DNSSEC, the DANE PKI

(and OpenSSL)

Viktor Dukhovni <viktor@openssl.org>, 08/10/2025

Assumed WebPKI background

- Basic asymmetric cryptography: private and public keys
- Certificates: signed bindings of public key to holder name(s) (SANs)
- Certification authorities (CAs)
  - Trust anchors (root CAs), subsidiary (intermediate) CAs, cross CAs
  - OCSP services, CRL distribution points
- CA/B forum, trust "bundles"
- Domain (control) Validation, ACME (Let's Encrypt, ...)

WebPKI limitations

- Lack of effective name constraints
  - All CAs in the trust bundle are equally trusted to certify any domain
  - CA "C=" In Fedora 41 trust bundle, are all these countries "friends"?

    59 **US**, 11 **DE**, 10 **CN**, 7 **ES**, 6 **GB**, 5 **PL**, 5 **BM**, 4 **TW**, 4 **JP**, 4 **GR**, 4 **CH**,
    3 **HU**, 3 **BE**, 2 **RO**, 2 **NO**, 2 **IN**, 2 **FR**, 1 **TR**, 1 **TN**, 1 **SK**, 1 **KR**, 1 **IT**, 1 **IE**,
    1 **HK**, 1 **FI**, 1 **AT**

- Insecure "domain validation"
- Authenticates logical hosts, rather than specific services

Domains and DNS

(DNS) Domain Hierarchy

Root zone

- Hosts **referrals** to >1400 top-level (TLD) delegated subdomains:
  - 1040 Generic (**gTLD**): .arpa, .com, .edu, .gov, .net, .mil, .org, ...
  - 248 ISO country code (**ccTLD**): .au, .br, .cz, .dk, .es, .fr, ...
  - Internationalised (**IDNA**): 61 country (.中國), 90 generic (.コム)
- Served by many "anycast" root server nodes
  - Managed by 13 Root Server operators: a–m.root-servers.net
- Small enough to replicate and serve locally: https://localroot.isi.edu/

Top-level domains (TLDs)

Registry, Registrar, Registrant (RRR)

- Operate **registries** of:
  - End-user (**registrant**) **2LD** domains (example.com)
  - Public-suffixes (.co.uk, .noda.chiba.jp, ...) with **3LD**, **4LD**, ... registrants
- DNS server operation may be (partly?) outsourced (Afilias, PCH, ...)
- **Registrants** typically obtain and manage their domains via a **Registrar** (Godaddy, Cloudflare, ...)
- End-user domain DNS service is often outsourced
- DNS operator starting to be formalised as a participant in the management model

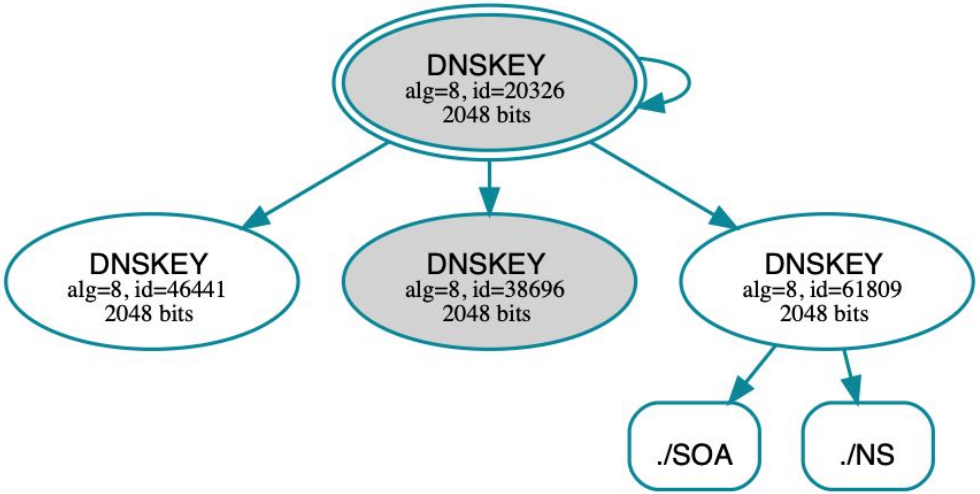
DNSSEC (specialised PKI)

Hardens DNS against spoofing and cache poisoning

- End-to-end authentication of DNS data (possibly multiple intermediate caches)
- Hierarchical and federated, each zone signs its own data
- Parent zone **DS** record set (RRset) validates child zone's **DNSKEY** RRset, which in turn validates the rest of the zone data
- Critically, protects the content or absence of **DS** records of any further delegated domains
- Authenticated denial of existence (DoE) validates **NODATA** and **NXDOMAIN** answers

DNSSEC root zone key-signing-key (KSK)

- Rotated every ~8 years (RFC 5011)
  https://dnsviz.net/d/root/aLeZHw/dnssec/
  https://stats.dnssec-tools.org/explore/?.



