anoBytes | TECHNOLOGY SOLUTIONS

DNS security records cheat sheet

The DNS records that secure a domain: what each one does, what can go wrong, and how to check it is working.

MAIL AUTHENTICATION · TRANSPORT ENCRYPTION · CERTIFICATE CONTROL · DNS INTEGRITY

General educational guidance, not affiliated with or endorsed by any vendor named. References follow the RFCs named inline and NIST SP 800-81r3 for DNSSEC. Records shown use example.com; substitute your domain. Published 2026-08-24.

WHAT AN UNPROTECTED DOMAIN ALLOWS

By default, anyone can send mail with your domain in the From line (spoofing), and most servers deliver it. Any certificate authority can issue a certificate for the domain. And a DNS answer for it can be forged, because plain DNS is unsigned. The first three rows below are email authentication; the rest cover transport, certificates, and DNS itself.

RECORD	LIVES AT	BLOCKS
SPF (TXT)	apex	mail from unauthorized servers
DKIM (TXT)	<sel>._domainkey	tampered or unsigned mail
DMARC (TXT)	_dmarc	delivery when both checks fail
MTA-STS	_mta-sts + file	inbound plaintext fallback
TLS-RPT (TXT)	_smtp._tls	silent TLS failures
DNSSEC	zone + DS at parent	forged DNS answers
CAA	apex	certificates from unlisted CAs

SPF: WHO MAY SEND AS THE DOMAIN (RFC 7208)

A TXT record at the apex (the bare domain, example.com with nothing in front) listing the servers allowed to send as the domain; receivers check the connecting IP against it.

```
"v=spf1 include:spf.provider.example ~all"
# include: your mail provider's server list; ~all: soft-fail the rest
```

Checking an SPF record may use at most 10 DNS lookups (each `include`, `a`, `mx`, `ptr`, `exists`, and `redirect` counts); the 11th returns a permanent error (permerror) and mail starts failing authentication. Count the lookups before adding another mail service. With DMARC in place, `~all` is the usual choice over the hard-fail `-all`: DMARC does the enforcing, and a hard fail can drop legitimately forwarded mail before DMARC evaluates it.

```
dig +short example.com TXT | grep spf1
```

DKIM: A SIGNATURE ON EVERY MESSAGE (RFC 6376)

Your provider signs each outgoing message with a private key; the public key in this TXT record lets any receiver verify the message arrived unmodified. The selector part of the record name comes from your provider (`google` for Google Workspace, `selector1` and `selector2` for Microsoft 365) and lets the provider move to a new key without breaking mail. Publish 2048-bit RSA keys (RFC 8301).

```
dig +short <selector>._domainkey.example.com TXT
# expect: v=DKIM1; k=rsa; p=<long base64 key>
```

Your provider's setup page gives the exact records to publish; re-run the check after DNS changes propagate.

DMARC: THE ENFORCEMENT POLICY (RFC 7489)

SPF and DKIM each pass or fail on their own; DMARC ties them to the From address the reader sees and tells receivers what to do when both fail: `p=none` takes no action, `p=quarantine` sends to spam, `p=reject` refuses delivery. Receivers also mail you aggregate reports of who is sending as your domain; they arrive as XML attachments meant for a parsing service, so plan to feed them to one rather than read them by hand.

```
"v=DMARC1; p=reject; rua=mailto:dmarc@example.com"
```

On a domain that already sends mail, roll out in stages: start at `p=none` and read the reports until every legitimate sender shows up as passing, then move to quarantine, then to reject. Leave the alignment and subdomain tags out; the defaults are right for most domains. Alignment checks that the domain SPF or DKIM approved matches the From domain, and the default counts a subdomain as a match; the strict setting (`adkim=s`) requires an exact match and breaks mail signed from a subdomain. Subdomains follow the main policy unless `sp=` sets a different one.

Reports can go to an address outside your domain only when that outside domain publishes an authorization record (RFC 7489 section 7.1); without it, the reports silently never arrive:

```
dig +short _dmarc.example.com TXT
# reports going to a vendor? check their authorization:
dig +short example.com._report._dmarc.<vendor> TXT
```

DEFENSIVE DOMAINS NEED DMARC TOO

Each domain needs its own DMARC record; the policy on your main domain covers only that domain and its subdomains. A parked domain with no mailboxes still gets spoofed, and with no policy published, that mail is delivered. On every domain that sends no mail, publish the records below; the null MX line belongs only when the domain receives no mail either:

```
_dmarc  TXT  "v=DMARC1; p=reject;"
@       TXT  "v=spf1 -all"
@       MX   0 .
# @ = the apex in most DNS dashboards; -all is right here:
# nothing legitimate is ever sent from a parked domain
# null MX (RFC 7505): accepts no mail. skip it where email
# forwarding exists; it cannot share a zone with real MX
```

TLS-RPT: THE FAILURE REPORTS (RFC 8460)

One TXT record asking senders for a daily JSON summary of their TLS sessions with your mail server, successes included. An MTA-STS policy in enforce mode refuses bad connections silently, so these reports are how an attack or a misconfiguration becomes visible. Publish it before the MTA-STS policy goes live.

```
_smtp._tls  TXT  "v=TLSRPTv1; rua=mailto:tlsrpt@example.com"
dig +short _smtp._tls.example.com TXT
```

Reports arrive about daily as JSON attachments, usually compressed (RFC 8460); a report with zero failures is the healthy reading.

