DNSSEC 101 – A Practical Guide for Administrators

This guide is written for administrators who are **new to DNSSEC** but already familiar with basic DNS. It explains the DNS and delegation basics you need, how DNSSEC adds security, and how each PRO tab in DNSSEC Checker fits into the full "security story" of a domain. Throughout the guide you'll see example commands using `dig` (Linux/macOS) and `nslookup` (Windows) so you can verify what the app is showing.

Sections in this guide

1. DNS & Delegation Basics (Parent vs Child)

DNS is a **hierarchical phone book** for the Internet. Names are organized in a tree: the root at the top, then top-level domains (TLDs) like `.com`, then domains like `example.com`, then subdomains like `www.example.com`.

Parent and child zones

- The **parent** is the zone above you. For `example.com`, the parent is `.com`.
- The **child** is the zone you operate. For `www.example.com`, the child zone is `example.com`.
- The parent lists which **nameservers** are authoritative for the child and, if DNSSEC is used, publishes a **DS (Delegation Signer)** record that points at the child's key.

> **Security view:** Think of the parent as saying, "If you want the truth about this child zone, go talk to these servers, and trust any DNSSEC signatures that chain back to this specific key."

Records you'll see mentioned often

- **NS** – which nameservers are authoritative for a zone.
- **SOA** – the "start of authority" record; includes the primary name server and the zone serial number.
- **A / AAAA** – address records (IPv4 and IPv6).
- **MX** – mail exchangers.

2. DNSSEC Basics in Security Terms

DNSSEC (DNS Security Extensions) adds **signatures** to DNS, so a resolver can tell whether data was tampered with or forged. It does not encrypt data; it *authenticates* it.

Keys, signatures, and fingerprints

- **DNSKEY** – public keys published in the zone. There are usually two roles:
  - **KSK (Key Signing Key)**: signs the DNSKEY set itself.
  - **ZSK (Zone Signing Key)**: signs the rest of the data (A, AAAA, MX, etc.).
- **RRSIG** – signature records that prove an RRset (a group of records of the same type) was signed with a valid DNSKEY.
- **DS** – small "fingerprint" of a DNSKEY stored in the **parent zone**.

When everything is correct, a validating resolver can start from a “trust anchor” (usually the root) and walk downwards: root → TLD → domain → record, checking that each DS matches a DNSKEY and each RRSIG is valid.

> **Security story:** DNSSEC answers two questions:
>
> 1. “Did this answer really come from the zone’s legitimate owner?”
> 2. “Was it changed while in transit?”
>
> The PRO tab cards help you see where that story breaks down.

3. Using dig and nslookup to verify DNSSEC

The DNSSEC Checker app runs many queries for you. This section shows how to reproduce some of the evidence manually using command-line tools.

3.1 Linux/macOS: dig

Basic A record lookup:

```
dig example.com
```

Ask for DNSSEC data (notice the `+dnssec` flag):

```
dig +dnssec example.com
```

Look at the zone’s DNSKEY set:

```
dig +dnssec DNSKEY example.com
```

Look at DS at the parent (here, from the `.com` zone):

```
dig +dnssec DS example.com
```

Query a specific authoritative nameserver directly:

```
dig @ns1.example.com +dnssec example.com
```

3.2 Windows: nslookup

Open a command prompt and start interactive nslookup:

```
nslookup
```

Set a specific DNS server (optional):

```
> server 8.8.8.8
```

Look up an A record:

```
> example.com
```

Windows' built-in `nslookup` does not show DNSSEC signatures directly, but you can still:

- Check that you are reaching the same nameservers as shown in the app:

```
> set type=ns
> example.com
```

- Check MX, SOA and other types:

```
> set type=soa
> example.com
```

> **Tip:** For full DNSSEC detail on Windows, install a dig-compatible toolset (for example via BIND tools or WSL) and use the dig examples above. Then you can visually match DNSKEY, DS, and RRSIG records against what the PRO cards show.

