DNSSEC

Protecting the Integrity of the Domain Name System

Klaus Möller
WP8-T1

Webinar, 7th of December 2020

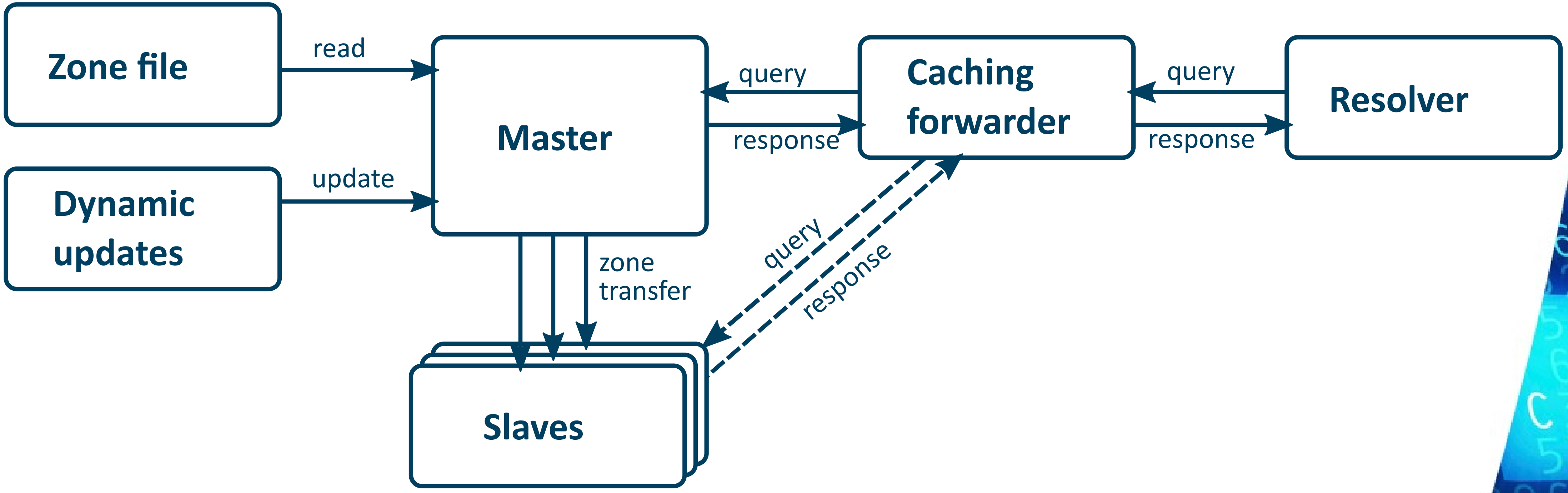
Public

www.geant.org

What we will cover today

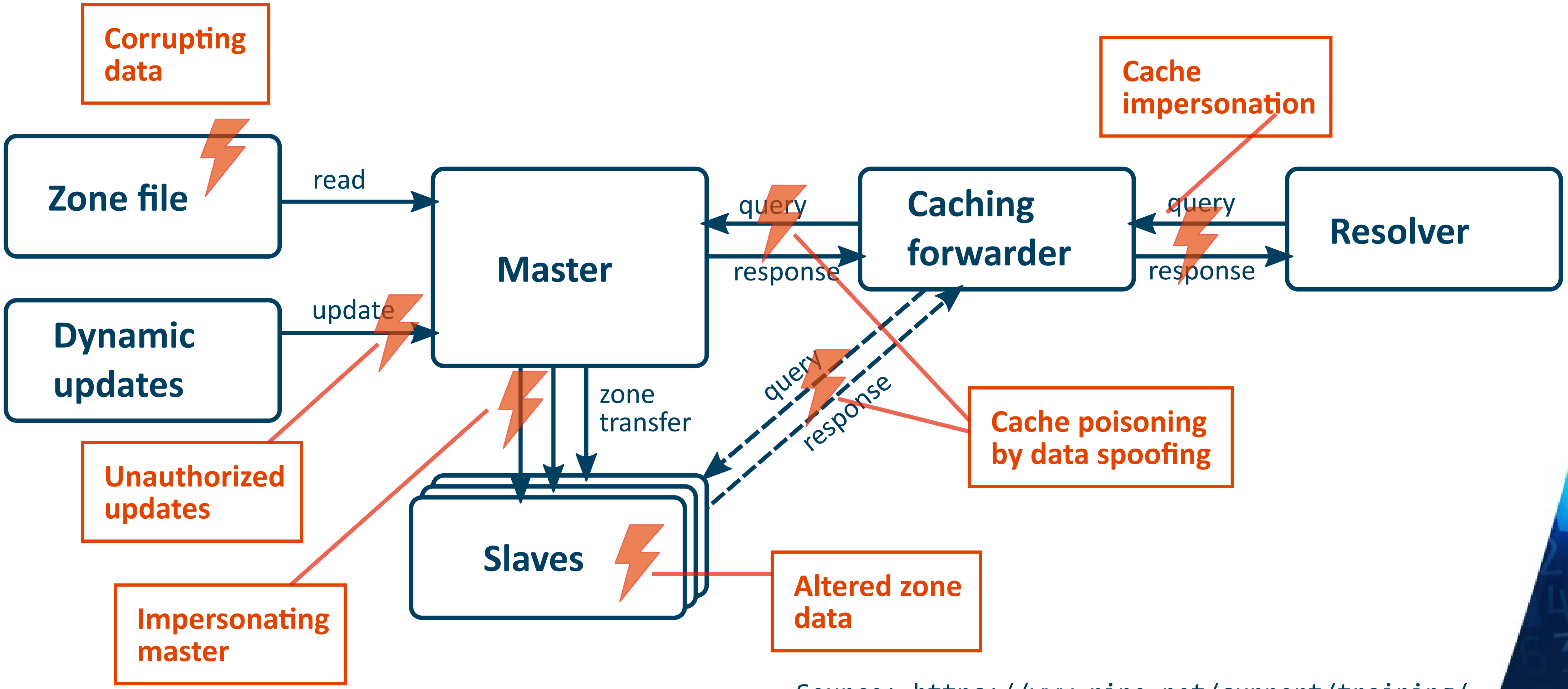
- DNSSEC
  - Motivation
  - Part 1: Transaction Signatures (TSIG)
  - Part 2: Basic DNSSEC Resource Records: RRSIG, DNSKEY, DS
  - Part 3: More DNSSEC RRs: NSEC & NSEC3
  - Part 4: Validating Resolvers
- Examples will use BIND 9(.16) as nameserver
  - And some other client SW (ldns, DNSSEC tools)