DNSKEY Age	Key Tag	Flags	Algorithm
3m 13d	46441	256	RSASHA256 (8)
13d 12h	61809	256	RSASHA256 (8)
7y 11m	20326	257	RSASHA256 (8)
8m 20d	38696	257	RSASHA256 (8)

.

(2025-10-01 13:52:55 UTC)

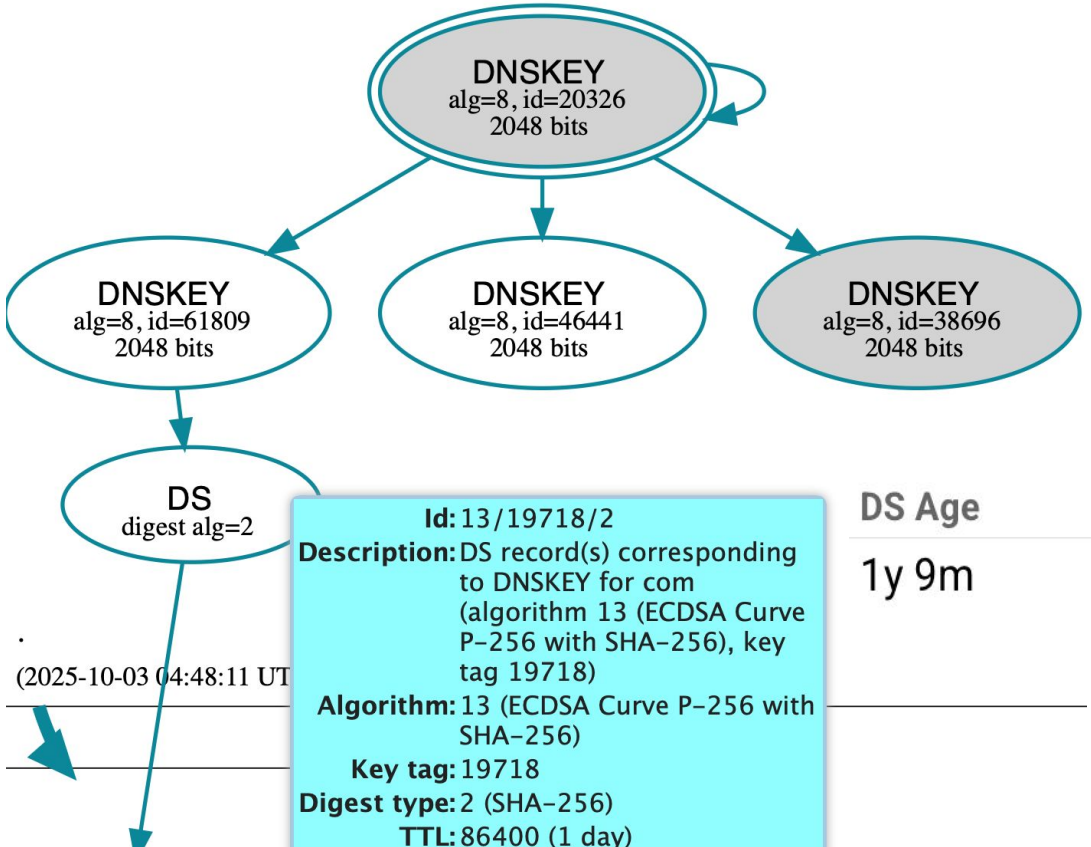
DNSSEC

Root zone operations

- ICANN: Root zone administrator and KSK custodian
- Verisign: Root Zone Maintainer and Root Zone Signing Key Operator
  - Operates the "a" root servers
- DNSSEC-signed since July 2010, last KSK rollover 2018, next 2026
  - Detailed history:

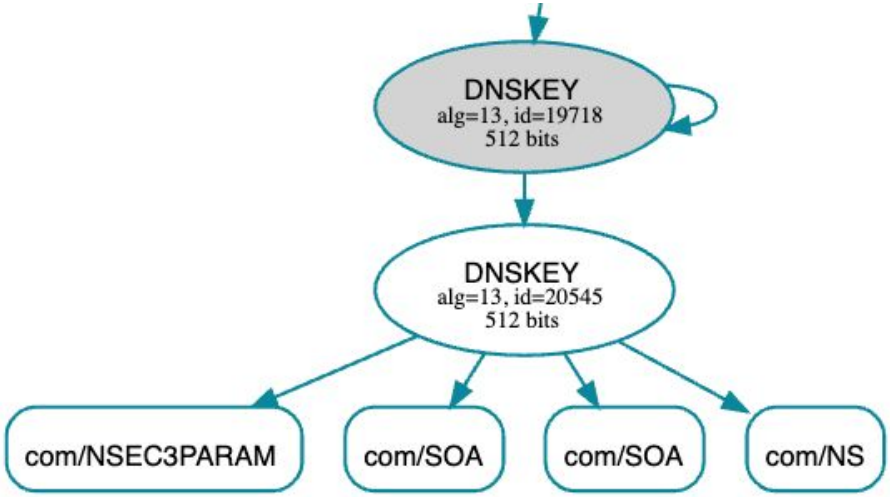
    https://www.ausnog.net/sites/default/files/ausnog-04/presentations/ausnog-04-d01p07-joe-abley-icann.pdf