4. How the PRO Cards Fit Together

Think of the PRO tab as a guided tour of your domain's DNSSEC posture:

- **Executive Scores** – "How healthy is this zone overall?"
- **Trust Chain + Parent DS + DS/DNSKEY views** – "Is my chain of trust intact?"
- **DNSKEY, Active Keys, Crypto Profile** – "Are my keys and algorithms safe and current?"

- **Authoritative NS + Per-NS + NS Details** – “Are all servers in sync and behaving?”
- **RRSIG Details + Coverage + Negative Validation** – “Is everything that should be signed actually signed and valid?”
- **Resolver Path** – “How do public resolvers see me?”
- **Recommendations & Reports** – “What do I fix first, and how do I show others?”

5. Executive Scores

At-a-glance health across five dimensions.

What this card tells you

This card summarizes complex checks into a small set of scores, plus an overall grade. If you are new to DNSSEC, start here first. Red or orange areas tell you where to look next.

How to sanity-check with tools

- If **Chain Integrity** is low:
  - Run:

    ```
    dig +dnssec DS example.com
    ```

  - Then:

    ```
    dig +dnssec DNSKEY example.com
    ```

  - You should see at least one DS whose key tag and digest type match a DNSKEY key tag and algorithm.
- If **Crypto Strength** is low (e.g. due to SHA-1):
  - Look at the `DNSKEY` section from `dig +dnssec DNSKEY example.com` .
  - Note the algorithm numbers (e.g., 8 = RSASHA256, 13 = ECDSAP256SHA256). Older algorithms (like pure SHA-1) reduce the crypto score.

6. Assessment Vantage

Explains where the app is “standing” when it looks at your DNSSEC.

Concept

DNSSEC Checker can probe:

- **Authoritative servers directly** (best for debugging operations).
- **Public resolvers** like Google or Cloudflare (best for understanding how users see you).

How to reproduce

- Query an authoritative server directly:

```
dig @ns1.example.com +dnssec example.com
```

- Query via a recursive resolver (e.g. Google DNS):

```
dig @8.8.8.8 +dnssec example.com
```

If answers differ between these two, it often indicates caching, filtering, or network issues that affect what users actually see.

7. Trust Chain Integrity

"Do the parent DS records actually match the child keys?"

What this card shows

This card checks whether the parent's **DS records** correctly match the child's **DNSKEY** set. You'll see a summary like:

MATCH at least one DS matches a DNSKEY (good)
NO_MATCH DS records exist, but none matches (broken)
MISSING no DS at parent (DNSSEC not anchored)

Verifying with dig

1. Check DS at the parent:

   ```
   dig +dnssec DS example.com
   ```

2. Check DNSKEY at the child:

```
dig +dnssec DNSKEY example.com
```

Note the **key tag** shown on DS records and match it to a **key tag** on a DNSKEY. If there is no matching DNSKEY, the trust chain is broken.

> **If the card says NO_MATCH:**
> It usually means a key rollover was done on the child zone but the parent's DS was not updated, or an old DS was left behind. Fix this first; it can cause all DNSSEC validation to fail for the domain.

8. Parent DS Link

Focuses on what the parent zone (e.g. TLD) says about you.

What this card shows

- Which parent nameservers are used.
- IPv4 "glue" records that let resolvers reach your nameservers.
- The DS records and how their signatures (RRSIG(DS)) line up with time.

Check with dig

To see DS and its signatures:

```
dig +dnssec DS example.com
```

The `ANSWER` section shows DS; the `RRSIG` lines under DS show who signed them and the expiration window. The app's view explains whether those windows are healthy.

9. DS / DNSKEY Records (merged)

Side-by-side view of DS fingerprints and the DNSKEYs they refer to.

What this card shows

- Each DS: key tag, algorithm, digest type and digest.
- Each DNSKEY: key tag, algorithm, flags (KSK or ZSK), Base64 key.
- Which pairs **match** (fingerprint matches key) and which do not.

How to check

Use the same two commands as for Trust Chain Integrity:

```
dig +dnssec DS example.com
dig +dnssec DNSKEY example.com
```

Compare key tags by eye; in most environments you will rely on the app’s matching logic, but this is how to confirm the evidence if needed.