MTA-STS: TLS REQUIRED INBOUND (RFC 8461)

SMTP encryption is opportunistic: the sender asks for STARTTLS and delivers in plaintext when the offer seems absent, and an attacker on the network path can strip the offer. MTA-STS publishes a policy that makes TLS mandatory: senders that honor it refuse the plaintext fallback, require a valid certificate, and deliver only to the MX hosts the policy names. Three pieces have to exist and stay consistent:

`_mta-sts TXT` the switch. Its **`id`** value tells senders the policy version; a changed id triggers a re-fetch.

the policy file the rules, served over HTTPS at **`https://mta-sts.<domain>/.well-known/mta-sts.txt`**

mta-sts subdomain the fixed hostname senders fetch from; it needs a valid certificate of its own.

```
version: STSv1
mode: enforce
mx: mail.example.com
mx: *.example.com
max_age: 1209600
# list every MX host your domain publishes
```

Senders cache the policy for **`max_age`** seconds (RFC 8461 expects weeks or greater) and re-check the id before bouncing any message, so a corrected policy with a new id repairs a mistake. **`mode: testing`** reports what enforce would have blocked; a domain whose mail is hosted at a major provider can go straight to enforce, since senders already deliver to that provider's servers under its own enforce policy.

Every edit to the policy file needs a new id in the TXT record, or senders keep the cached copy and never see the change. Update the policy before changing MX hosts: senders holding the old policy refuse a new MX it does not name. To retire MTA-STS cleanly, publish mode: none with a small max_age, bump the id, wait out the old lifetime, then remove the records (RFC 8461 section 8.3).

```
dig +short _mta-sts.example.com TXT
curl -s https://mta-sts.example.com/.well-known/mta-sts.txt
```

BIMI (a TXT record at **`default._bimi`**) shows your logo beside authenticated mail in some clients. It requires a DMARC policy at quarantine or reject and, for the major mailbox providers, a paid Verified Mark Certificate renewed annually, so most small domains skip it.

THE ROLLOUT ORDER

The first three steps build on each other; the rest can be added any time.

```
1  SPF + DKIM      publish; verify both pass
2  DMARC p=none    read reports; find forgotten senders
3  DMARC p=reject  via quarantine if flows are complex
4  TLS-RPT         the reporting channel, first
5  MTA-STS         then the TLS requirement
6  DNSSEC          sign the zone; publish the DS
7  CAA             list your CAs; add iodef
8  parked domains  the reject-all records above
```

TEST IT FROM OUTSIDE

Google Admin Toolbox Check MX (toolbox.googleapps.com/apps/checkmx) checks a domain's mail records without an account. UpGuard's free website security scan (upguard.com/webscan) grades the domain from outside. The definitive mail test: send a message to a mailbox you control elsewhere, open its headers, and read the Authentication-Results line; a healthy result shows three passes naming your domain:

```
Authentication-Results: ...
  spf=pass ... dkim=pass header.d=example.com
  dmarc=pass header.from=example.com
```

DNSSEC: SIGNED DNS ANSWERS

DNSSEC signs your DNS records so a resolver can verify every answer came from you and was not altered on the way (NIST SP 800-81r3). Without it, a forged answer can redirect mail or visitors, and every record above can itself be forged in transit. Enable signing at the DNS host, then publish the DS record it gives you at the registrar; that links your domain into the global chain of trust. Pick ECDSA P-256 (algorithm 13) where offered; its signatures are smaller than RSA's (SP 800-81r3).

```
dig example.com A +dnssec | grep flags
# "ad" among the flags = answers validate
dig +short example.com DS
# the DS at the parent completes the chain
```

Moving DNS hosts or registrars while the old DS record stands makes every validating resolver reject the whole domain: remove the DS first, wait out its TTL, then move.

CAA: WHO MAY ISSUE CERTIFICATES (RFC 8659)

CAA records name the certificate authorities allowed to issue certificates for the domain; any CA not on the list must refuse the request. **`issue`** covers named hosts, **`issuewild`** covers wildcards, and **`iodef`** gives CAs an address to report refused or suspicious requests.

```
dig +short example.com CAA
# 0 issue "letsencrypt.org"
# 0 iodef "mailto:security@example.com"
```

Before adding any service that will hold a certificate for your domain (a mail gateway, a second CDN, an app on a subdomain), put that service's CA on this list. Otherwise the certificate request fails at that service with an error that says nothing about CAA.

Check the records with dig rather than your DNS dashboard: some DNS hosts quietly add CAA entries for their own certificate partners, so the dashboard can show fewer records than the internet sees.

CAA blocks issuance from unlisted CAs; certificate transparency (CT) reports what was issued anyway. Every publicly trusted certificate lands in public logs: a CT monitoring service emails you when one appears for your domain, and crt.sh shows the history on demand.

THE CONTROLS OUTSIDE DNS

`registrar lock` blocks domain transfers until you unlock; **`whois`** should show **`clientTransferProhibited`**

`expiry + auto-renew` a lapsed domain takes the site and mail with it.

`2FA on registrar and DNS host` either account can redirect the domain's mail and traffic.

`recovery address off-domain` a recovery email on the domain itself stops working exactly when the domain is lost.

`no wildcard record` a wildcard (*.example.com) makes every subdomain resolve, invented ones included. If its target is ever a shared or abandoned service, whoever claims it controls a working site under your name, valid certificate included, and can phish from it (a subdomain takeover).

`zone transfers refused` **`dig example.com AXFR @ns1`** should fail; a successful transfer hands an attacker every name in your DNS.