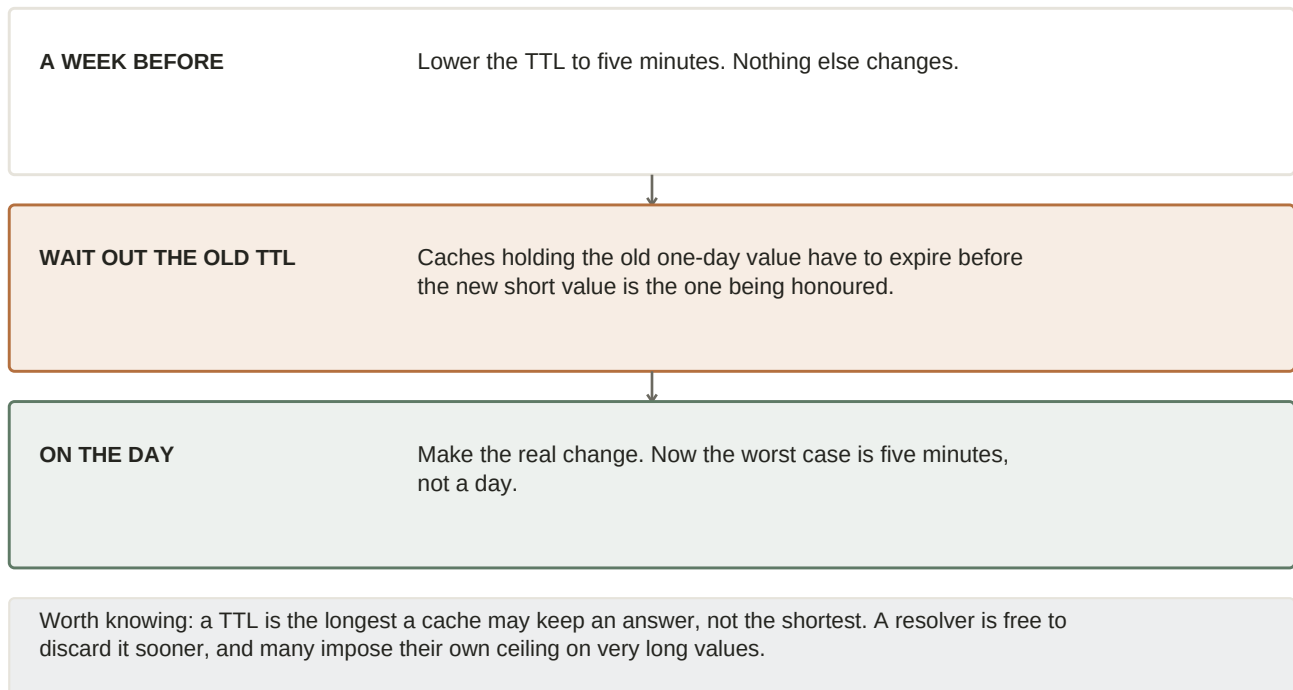


Figure 5 — Lower it before you need it low.

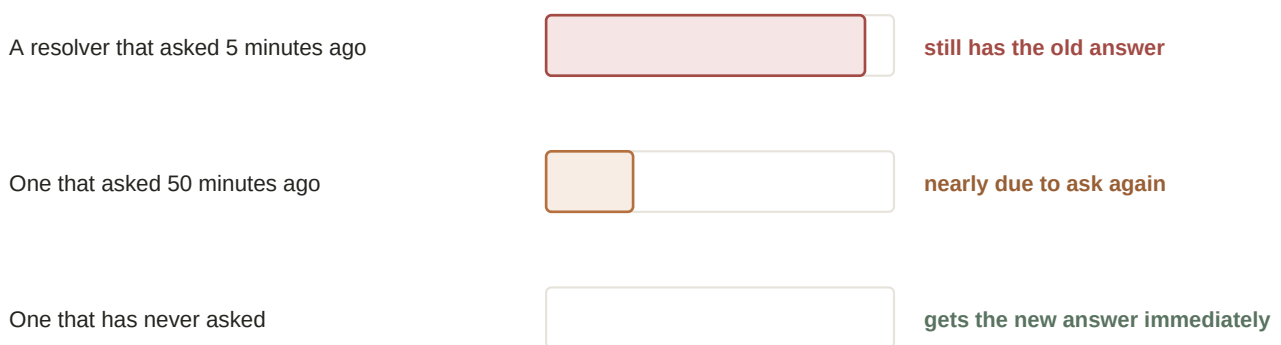
Nothing propagates

The most misleading word in DNS.

People say a DNS change is "propagating", which sounds like an update spreading outward from your zone to the rest of the world. That is not what happens, and believing it makes the timing impossible to reason about.

Nothing is sent anywhere. Your nameservers simply start giving a different answer to anyone who asks. Everything that already asked is still holding the old answer until its own clock runs out.