10. DS Records (full list)

The complete DS set visible at the parent.

Why this matters

When configuring DNSSEC in a registrar portal, you often copy DS lines from your DNS provider’s interface. This card gives you the authoritative view of what the parent currently has, so you can spot stale or incorrect entries.

Verify with dig

```
dig +dnssec DS example.com
```

If the app shows more DS records than you expect (for example, an old KSK’s DS left behind), use this command to confirm and then remove the stale DS at the registrar.

11. DNSKEY Records (full list)

Everything the child zone publishes as keys.

What you’ll see

- Key tag – numeric ID.
- Flags – typically 257 for KSK, 256 for ZSK.
- Algorithm – which crypto algorithm is used (e.g., 8 = RSASHA256, 13 = ECDSA P-256).
- Public key – Base64 string.

Check with dig

```
dig +dnssec DNSKEY example.com
```

> **For novices:** You don't need to understand the Base64 key itself. Focus on:
>
> - How many KSK (flags=257) and ZSK (flags=256) you see.
> - Which algorithms they use.
> - Which ones your DS records point to.

12. Active Keys & Expiry (Keys Validity)

Are the signatures on your DNSKEY set current and healthy?

What this card shows

DNSSEC signs the `DNSKEY` RRset itself. This card focuses on the **RRSIG(DNSKEY)** records: when they started being valid, when they expire, and how many days are left. Keys that are about to run out of signature coverage are highlighted.

Inspecting with dig

```
dig +dnssec DNSKEY example.com
```

In the output, look for `RRSIG DNSKEY` lines. They include an **inception time**, an **expiry time**, and the key tag that signed the DNSKEY set.

> **If days remaining is low:**
> Make sure your DNS provider's signing process is running, and that all authoritative servers have loaded the freshly signed zone. Expired RRSIG(DNSKEY) can cause all DNSSEC validation to fail.

13. Cryptographic Profile

Which algorithms and key types are used, and how strong they are.

What this card summarizes

- Which algorithms are in use (RSA, ECDSA, Ed25519, etc.).
- Key sizes for RSA keys (e.g., 1024 vs 2048+).
- Whether any **SHA-1-based** algorithms or digests are present (which cap the score).

How to see algorithms directly

Use the DNSKEY view:

```
dig +dnssec DNSKEY example.com
```

The `alg` field shows an integer that corresponds to a specific algorithm. Your app converts this to human-readable labels (e.g., "ECDSA P-256").

> **Important:** SHA-1 and short RSA keys are considered weak by modern standards. If the card flags SHA-1 usage, plan to migrate to newer algorithms like RSASHA256 or ECDSA P-256.

14. Authoritative Nameservers

Which servers are supposed to speak for your zone.

What this card shows

- NS names discovered from the parent zone.
- IPv4 glue (A records) used to reach them.
- Notes about IPv6-only servers or missing glue.

Check from the command line

Linux/macOS:

```
dig NS example.com
dig +trace example.com   # walks the chain from root down
```

Windows nslookup:

```
nslookup
> set type=ns
> example.com
```

> **Why this matters:** If a nameserver is listed here but unreachable, or has wrong glue, resolvers may never get your signed data, and validation can fail even if your keys are perfect.

15. Per-Nameserver Evidence (grid of mini-cards)

Compares each nameserver's behavior at a glance.

What this card highlights

- Whether each NS supports EDNS, DNSSEC (DO bit), and TCP fallback.
- Zone serial (SOA) on that server and whether it matches its peers.
- Quick view of RRSIG health on that server.

Check a single server with dig

```
dig @ns1.example.com SOA example.com
dig @ns1.example.com +dnssec example.com
```

Compare SOA serial numbers across servers; large differences indicate one server is behind. If a server doesn't return DNSSEC records even with `+dnssec`, it may be misconfigured or filtered.

16. Nameserver Details

Deep-dive evidence for each individual NS.