Root zone TLD (.com) delegation



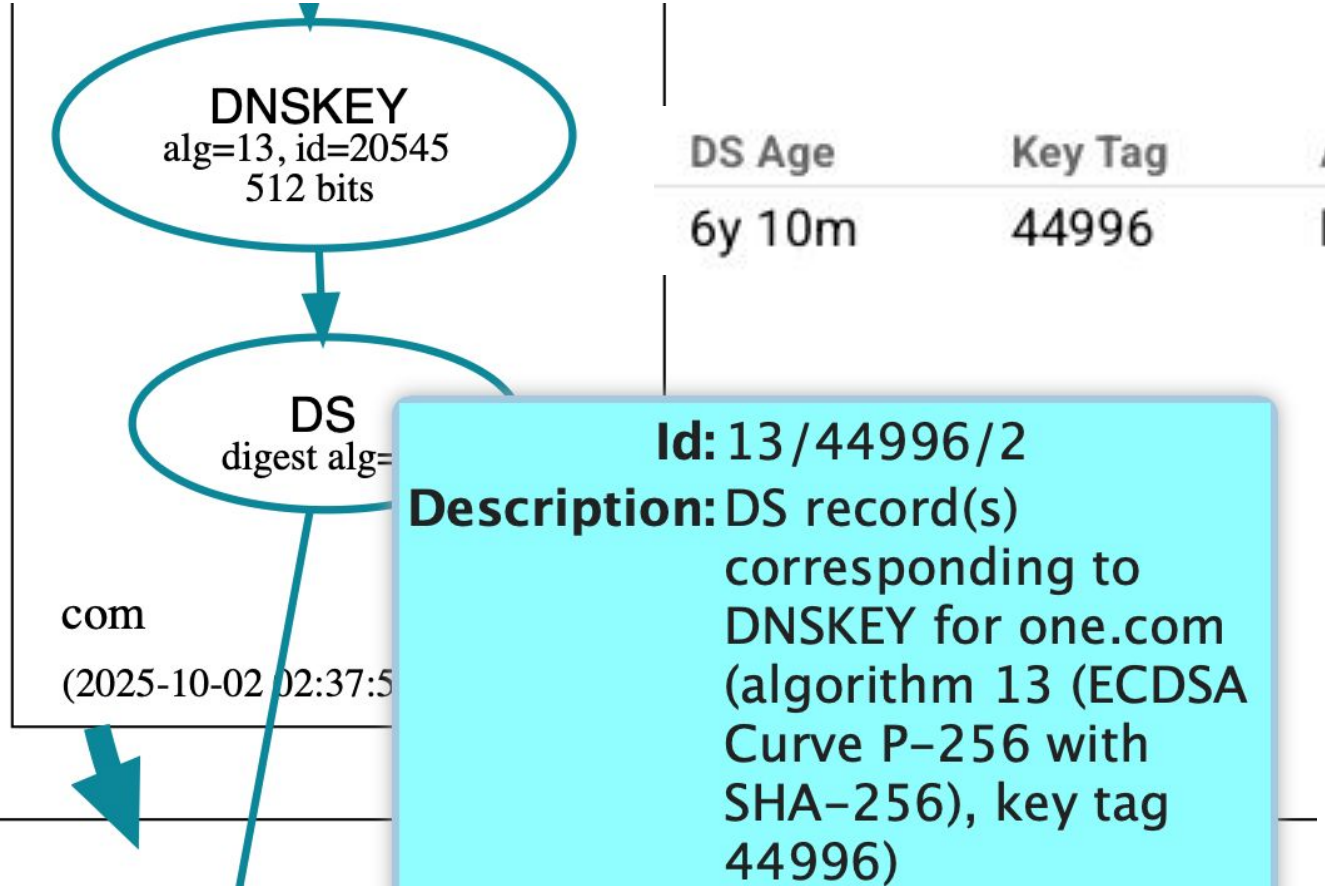
DS Age	Key Tag	Algorithm
1y 9m	19718	ECDSAP256SHA256 (13)

