A Story on Unsupported DNSSEC Algorithms

Matthijs Mekking, ISC

rm unsupported algorithms

- RFC 6944
- draft-ietf-dnsop-algorithm-update
- Removed in BIND 9:
  - RSAMD5
  - GOST
  - DSA
  - DSA-NSEC3-SHA1

```
700  689          switch (alg) {
701       -       case DNS_KEYALG_RSAMD5:
702  690          case DNS_KEYALG_RSASHA1:
703  691          case DNS_KEYALG_NSEC3RSASHA1:
704  692          case DNS_KEYALG_RSASHA256:
...  ...     @@ -780,7 +768,6 @@ main(int argc, char **argv) {
780  768          }
781  769
782  770          switch(alg) {
783       -       case DNS_KEYALG_RSAMD5:
784  771          case DNS_KEYALG_RSASHA1:
785  772          case DNS_KEYALG_NSEC3RSASHA1:
786  773          case DNS_KEYALG_RSASHA256:
...  ...
```

What to expect?

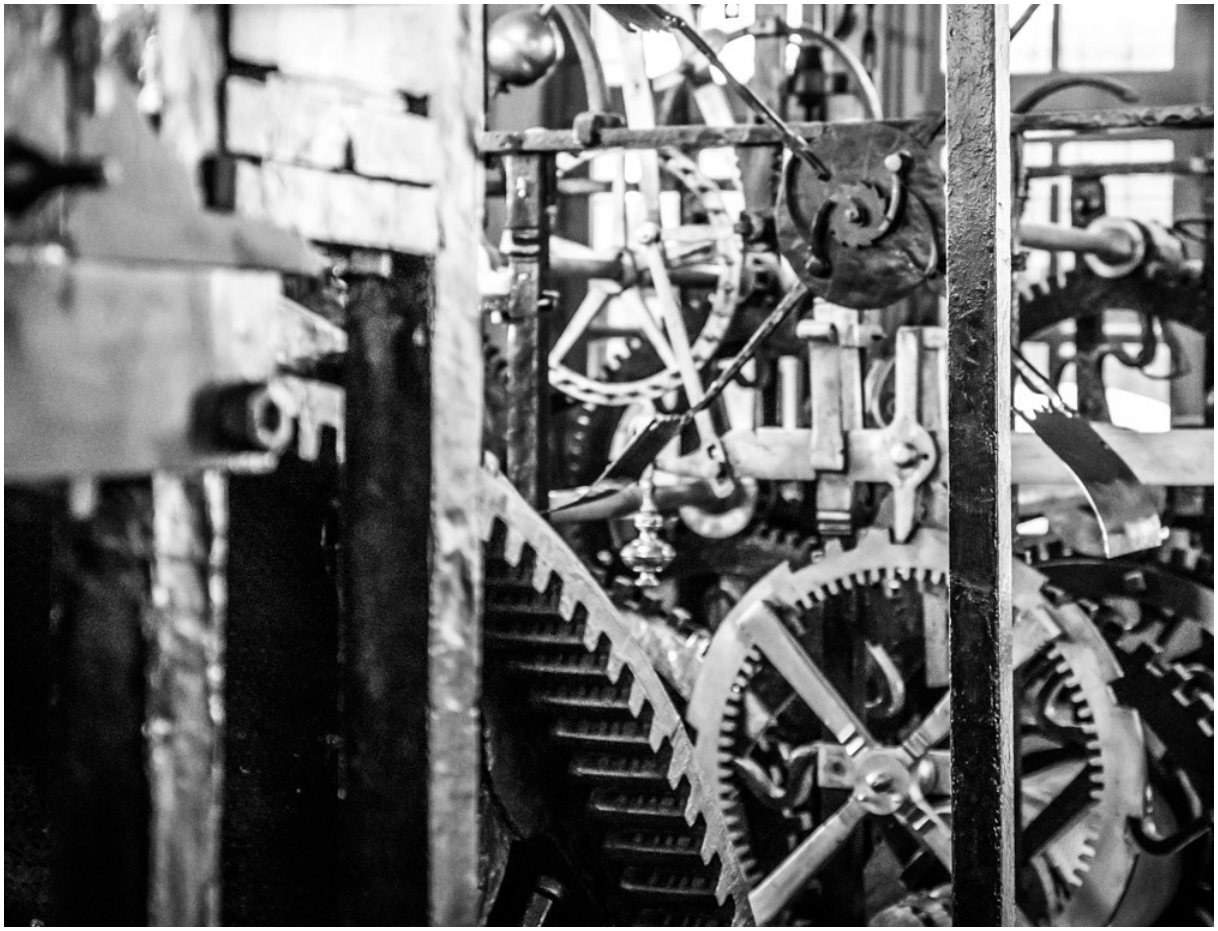
Bogus:
The validating resolver has a trust anchor and a secure delegation indicating that subsidiary data is signed, but the response fails to validate for some reason: missing signatures, expired signatures, signatures with **unsupported algorithms**, data missing that the relevant NSEC RR says should be present, and so forth.