Motivation: DNS Data Flow



Source: https://www.ripe.net/support/training/material/ripe-ncc-training-material#DNSSEC

Motivation: DNS Attacks



Source: https://www.ripe.net/support/training/material/ripe-ncc-training-material#DNSSEC

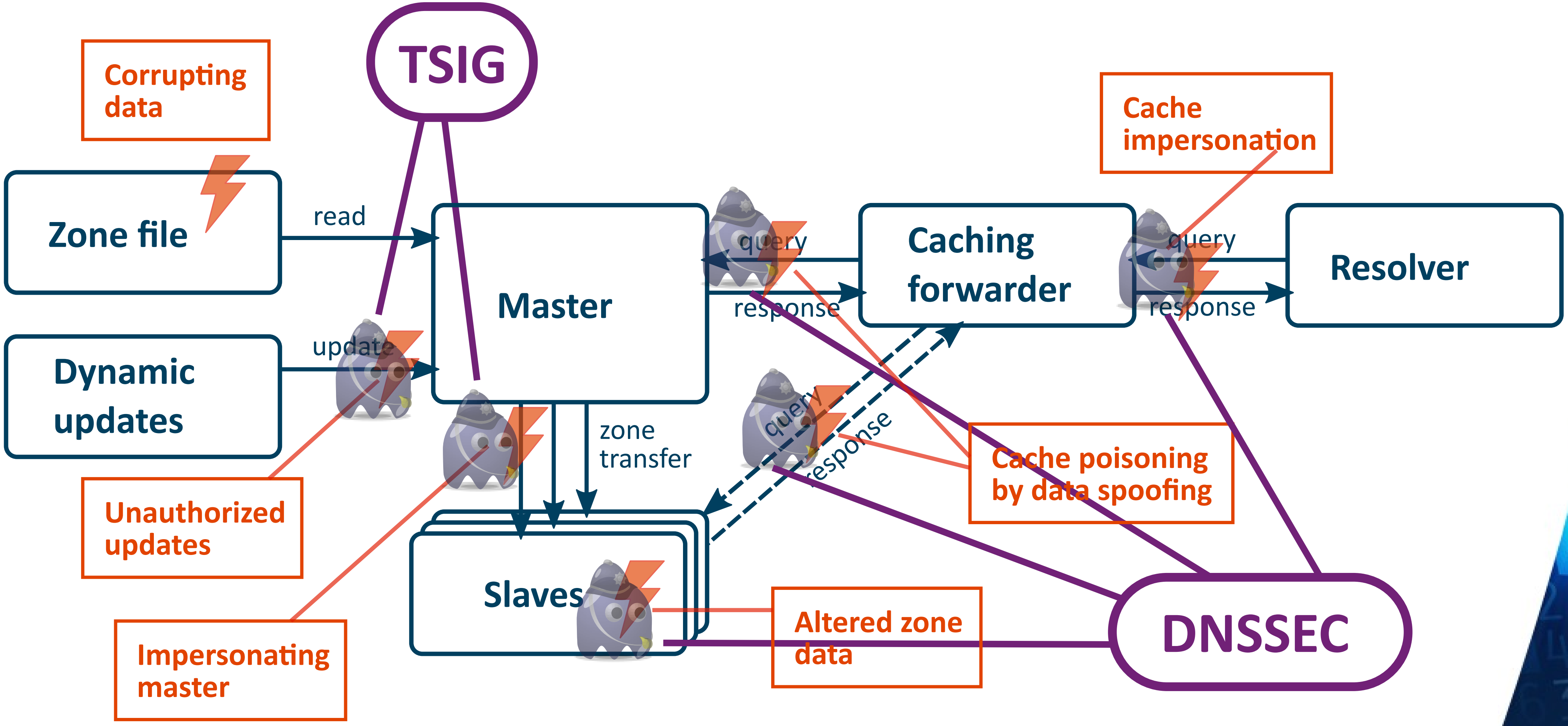
Motivation for DNSSEC

- DNS has no build-in security
  - I.e. no protection of confidentiality, integrity or authenticity (CIA)
  - Practically every other service depends on the integrity of name to address mappings
  - DNS is increasingly used for key verification: SSHFP, DANE, TLSA, CERT, …
- DNSSEC is there for Integrity and Authenticity of
  - Zone Transfers
  - Dynamic Updates
  - Data in Lookups