.COM zone apex



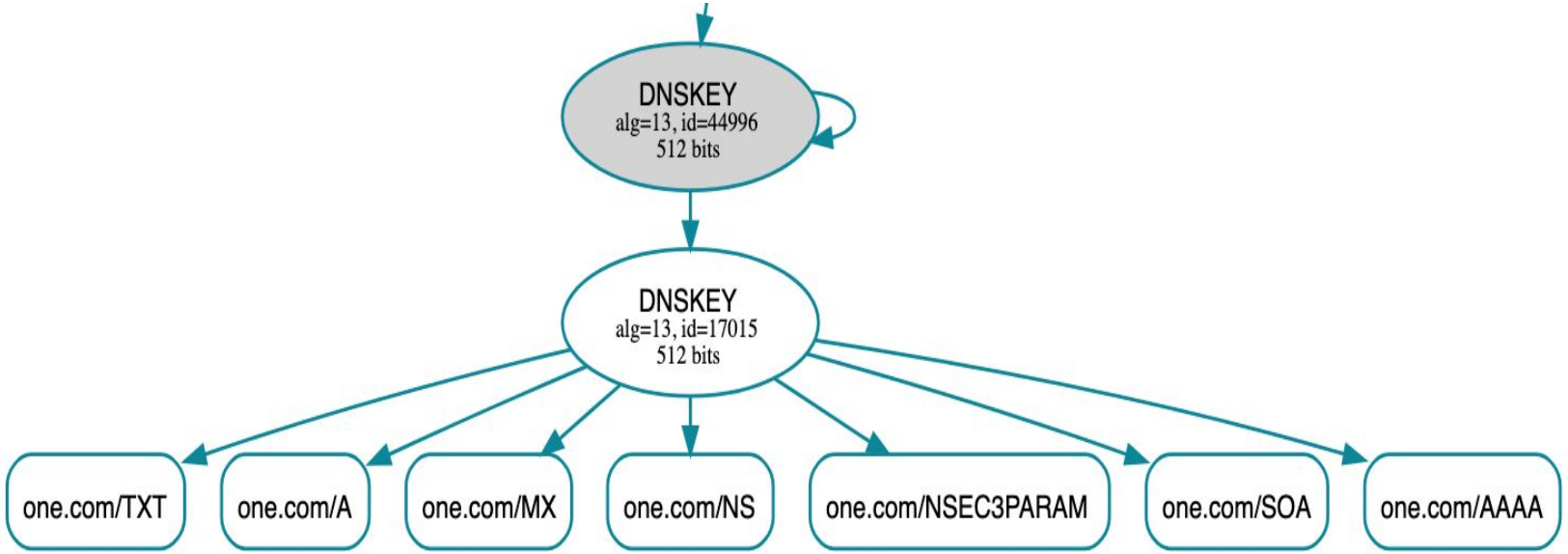
DNSKEY Age	Key Tag	Flags	Algorithm
2m 23d	20545	256	ECDSAP256SHA256 (13)
1y 9m	19718	257	ECDSAP256SHA256 (13)

2LD Delegation (.com → one.com)