You changed a record with a one-hour TTL. Here is what three resolvers see.



Nothing was sent anywhere. No update travelled outward. Each resolver simply forgot at its own moment and asked again — which is why a change is live for you and not yet for a colleague.

Figure 6 — Three resolvers, three different clocks.

Why the change is live for you and not for them

Because you and they asked at different moments, so their countdowns started at different moments. There is no queue, no rollout, and nothing to check the progress of — only caches that have not yet forgotten.

Why a brand-new record seems not to work

The part almost nobody is told.

Here is the situation that generates more confusion than anything else in this series. You publish a DMARC record. You check it with an online tool. The tool says there is no DMARC record. You check your control panel — the record is definitely there.

The answer "there is no such record" gets remembered too.

BEFORE YOU ADD THE RECORD

Something asks for your DMARC record. It does not exist yet, so the answer comes back "no such record" — and that answer is cached, exactly like a real one would be.

AFTER YOU ADD IT

The record is live on your nameservers. But anything holding that cached "no such record" will keep believing it until that expires — which has nothing to do with the TTL on the record you just created.

How long is set by the last number in your zone's SOA record, often an hour or a day. If a record you have just published appears not to work, this is usually the reason, and waiting is usually the fix.

Figure 7 — "It does not exist" is an answer, and answers get cached.

Nothing is wrong. Something asked for that record before you created it, got told there was none, and remembered. How long it remembers is governed by a field in your zone's housekeeping record, which is frequently set to an hour and sometimes to a day.

What to do about it

Wait, mostly. If you want to confirm the record really is live, query your own nameservers directly rather than going through a resolver that may be holding a stale "no" — any decent lookup tool can do this, including ReaperCloud's.

The ways DNS usually breaks

In rough order of how often they happen.

- **Editing records in the wrong place.** The nameservers point at one provider and the records are being typed into another. See Section 02.
- **Expecting a change immediately.** Caches are still holding the old answer, and no amount of refreshing changes that. See Section 06.
- **A TXT record split wrongly.** Long values have to be broken into chunks, and most control panels do it for you — but pasting a value that already contains quote marks can produce something subtly wrong that still saves without complaint.
- **A trailing dot, present or missing.** In most control panels `mail.yourdomain.com` and `mail.yourdomain.com.` mean different things, and some panels silently append your domain to anything without the dot.
- **Two records that should be one.** A domain must have exactly one SPF record and one DMARC record. Two of either is not twice as good; it is an error, and the usual result is that neither works.

The useful instinct

When something looks wrong, ask what your nameservers are actually serving right now, rather than what your control panel shows. Those are different questions, and only the first one matters to anyone else.

The records that carry your email

A map, not an explanation.

Everything the rest of this series asks you to publish.

<code>yourdomain.com</code>	MX	where your mail arrives
<code>yourdomain.com</code>	TXT	your SPF record
<code>sel._domainkey.yourdomain.com</code>	TXT	a DKIM key
<code>_dmarc.yourdomain.com</code>	TXT	your DMARC policy
<code>_mta-sts.yourdomain.com</code>	TXT	points at your MTA-STS policy
<code>_smtp._tls.yourdomain.com</code>	TXT	where to send TLS reports
<code>_25._tcp.mx.yourdomain.com</code>	TLSA	your DANE fingerprint

The names beginning with an underscore are ordinary DNS names. The underscore is a convention that keeps them from colliding with the name of a real machine.

Figure 8 — Everything the rest of the series asks you to publish.

Each of these has its own guide in this series, and this page deliberately does not repeat them. What is worth noticing is that they are all ordinary DNS records — the same four parts as anything else, published in the same zone, subject to the same caching.

The odd-looking names beginning with an underscore are not a special mechanism. They are just names, kept out of the way of anything that might one day be a real machine.

A short checklist

Before you change anything, and after.

Before you change anything

- **Confirm where your nameservers point.** That is where the records have to be edited.
- **Lower the TTL a week ahead** if you are planning something you may need to undo quickly.
- **Write down what the record says now**, so you can put it back exactly.

After you change something

- **Query your own nameservers directly** to confirm the change is live at the source.
- **Then wait.** Anyone who asked before your change will keep the old answer until their own countdown expires.
- **If a new record seems missing, wait longer.** The "no such record" answer was cached too, and that is governed by a different number. See Section 07.

Check your own domain

ReaperCloud's DNS tool shows what your nameservers are actually serving, reports your TTLs, and finds the underscore records that ordinary lookups miss. `reapercloud.com/dns.php`, or `reapercloud.com/report.php` for the whole email picture at once.

This document describes DNS as specified in RFC 1034 and RFC 1035, with the clarifications of RFC 2181 and the negative-caching rules of RFC 2308. Terminology follows RFC 9499. It is an explanation, not a specification: where they differ, the RFCs are correct and this is simplified.