Zone Transfers, Dynamic Updates: **TSIG/SIG0** (Transaction Signature)

Data in Lookups: **DNSKEY/DS/RRSIG/NSEC/…** (Integrity)

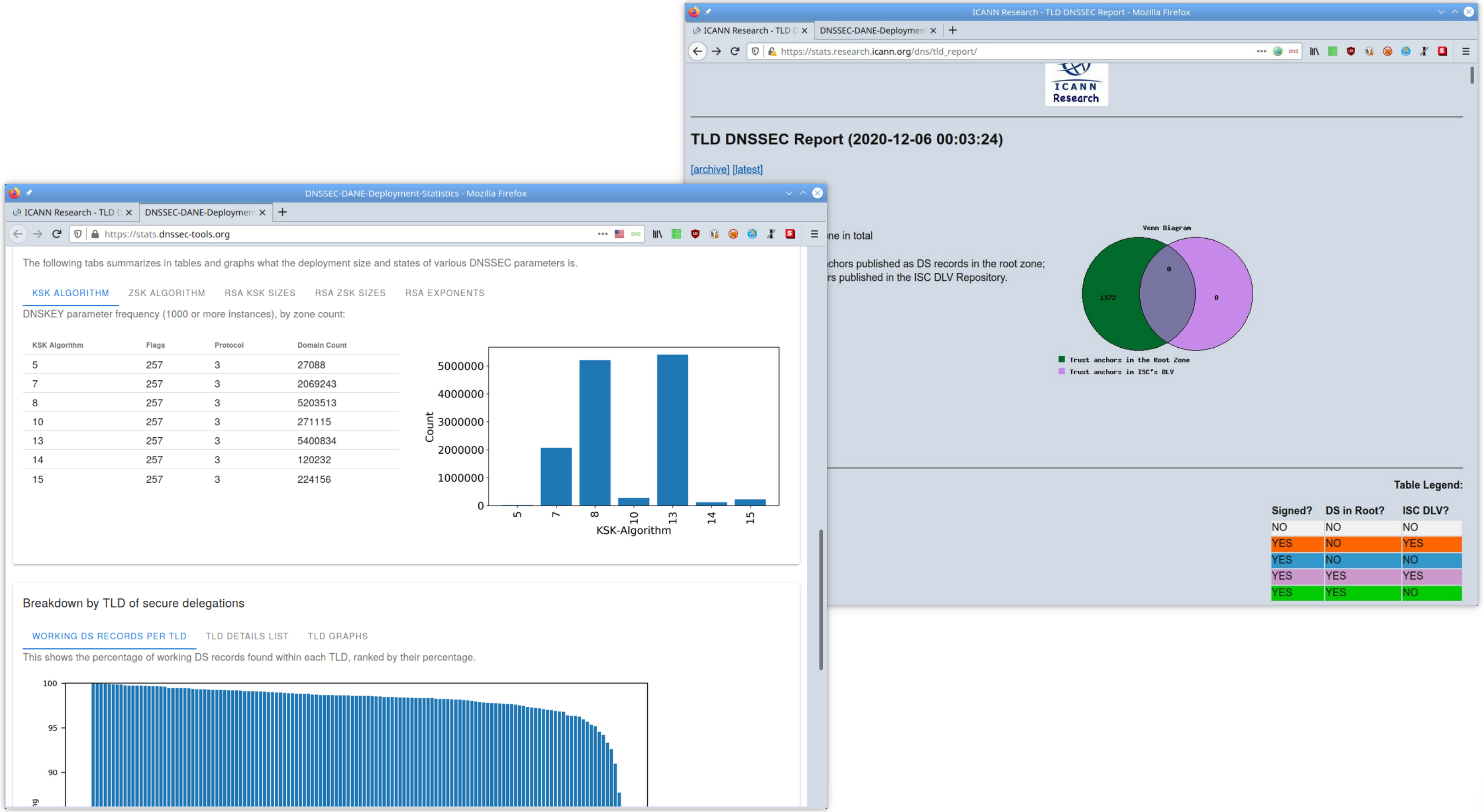
Motivation: What DNSSEC Can Do



Source: https://www.ripe.net/support/training/material/ripe-ncc-training-material#DNSSEC

DNSSEC Deployment

- How many top-level domains (TLDs) are DNSSEC protected?



DNSSEC Part 1: TSIG

www.geant.org

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 731122 (GN4-2).