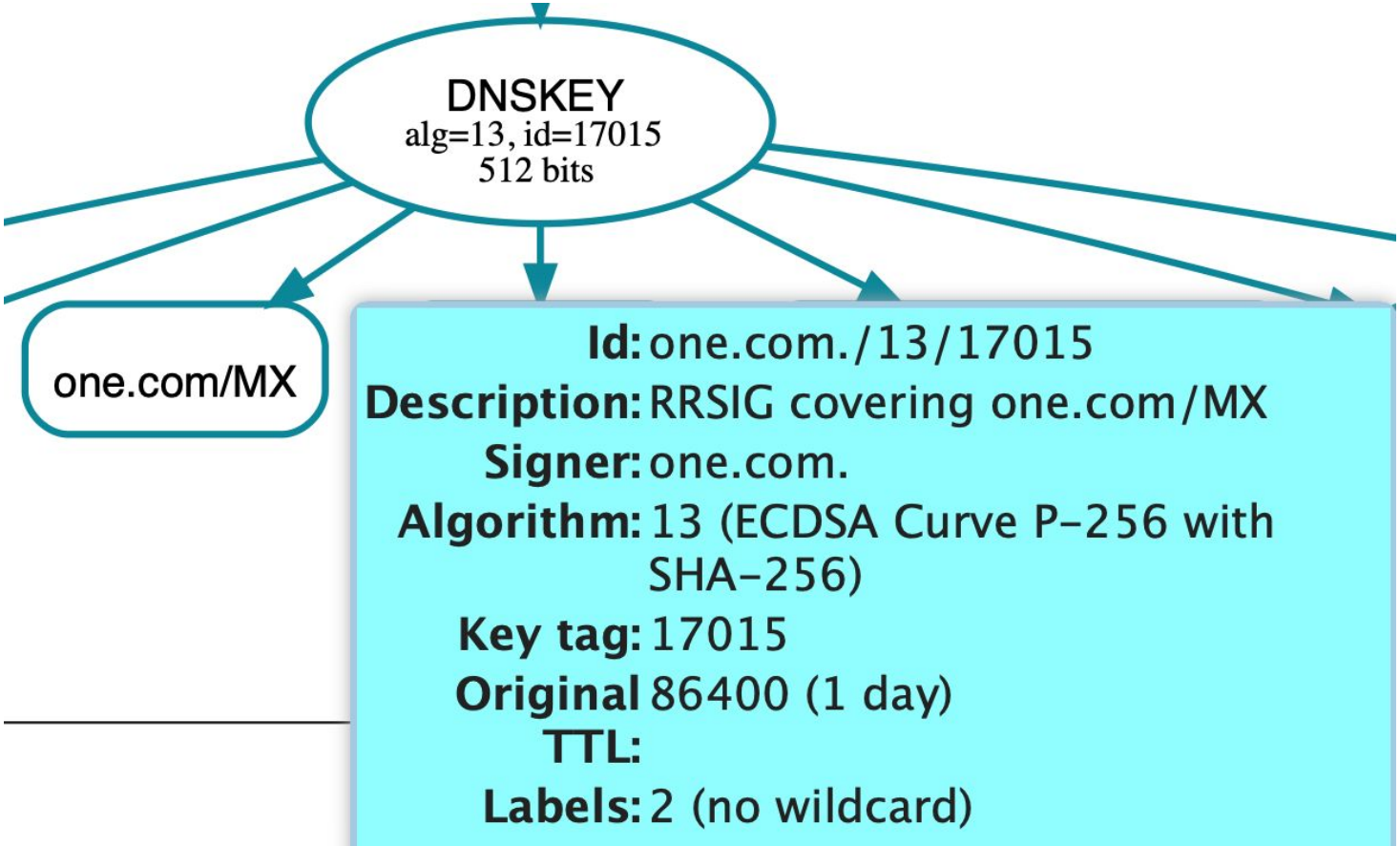
DS Age	Key Tag	Algorithm
6y 10m	44996	ECDSAP256SHA256 (13)

one.com zone apex



DNSKEY Age	Key Tag	Flags	Algorithm
6y 10m	17015	256	ECDSAP256SHA256 (13)
6y 10m	44996	257	ECDSAP256SHA256 (13)