If the resolver **does not support any of the algorithms listed** in an authenticated DS RRset, then the resolver will not be able to verify the authentication path to the child zone. In this case, the resolver SHOULD treat the child zone as if it were unsigned.

If the validator **does not support any of the algorithms listed** in an authenticated DS RRset, then the resolver has no supported authentication path leading from the parent to the child. The resolver should treat this case as it would the case of an authenticated NSEC RRset proving that no DS RRset exists, as described above.

Write tests

- Key creation
- Signing
- Publication
- Validation
- Trust anchor
- DLV
- 5011



Belfort, the bell tower of Bruges

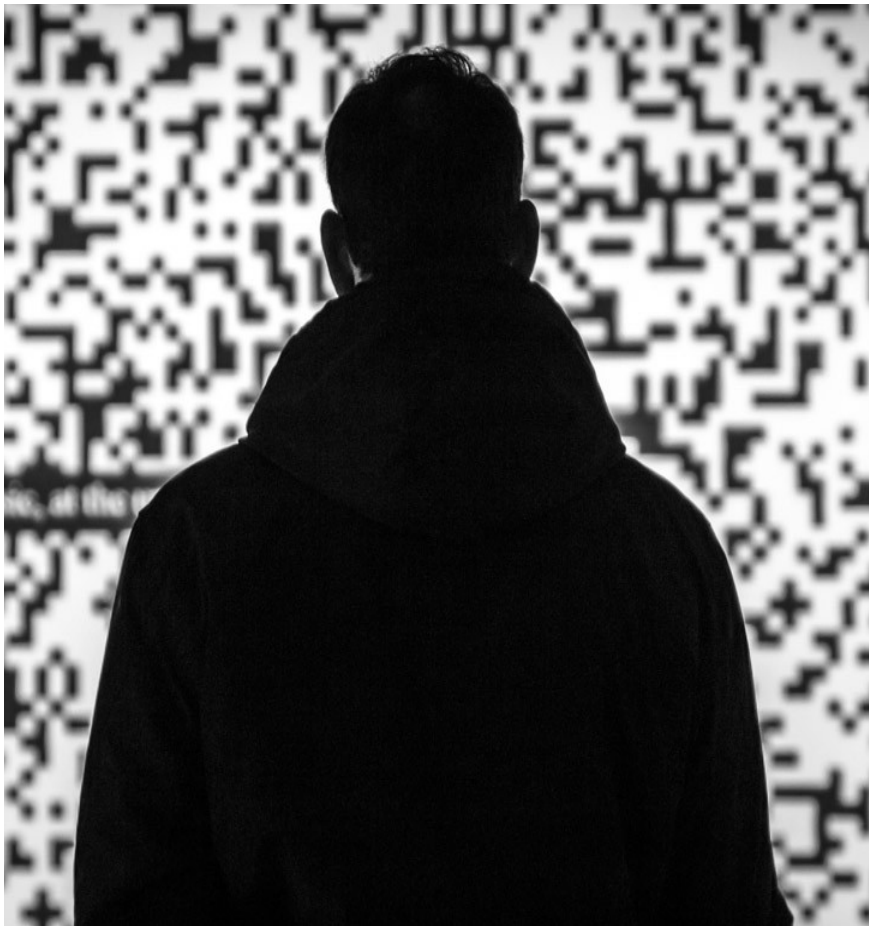
Testing DNS software

- BIND 9.13
- Unbound 1.9
- Knot DNS 2.7.6
- Knot Resolver 3.2.1
- PowerDNS 4.1.6
- PDNS Recursor 4.1.12
- OpenDNSSEC 2.1.3