DNSSEC: Transaction Signature (TSIG)

- Idea: cryptographically sign DNS transactions
- Use cases:
  - Zone Transfers (AXFR, IXFR)
  - Dynamic Updates – the ones available within the DNS protocol
    - Not the web API stuff of Dynamic DNS providers
- In the basic case: use of a shared secret
  - Slightly more advanced: Derive shared secret through GSS-API (RFC 6045)
  - Used in Windows Active Directory
- Advanced case: Public Key SIG0 RR
  - Better suited for use cases with a large number of (non-domain) clients
  - Rarely used
  - Probably not fully supported by all implementations

TSIG at work (live demo)

- Assume a zone, like `example.net`
- We want to generate keys, so that a host can update its A record through dynamic updates

```
() dormal — Konsole <2>
dd          ddbugtopbm
moeller@flaubert:~> ssh dormal
Last login: Tue Dec  1 17:47:49 2020 from 2001:638:714:2c30:6::4
Have a lot of fun...
klaus@dormal:~> /usr/sbin/ddns-confgen --help
ddns-confgen: invalid argument --
Usage:
 ddns-confgen [-a alg] [-k keyname] [-q] [-s name | -z zone]
  -a alg:        algorithm (default hmac-sha256)
  -k keyname:    name of the key as it will be used in named.conf
  -s name:       domain name to be updated using the created key
  -z zone:       name of the zone as it will be used in named.conf
  -q:            quiet mode: print the key, with no explanatory text
klaus@dormal:~> /usr/sbin/ddns-confgen -a sha512 -k flaubert-example -z dyn.example.net
# To activate this key, place the following in named.conf, and
# in a separate keyfile on the system or systems from which nsupdate
# will be run:
key "flaubert-example" {
        algorithm hmac-sha512;
        secret "pJSQIN6lzYy3sjPVLYCazBiN7FB8Dn1ZQY+uF04EzMlU7zdzFv2ZA3/0Y47VrgmdcIwEvx6TSvfhXqxrEsyOIg==";
};

# Then, in the "zone" definition statement for "dyn.example.net",
# place an "update-policy" statement like this one, adjusted as
# needed for your preferred permissions:
update-policy {
          grant flaubert-example zonesub ANY;
};

# After the keyfile has been placed, the following command will
# execute nsupdate using this key:
nsupdate -k <keyfile>
klaus@dormal:~>
```

DFN DEUTSCHES FORSCHUNGSNETZ DFN CERT

GÉANT

DNSSEC Part 2: DNSKEY, DS & RRSIG

www.geant.org

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 731122 (GN4-2).

HOW DNSSEC Works: High Level Overview

- Assume you are the operator of a DNS zone
- You need a public/private keypair (well … two actually)
- Public Key is published as a RR: **DNSKEY**
- The RRs (A, CNAME, MX, etc.) of your zone are signed with the private key
- Signature is put into another RR: **RRSIG**
- Others can now verify that received data is correct by comparing the hash of the received RR with the RRSIG data and the DNSKEY of the zone
- Operator of your parent zone signs your key and puts the signature in (yet) another RR: **DS** (Delegation Signer)
  - You do the the same for all sub-zones of yours
  - And their operators for for their sub-zones, …

HOW DNSSEC Works: High Level Overview (cont.)