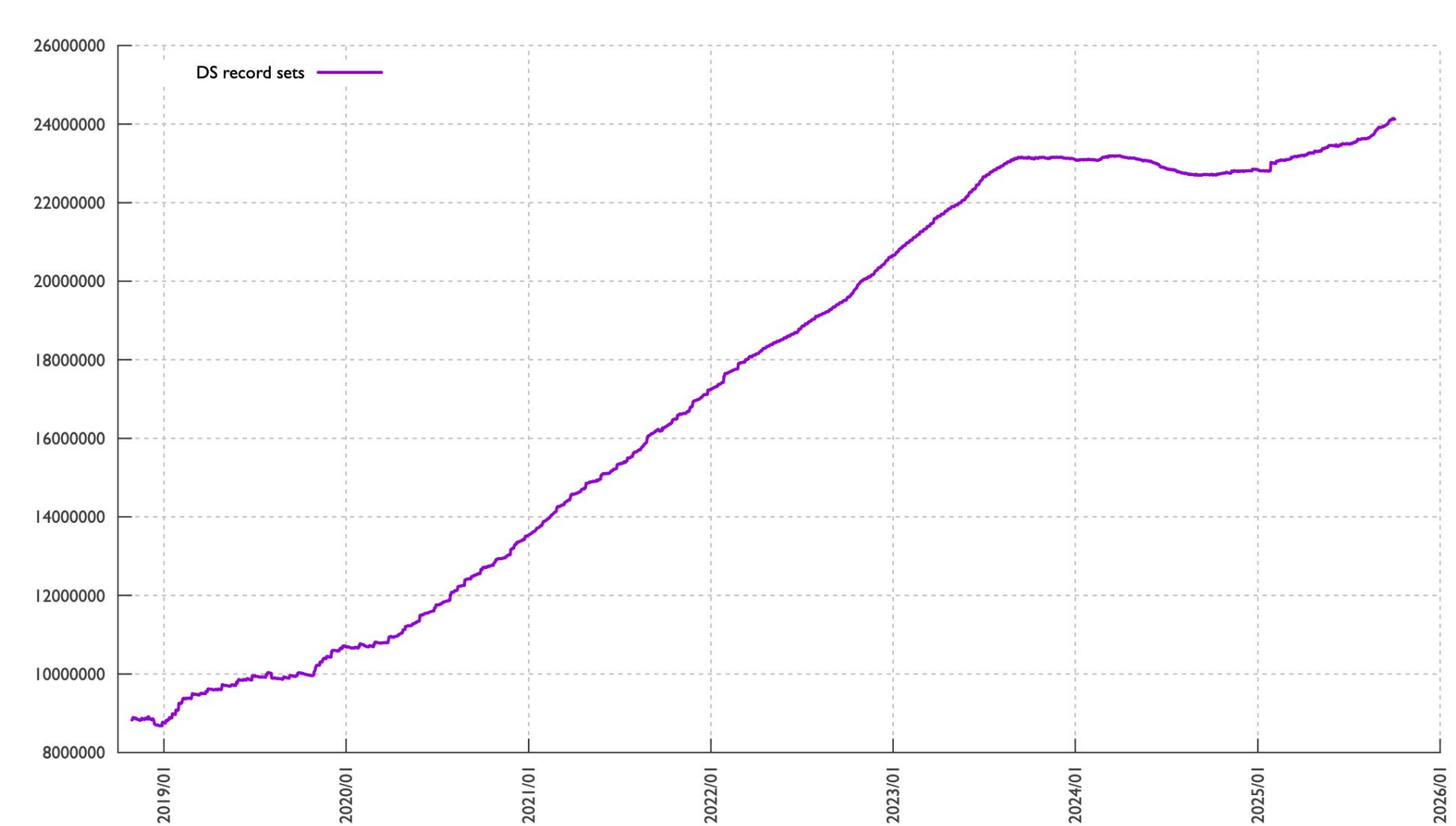
one.com MX RRset signature



DNSSEC

Summary

- DNSSEC is a PKI
  - With built-in name constraints, .ru can't sign .mil domains
  - With generally short **signature** lifetimes
  - No need for 3rd-party "domain validation"
- Recognising rôle of DNS operators in DNSKEY management is one of the goals of IETF DELEG WG
- Adoption is growing, for now low outside of EU and Brazil


DS record sets
26000000
24000000
22000000
20000000
18000000
16000000
14000000
12000000
10000000
8000000
2019/01
2020/01
2021/01
2022/01
2023/01
2024/01
2025/01
2026/01

DANE TLSA

DNSSEC PKI for TLS-enabled applications

(RFC 6698, RFC 7671)

DNS TLSA records

- DANE TLSA DNS Resource Record (RR) set:

```
_25._tcp.mail.example.org. IN TLSA 3 1 1 0123...cdef
_<port>._<proto>.<name>. IN TLSA <usage> <selector> <mtype> <data>
```

- The **usage** determines whether matching end-entity (1/3) or trust anchor (0/2), and whether it constrains (0/1) or overrides (2/3) local trust store
- Each RR associates the (port, protocol, DNS name) triple with one or more
  - public keys or enclosing certificates (per the **selector**),
  - their full DER encoding or a digest (per the **mtype**)
- Data is hex-encoded in zone files, and raw binary "on the wire"

TLSA records…

- Any **one** matching TLSA record is sufficient, making rollover non-disruptive
- Mnemonics in RFC 7218:

```
... IN TLSA DANE-EE(3) SPKI(1) SHA2-256(1) ...
```

- Example hotmail.cz:
  - Authenticated either via given issuer CA or given server public key
  - Both TLSA records match (two DANE-EE records present during key rollovers)

```
hotmail.cz. IN MX 0 hotmail-cz.f-v1.mx.microsoft.
_25._tcp.hotmail-cz.f-v1.mx.microsoft. IN CNAME smtpdane.mx.microsoft.
smtpdane.mx.microsoft. IN TLSA 2 0 1 5f88…dd44
smtpdane.mx.microsoft. IN TLSA 3 1 1 c495…b0ea
```

DANE for SMTP (and XMPP)

(poorly served by WebPKI)

- SMTP (and XMPP) servers are found indirectly through MX (and SRV) records
- WebPKI assumes application has access to a trusted server name
  - But with DNS (mostly) unsigned SMTP and XMPP server names are untrusted
  - SMTP TLS is by default optional and unauthenticated, need downgrade-resistant signal to mitigate active (MiTM) attacks
  - RFC 7672 section 1.3 motivates use of DANE for SMTP
    - RFC 7673 follows suit for SRV-record based indirection
- DANE deployment cost scales with server count, not hosted domain count!