Beachcombing Museum, Texel

Observations

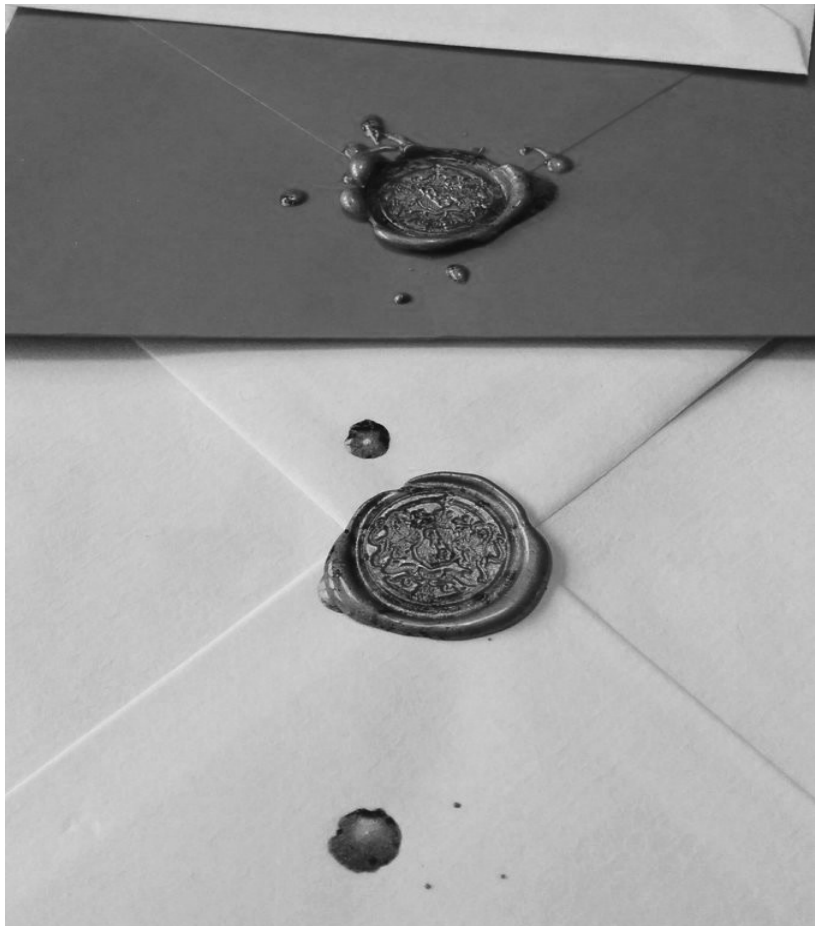
- **The Good**
  Most test pass
- **The Bad**
  Some quirky behavior
- **The Ugly**
  Found some crashes



Deletion Process: Only You Can See My History (Oddstream)

Test signers

- dnssec-keygen, etc
  - example.com
  - RSAMD5, 255
- dnssec-signzone
- Automated DNSSEC
- Publish 255 DNSKEY



Envelope sealing

[1] Key creation should fail

```
$ dnssec-keygen -a RSAMD5 example.com.
dnssec-keygen: fatal: unsupported algorithm: 1
```

```
$ sudo keymgr -c knot.sample.conf example.com. generate algorithm=255
Error (invalid key algorithm)
$ keymgr -c knot.sample.conf example.com. generate algorithm=RSAMD5
Unknown algorithm: RSAMD5
Error (invalid parameter)
```

```
pdnsutil generate-zone-key
Syntax: pdnsutil generate-zone-key zsk|ksk [rsasha1|rsasha256|rsasha512|
gost|ecdsa256|ecdsa384] [bits]
pdnsutil generate-zone-key ksk rsamd5 768
Generating a KSK with algorithm = 1
Requesting specific key size of 768 bits
Error: Request to create key object for unknown algorithm number 1
```


```
ods-enforcerd: 1 zone(s) found on policy "rsamd5"
ods-enforcerd: [hsm_key_factory_generate] 53 keys needed for 1 zones
covering 31536000 seconds, generating 53 keys for policy rsamd5
ods-enforcerd: 53 new ZSK(s) (768 bits) need to be created.
ods-enforcerd: [hsm_key_factory_generate] key generation failed
```