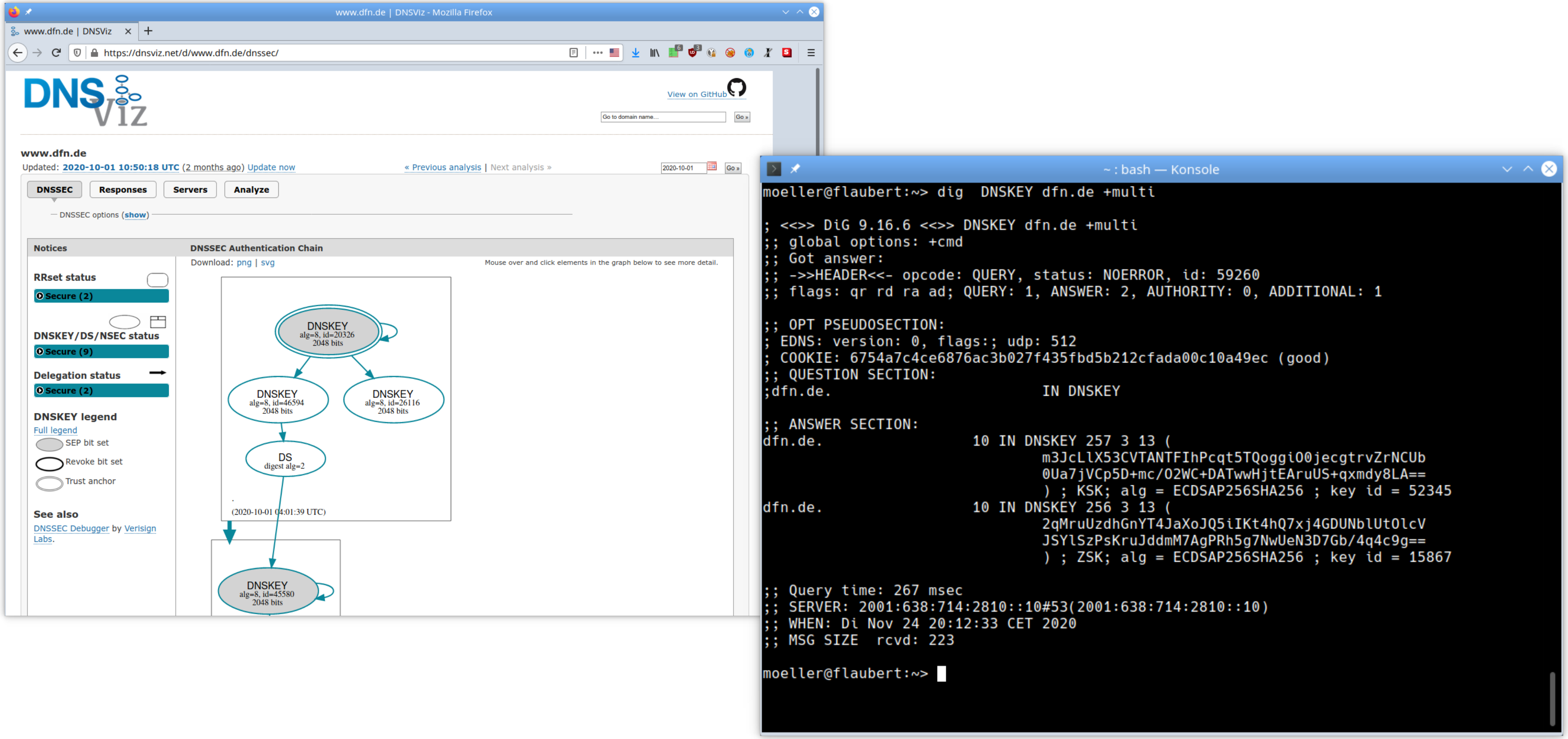
- Security rests with trust in the DNSKEY
- Q: *How do we know the DNSKEY is genuine?*
- A: *Your DNSKEY is signed by the DNSKEY of the parent zone (**DS**)*
  - Can walk all the way up to the root zone
- Q: *How do we trust their key?*
- A: *One key is assumed to be trusted, it's called the **trust anchor***
  - With the public internet, that's the root zones (.) DNSKEY
  - But we can't get that key from the DNS (chicken & egg problem)
  - Has to be delivered out-of-band, i.e. shipped with your OS/nameserver, …

How DNSSEC Works: Chain of Trust

- From the trust anchor, a *Chain of Trust* can be build down to the RRs we received
- Concept similar to that of X.509 Keys and Certificates
- But more limited: Integrity protection only, no encryption
- **DNSSEC is not a PKI!**
  - Policies of parent zones do not apply to child zones
  - No CRLs
- **DNSSEC scope is narrow: Integrity of RR data only**
  - DNSSEC keys work only within DNS, nowhere else
  - Confidentiality of DNS transactions not addressed

DNSSEC in Action (Live Demo)

See DNSSEC live and in graphic detail (`dig`, `https://dnsviz.net/`)



DFN DEUTSCHES FORSCHUNGSNETZ
DFN CERT®

GÉANT

DNSSEC RRs: DNSKEY

- The public key part of the zone's keys

```
> dig DNSSEC dfn.de +multi

…

dfn.de.   10 IN DNSKEY  256 3 13 ( 2qrMuUzdhGnY … 4q4c9g== ) ;
… ; key id = 15867
```

Needed by other DNSSEC RRs

Key Type (Flags)
256: KSK = Key Signing Key
257: ZSK = Zone Signing Key

Protocol (always 3 = DNSSEC)

Algorithm (for signing)

The actual public key (Base64)

DNSKEY Algorithm & Digest Numbers

Number	Description	Zone Sign.	Trans. Sec.
0	Delete DS	N	N
1	RSA/MD5 (deprecated)	N	Y
2	Diffie-Hellman	N	Y
3	DSA/SHA1	Y	Y
5	RSA/SHA-1	Y	Y
6	DSA-NSEC3-SHA1	Y	Y
7	RSASHA1-NSEC3-SHA1	Y	Y
8	RSA/SHA-256	Y	*
10	RSA/SHA-512	Y	*

Number	Description	Zone Sign.	Trans. Sec.
12	GOST R 34.10-2001	Y	*
13	ECDSA Curve P-256 with SHA-256	Y	*
14	ECDSA Curve P-384 with SHA-384	Y	*
15	Ed25519	Y	*
16	Ed448	Y	*
253	private algorithm	Y	Y
254	private algorithm OID	Y	Y

Number	Description
1	SHA-1
2	SHA-256
3	GOST R 34-10.2001
4	SHA-384

DNSSEC RRs: RRSIG

- The RR that will be used to verify that a given RR is genuine

Signing algorithm

No of labels covered

RR type covered

Original TTL

```
> dig  +multi www.dfn.de
…
www.dfn.de.  83582 IN A …
www.dfn.de.  83582 IN RRSIG A 13 3 86400 (20201210212709 20201110202709
15867 dfn.de.  V3RH7 … moepg== )
```