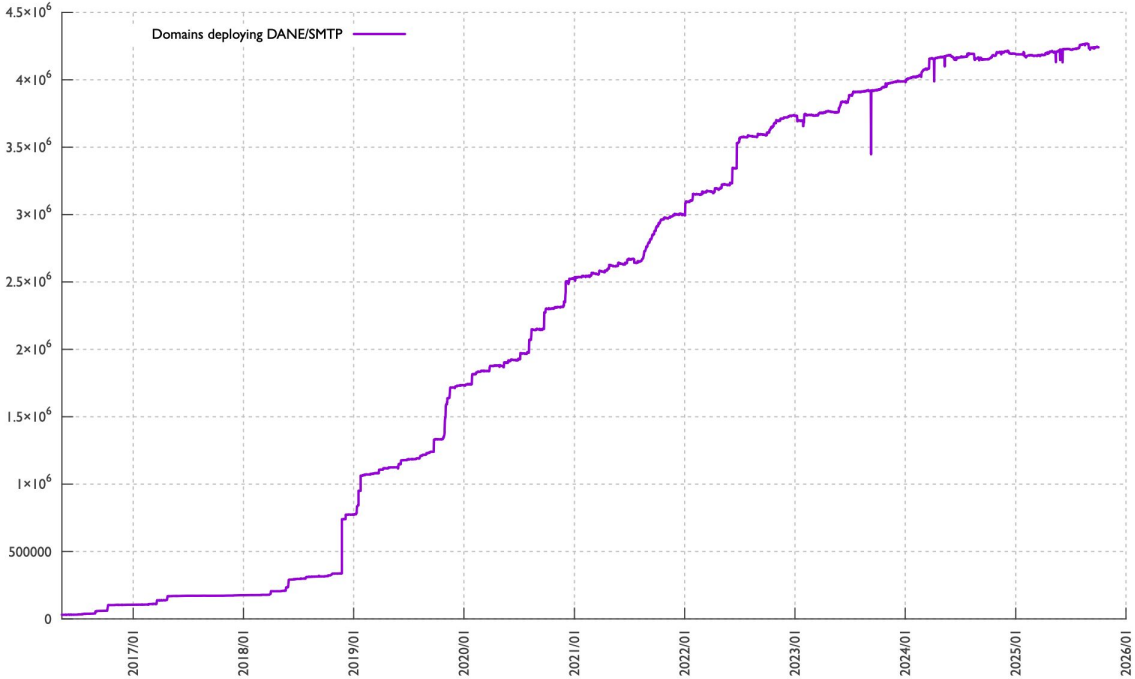
DANE operations

- Better scaling than MTA-STS (no per-domain effort)
- No need for CRLs or OCSP: just change the TLSA records in DNS
- Since any one matching TLSA record is sufficient
  - Add TLSA records for upcoming cert/key before deployment
  - Remove TLSA record for no longer live cert/key after deployment
- Best practice TLSA RRsets are dual "current + next" DANE-EE(3) SPKI(1) SHA2-256(1).
  - The next key TLSA published at least a few TTLs in advance of cert deployment
  - DANE-EE(3) SPKI(1) records are compatible with RFC 7250 raw public keys
- Robust automation is a must

DANE SMTP adoption

https://stats.dnssec-tools.org

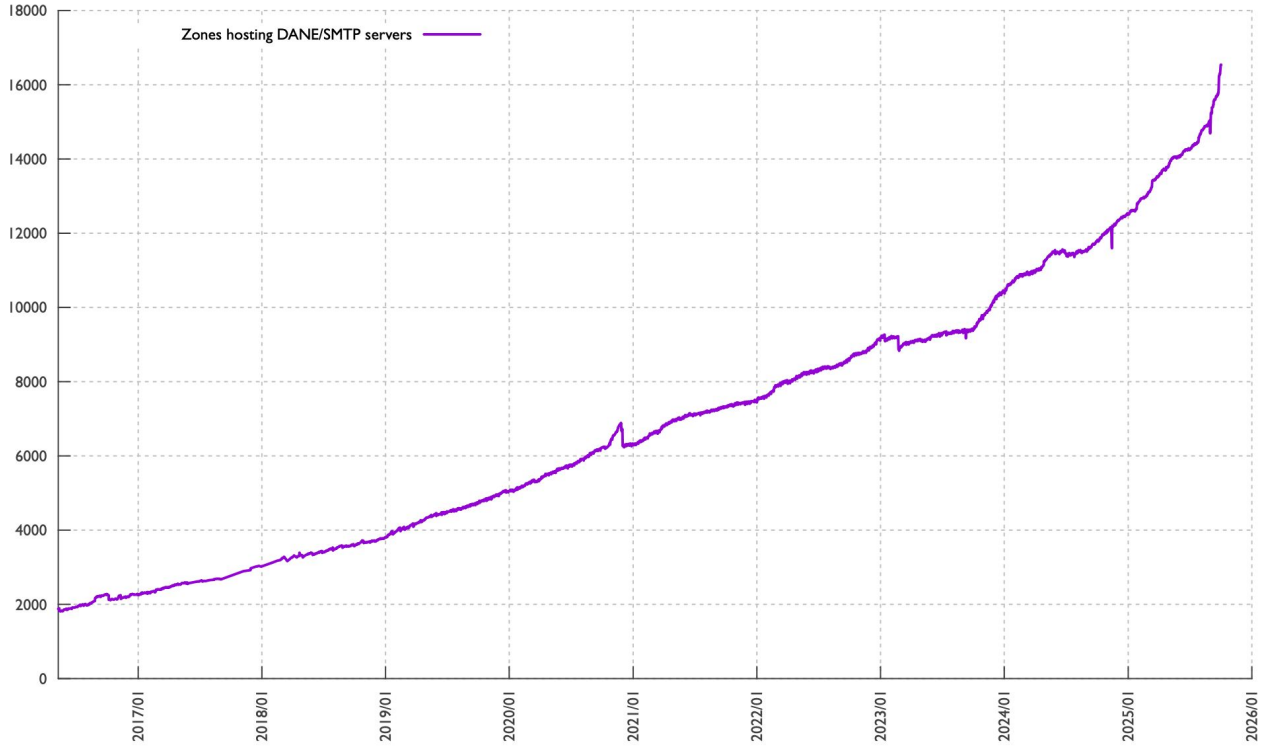
- Used by MX hosts of ~4.25 million (~18% of all DNSSEC-signed) zones with a "public suffix" parent