What you can see

- Exact EDNS buffer size, whether DO is honored, and whether TC (truncation) is set.
- SOA serial and whether it matches peers.
- Full RRSIG matrix: which RRsets are signed, by which key, and with what validity window.

Useful dig commands

```
dig @ns1.example.com +dnssec +bufsize=1232 example.com
dig @ns1.example.com +dnssec SOA example.com
```

> **Look for:** one server that:
>
> - Has a much older serial.
> - Returns smaller responses or omits DNSSEC records.
> - Shows expired signatures while others are current.
>
> That server is a likely cause of intermittent DNSSEC failures.

17. RRSIG Details

Status of signatures for each important RRset.

What this card summarizes

- For each RR type (SOA, NS, DNSKEY, A/AAAA, MX, etc.):
- Which key signed it (key tag and algorithm).
- Validity window (inception → expiry, days left).
- Whether the signature is OK, FAIL, or OK but only with SHA-1.

See them with dig

```
dig +dnssec example.com         # look for RRSIG A / AAAA / NS
dig +dnssec SOA example.com      # look for RRSIG SOA
dig +dnssec DNSKEY example.com   # RRSIG DNSKEY
```

In each case, the `RRSIG` lines directly correspond to what this card is summarizing.

18. Coverage of Signed RRsets

How much of what matters is actually signed and validating.

Concept

It's possible to have keys and some signatures, but still miss critical RRsets. This card looks at whether important RRsets (SOA, NS, DNSKEY, apex A/AAAA, MX) are all signed and validate correctly.

Manual checks

Use dig with `+dnssec` and check that each answer has RRSIGs:

```
dig +dnssec SOA example.com
dig +dnssec NS example.com
dig +dnssec example.com          # A / AAAA
dig +dnssec MX example.com       # if you have mail
```

> **If coverage is less than 100%:**
> Look at which RR types are marked missing or invalid in the card. Those types should be in your DNSSEC signing policy but currently aren't properly covered.

19. Negative Answer Validation

Secure proof that a name or record type does not exist.

Why this matters

DNSSEC also needs to prove that certain names or record types do **not** exist. It does this with `NSEC` or `NSEC3` records, which are themselves signed.

How to test an NXDOMAIN

Ask for a clearly non-existent name:

```
dig +dnssec nosuchname-definitely-not-real.example.com
```

In a DNSSEC-signed zone, you should see `NSEC` or `NSEC3` records in the `AUTHORITY` section with their own `RRSIG`. The app's card reports whether those negative proofs validated.

20. Resolver Path / Capabilities

How common resolvers (like ISPs or public DNS) see your domain.

What this card shows

- Per resolver: whether it sends DO (DNSSEC OK) and whether responses have AD (Authenticated Data) set.
- Whether any resolver seems to suppress DNSSEC data.

Checking AD with dig

```
dig @8.8.8.8 +dnssec example.com
```

In the output header, look for `ad` in the flags list. If present, the resolver claims the answer was DNSSEC-validated. If the app shows AD=0 for a resolver, it means that resolver is not validating or validation failed.

21. Recommendations

Prioritized action plan.

How to use this card

For each Critical / High / Medium / Low recommendation, the app links it back to the evidence shown on the other cards. As a novice, focus on:

1. Fix anything marked Critical (broken DS link, expired signatures, unreachable authoritative servers).
2. Then address High items (SHA-1 usage, short signing windows, serious coverage gaps).
3. Use Medium/Low for ongoing hygiene and future improvements.

> **Practical workflow:** Treat this card as your “runbook.” Open each linked card, verify a few sample records with `dig` or `nslookup`, then make the required changes in DNS or at the registrar.

22. Reports

Sharing the story with others.

What this card provides

- Buttons to generate HTML / JSON (and PDF if enabled) reports.
- Options to open, share, or email the report.

Use these reports to:

- Provide evidence to auditors that DNSSEC is correctly deployed.
- Share specific issues (for example, broken DS or misconfigured nameserver) with your DNS or registrar support team.
- Keep a historical record of your DNSSEC posture over time.