Key id

Signer‘s name (zone name)

Signature data

Signature valid until

Signature valid from (inception)

DFN DEUTSCHES FORSCHUNGSNETZ

DFN CERT®

GÉANT

DNSSEC RRs: DS

- DNSKEY used to publish the public part of the signing key
- RR needed to publish signature of DNSKEYs of child zones: DS
  - The *certificates* of DNSSEC, or figuratively as the *links* in the *chain of trust*
- DS is kept and maintained in the parent zone
  - How a DS is obtained is dependant on the policy of the parent zones operator

```
> dig DS dfn.de +multi
…

dfn.de.  10 IN DS 52345 13 2 (CB45E805D6DE033CD6 … F16)
```

Key id

Signing algorithm

Hash algorithm

Signature data

DFN DEUTSCHES FORSCHUNGSNETZ DFN CERT

GÉANT

DNSSEC: KSK and ZSK

- Why two signing keys?
- One key (ZSK) for daily business (i.e. RR signing)
  - Shorter, for faster signing (there may be millions of RRs),
  - Shorter is less secure, give it a shorte lifetime (will be changed more often)
  - If server is compromised, so will be this key
- One key (KSK) for signing the DNSKEY
  - This key will be signed by the parents zones key (DS RR)
  - Doesn’t need to be fast, can be longer, more secure, has longer lifetime
  - Should be kept very secure (air-gapped, HSM)
  - Compromise of server will not affect this key, so no need for re-signing DNSKEYs
  - Needed only for DS generation and ZSK Key rollover (compromise or expiration)

DNSSEC: Tools of the Trade (BIND)

- Key generation: **`dnssec-keygen`**
- Signing of a zone: **`dnssec-signzone`** or automatically with newer nameservers
- DS generation:
- Zone verification:
- Syntax check: **`named-checkzone, named-checkconf`**
- Debugging: **`dig, delv`**
- If you want a different codebase than BIND
  - ldns (**`drill`**) or
  - DNSSEC-tools (**`validate, donuts`**)

DNSSEC Part 3: NSEC & NSEC3

www.geant.org

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 731122 (GN4-2).

DNSSEC NSEC

- But, … what if an attacker smuggles in an unsigned RR which does not have a signed counterpart?
  - I.e. the RR does not exist in the real zone file
  - It will be unauthenticated, but may still be used (and believed)
- Need a way to tell that an RR does not exist
- Idea (NSEC RR):
  - Sort the records in a zone file
  - For each RR, build a “pointer” to the next (and previous) RR
  - Sign that RR and publish it in the zone file
  - Last records pointer wraps around to first record

DNSSEC: Next SECure (NSEC)

- Next SECure RR
  - NSEC RR contains no signature, but there is an RRSIG for it

```
> dig +multi NSEC www.dfn.de
…

www.dfn.de.  3600 IN NSEC www-dev.dfn.de. A AAAA RRSIG NSEC
```

Next owner name

RR Types present for next name

NSEC3 Problems

- But, if an attacker knows one RR, it could walk along the chain of NSEC RRs and find all RRs in a zone
  - This is called *NSEC walking*
- Solution: Hash the names und build NSEC RRs with the hashes
- NSEC3 does that
  - Unfortunately, it can be easily brute-forced → NSEC5 (draft)
    - Number of iterations has little impact
- Use NSEC or NSEC3?
  - NSEC is much easier to troubleshoot
  - NSEC3 signing takes more reources (bigger RRs too)
  - Is *NSEC walking* a problem? Atackers have other ways to enumerate zones

DNSSEC: NSEC3 RR

- Basically the same RR as NSEC, only with hashes instead of plain names
- Additional information tell how the hash was build

```
B7ab … 206d 900 IN NSEC3 1 0 100 0480 … b390 54 … cf A RRSIG NSEC
```

Hash of owners name (FQDN used)

Hash Algorithm

Flags

Iterations

Hash of next owners name

Salt

RR Types present for next name

- Only use of the Flag field now is to show wether the zone contains unsigned delegations (i.e. sub-zones)
- Again, the RR itself contains no signature, but there is an RRSIG RR for it

NSEC3 Parameters (NSECPARAM)

- Used when verifying that a name does exist or not
- Cant ask directly for the NSEC3 RR, as we don’t know the hash
- Need the parameters: Algorithm, Iterations, Salt (and Flags)
- → NSEC3PARAM

```
example.net 900 IN NSEC3PARAM 1 0 100  54bd921f6ecbbd2534207cf
```