DANE SMTP adoption

https://stats.dnssec-tools.org

- These MX hosts are in ~16 thousand zones (~28k TLSA RRsets)



DANE in OpenSSL (1.1.0+)

DANE pre-connection initialisation

- Same `ctx` can be used for both DANE and non-DANE TLS

```
SSL_CTX *ctx;
SSL *ssl;
const char *dane_tlsa_domain = "smtp.example.com";

if ((ctx = SSL_CTX_new(TLS_client_method())) == NULL
    || SSL_CTX_dane_enable(ctx) <= 0
    || (ssl = SSL_new(ctx)) == NULL
    || SSL_dane_enable(ssl, dane_tlsa_domain) <= 0)
    /* error */

SSL_set_hostflags(ssl, X509_CHECK_FLAG_NO_PARTIAL_WILDCARDS);

/* Application specific, when no "unknown keyshare" attacks */
SSL_dane_set_flags(ssl, DANE_FLAG_NO_DANE_EE_NAMECHECKS);
```

Adding peer TLSA records

Obtained from DNS or local policy

- Peer's TLSA records, ad hoc trust anchors, required intermediate issuers, RFC 7250 trusted raw public keys, …

```
uint8_t usage, selector, mtype;

for (... each applicable TLSA record ...) {
    unsigned char *data;
    size_t dlen;
    ...
    ret = SSL_dane_tlsa_add(ssl, usage, selector, mtype, data, dlen);
    if (ret < 0)
        /* handle SSL library internal error */
    else if (ret == 0)
        /* handle unusable TLSA record */
    else
        ++num_usable;
}
```

DANE connection setup

```
int (*verify_cb)(int ok, X509_STORE_CTX *sctx) = NULL;
…
if (num_usable == 0) {
    /* Handle all records unusable */
} else {
    SSL_set_verify(ssl, SSL_VERIFY_PEER, verify_cb);
}
/* Complete SSL_connect() handshake and handle errors here */
if (SSL_session_reused(ssl)) {
    /* Verification status available,
     * but not DANE match details */
} else if (SSL_get_verify_result(ssl) == X509_V_OK) {
    … continued …
```

DANE post-handshake reflection

```
    const char *peername = SSL_get0_peername(ssl);
    EVP_PKEY *mspki = NULL;
    int depth = SSL_get0_dane_authority(ssl, NULL, &mspki);

    if (depth >= 0) {
        (void) SSL_get0_dane_tlsa(ssl, &usage, &selector,
                                  &mtype, NULL, NULL);
        printf("DANE TLSA %d %d %d ", usage, selector, mtype);
        if (SSL_get0_peer_rpk(ssl) == NULL)
            printf("%s certificate at depth %d\n",
                   (mspki != NULL) ? "signed the peer" :
                    mdpth ? "matched the TA" :
                            "matched the EE", mdpth);
        else
            printf(bio, "matched the peer raw public key\n");
} else {
    /* Not authenticated, presumably all TLSA rrs unusable */
}
```

Summary

- DANE well suited for cross-org server-to-server TLS
  - Especially with services using SRV or MX records
- Supports raw public keys with DANE-EE(3) SPKI(1) TLSA records
- Supports locally synthesised records for various forms of "pinning"
- Can replace or harden (constrain) the WebPKI
  (by requiring a match with a listed intermediate CA, or specific EE cert)