Algorithm

Flags

Iterations

Salt

DNSSEC Part 4: Validating Resolvers

www.geant.org

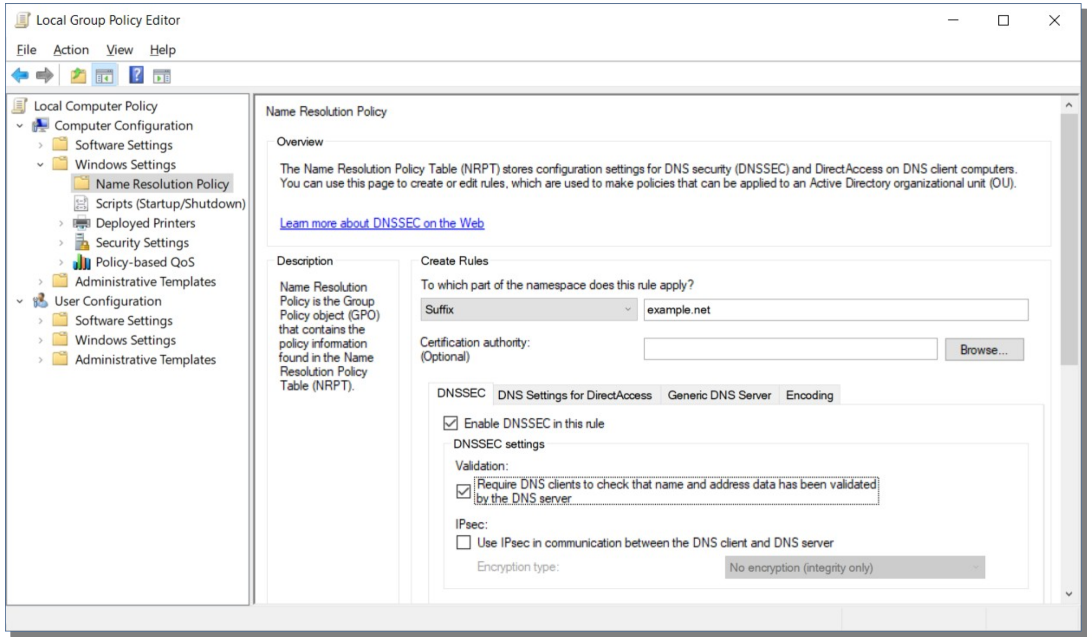
The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 731122 (GN4-2).

Validating Resolvers

- Validation: checking if DNSSEC RRs are present and signatures verify
  - Up to the trust anchor
- Strict Validating Resolver: DNS lookup fails if validation fails
- Opportunistic V. R.: falls back to DNS if no DNSSEC RRs present
  - Queries still fail if DNSSEC RRs are present and signature check fails
- Usually built into a recursive resolver (nameserver)
- OS stub resolvers will (usually) not validate
  - *“dnssec ok”* (do) flag in query requests validation
  - *“checking disabled”* (cd) query flags tells recursive resovler **not** to validate
  - *“authenticated data”* (ad) flag in response indicates validation

Client Configuration

- Linux (GNU libc 2.31)
  - Set **`options edns0 trust-ad`** in **`/etc/resolv.conf`** or use **`RES_OPTIONS="edns0 trust-ad"`** in your shell
  - See “Evaluating Local DNSSEC Validators” for other stub resolvers
    - systemd-resolved, dnsmasq, Knot Resolver, Unbound, PowerDNS Recursor
- Windows
  - Gpedit.msc
    → Windows Settings
    → Name Resolution Policy



Validation Problems

1. Captive Portals
   - Answers all name queries with IP address of the portal
   - Breaks strict validation
   - Can only disable validation until logged on to the portal
   - May have to clear cache too, to remove negative response entries
2. Internal domains (`.corp, .internal, .lan, .fritz.box`)
   - Won't be signed by TLD operators, some will never be
   - Need Negative Trust Anchors (NTA) then
     - Meant as a temporary workaround in case of sth. broken
     - Limited lifetime (1h default in BIND, polls every 5min)
     - BIND: `rndc nta <domain>`

DFN DEUTSCHES FORSCHUNGSNETZ DFN CERT®

GÉANT

Miscellaneous

- Everything is bigger with DNSSEC
  - Zones: 3x to 10x
  - Responses – the DNS protocol had to be upgraded to EDNS
    - TCP used more often
  - CPU: for signing **and** for verifying signatures
- Errors in DNSSEC configuration more severe than …?
- Make plans
  - For key rollover (or revocation)
  - For updating the trust anchors in your resolvers
  - Don't forget to test & train

What have you learned?

- DS ← DNSKEY → RRSIGS, NSEC(3), NSEC3PARAM RRs
  - Zone files and responses get bigger
- Deployment is much further than it used to be
  - Most TLDs are DNSSEC signed
  - Some operators offer automated DS generation
- Process is not easy, but manageable
  - Many technical tasks can be automated
  - Plan for Trust Anchor updates and key rollovers

What has been left out?

- Confidentialitiy of DNS lookups: DNS over TLS/HTTPS/QUIC

Thank you

Any questions?

Next module: ***DNS Privacy Protocols***, 10th of December 2020

www.geant.org

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 731122 (GN4-2).

Tools

- DNSSEC Tools: https://dnssec-tools.org/
- ldns (lots of tools, incl. “drill”): https://www.nlnetlabs.nl/projects/ldns/about/
- Nameserver
  - BIND: https://www.isc.org/bind/
  - Dnsmasq: http://www.thekelleys.org.uk/dnsmasq/doc.html
  - Knot DNS: https://www.knot-dns.cz/
  - NSD: https://www.nlnetlabs.nl/projects/nsd/about/
  - PowerDNS: https://www.powerdns.com/
  - Unbound: https://nlnetlabs.nl/projects/unbound/about/
  - Yadifa: https://www.yadifa.eu/

Websites

- ICANN TLD DNSSEC Report: https://stats.research.icann.org/dns/tld_report/
- DNSSEC statistics from the Internet Society: https://www.internetsociety.org/deploy360/dnssec/statistics/
- DNS Visualizer: https://dnsviz.net/
- DNSSEC Resolver Test: https://dnssec.vs.uni-due.de/
- Operational considerations: https://www.ripe.net/analyse/archived-projects/disi/dnssec-operations-and-security-practice-statement
- IANA root zone signing ceremonies: https://www.iana.org/dnssec/ceremonies/

References

- Draft (2007): Split-View DNSSEC Operational Practices: https://tools.ietf.org/html/draft-krishnaswamy-dnsop-dnssec-split-view-04
- BIND DNSSEC Deployment Guide: https://dnssec-guide.readthedocs.io/en/latest/
- BIND 9 Administrator Reference Manual (ARM): https://bind9.readthedocs.io/
- Blog entries from SWITCH about DNSSEC: https://securityblog.switch.ch/2020/12/01/dnssec-signing-your-domain-with-bind-9-16/
- Glibc Wiki: https://sourceware.org/glibc/wiki/DNSSEC
- Local DNSSEC Validators https://www.redpill-linpro.com/techblog/2019/08/27/evaluating-local-dnssec-validators.html
- M. Wander: Measurement Survey of Server-Side DNSSEC Adoption
  - Paper: https://www.researchgate.net/publication/318980559_Measurement_survey_of_server-side_DNSSEC_adoption
  - Slides: https://archive.icann.org/meetings/icann56/schd.ws/hosted_files/icann562016/f5/Wander-ICANN56-DNSSEC-Adoption-v2.pdf
  - Pres. Video: https://www.youtube.com/watch?v=pa9-dEkIRMM

RFCs (1)

- RFC 2136, Vixie et al.: Dynamic Updates in the Domain Name System (DNS UPDATE), `https://tools.ietf.org/html/rfc2136`
- RFC 2845, Vixie et al.: Secret Key Transaction Authentication for DNS (TSIG), `https://tools.ietf.org/html/rfc2845`
- RFC 2931, Eastlake et al.: DNS Request and Transaction Signatures (SIG(0)s), `https://tools.ietf.org/html/rfc2931`
- RFC 3645, Kwan et al.: Generic Security Service Algorithm for Secret Key Transaction Authentication for DNS (GSS-TSIG), `https://tools.ietf.org/html/rfc3645`
- RFC 3658, Gudmundsson: Delegation Signer (DS) Resource Record (RR), `https://tools.ietf.org/html/rfc3658`
- RFC 4033, Arends et al.: DNS Security Introduction and Requirements, `https://tools.ietf.org/html/rfc4033`
- RFC 4034, Arends et al.: Resource Records for the DNS Security Extensions, `https://tools.ietf.org/html/rfc4034`
- RFC 4035, Arends et al.: Protocol Modifications for the DNS Security Extensions, `https://tools.ietf.org/html/rfc4035`
- RFC 4367, Rosenberg: What's in a Name: False Assumptions about DNS Names, `https://tools.ietf.org/html/rfc4367`
- RFC 4509, Hardaker: Use of SHA-256 in DNSSEC Delegation Signer (DS) Resource Records (RRs), `https://tools.ietf.org/html/rfc4509`
- RFC 4635, Eastlake et al.: HMAC SHA TSIG Algorithm Identifiers, `https://tools.ietf.org/html/rfc4635`

RFCs (2)

- RFC 5011, StJohns: Automated Updates of DNS Security (DNSSEC) Trust Anchors, https://tools.ietf.org/html/rfc5011
- RFC 5155, Laurie et al.: DNS Security (DNSSEC) Hashed Authenticated Denial of Existence, https://tools.ietf.org/html/rfc5155
- RFC 5933, Dolmatov et al.: Use of GOST Signature Algorithms in DNSKEY and RRSIG Resource Records for DNSSEC, https://tools.ietf.org/html/rfc5933
- RFC 6014, Hoffman: Cryptographic Algorithm Identifier Allocation for DNSSEC, https://tools.ietf.org/html/rfc6014
- RFC 6605, https://tools.ietf.org/html/rfc6605
- RFC 6840, Weiler et al.: Clarifications and Implementation Notes for DNS Security (DNSSEC), https://tools.ietf.org/html/rfc6840
- RFC 6895, Eastlake et al.: Domain Name System (DNS) IANA Considerations, https://tools.ietf.org/html/rfc6895
- RFC 6944, Rose: Applicability Statement: DNS Security (DNSSEC) DNSKEY Algorithm Implementation Status, https://tools.ietf.org/html/rfc6944
- RFC 8080, Sury et al.: Edwards-Curve Digital Security Algorithm (EdDSA) for DNSSEC, https://tools.ietf.org/html/rfc8080