ISC

[1] Key creation should fail

```
$ dnssec-keygen -a RSAMD5 example.com.
dnssec-keygen: fatal: unsupported algorithm: 1
```

```
$ sudo keymgr -c knot.sample.conf example.com. generate algorithm=255
Error (invalid key algorithm)
$ keymgr -c knot.sample.conf example.com. generate algorithm=RSAMD5
Unknown algorithm: RSAMD5
Error (invalid parameter)
```

```
pdnsutil generate-zone-key
Syntax: pdnsutil generate-zone-key zsk|ksk [rsasha1|rsasha256|rsasha512|
gost|ecdsa256|ecdsa384] [bits]
pdnsutil generate-zone-key ksk rsamd5 768
Generating a KSK with algorithm = 1
Requesting specific key size of 768 bits
Error: Request to create key object for unknown algorithm number 1
```

```
ods-enforcerd: 1 zone(s) found on policy "rsamd5"
ods-enforcerd: [hsm_key_factory_generate] 53 keys needed for 1 zones
covering 31536000 seconds, generating 53 keys for policy rsamd5
ods-enforcerd: 53 new ZSK(s) (768 bits) need to be created.
ods-enforcerd: [hsm_key_factory_generate] key generation failed
```

[2] Signing should fail

```
$ dnssec-signzone foo.example.db Kfoo.example.+001+53634
dnssec-signzone: fatal: cannot load dnskey Kexample.+001+53634:
algorithm is unsupported

Configuration excerpt:

    zone "foo.example" IN {
        type master;
        file "db/foo.example.db";
        key-directory "keys/";
        auto-dnssec maintain;
    }

dns_dnssec_findmatchingkeys: error reading key file Kfoo.example.
+001+53634.private: algorithm is unsupported
```

```
pdnsutil add-zone-key foo.example. ksk 768 active rsamd5
Error: Request to create key object for unknown algorithm number 1
```

[2] Signing should fail

Configuration excerpt:

```
    policy:
      - id: unsupported
        algorithm: 255
```

```
error: config, file 'knot.conf', line 20, item'algorithm', value '255'
(invalid parameter)
```

```
    policy:
      - id: dsa
        algorithm: RSAMD5
```

```
error: config, file 'knot.conf', line 26, policy 'rsamd5' (RSAMD5
algorithm no longer supported)
```

See [1]

[3] Allow publication

```
example.com.    3600    IN    DNSKEY    257 3 13 ...
example.com.    3600    IN    DNSKEY    257 3 255 ...
example.com.    3600    IN    RRSIG     DNSKEY 13 2 3600 (
                                        20190319133330 20190219133330 9826
                                        example.com. … )
```

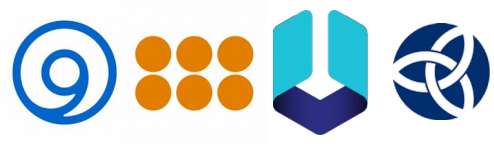
Test validators

- Basic query
- Trust anchors
- RFC 5011
- DLV

The Inspector Cluzo at Valkhoffestival

[4] Validation status insecure

$ dig test.foo.example TXT



example. SOA ...

...

foo.example. IN NS ns.foo.example.

foo.example. IN DS 303 1 2 ...

foo.example. SOA ...

foo.example. IN DNSKEY 257 3 1 ...

...

test.foo.example. IN TXT ...

test.foo.example. IN RRSIG TXT 1 3 ...

[4] Validation status insecure

```
;; ANSWER SECTION:
test.foo.example. 60 IN    TXT "I am a zone that uses an unsupported
DNSSEC algorithm"
```

```
validating test.foo.example/TXT: no supported algorithm/digest (example.com/DS)
validating test.foo.example/TXT: marking as answer (proveunsecure (5))
```

```
info: Verified that response is INSECURE
```

```
AD: request NOT classified as SECURE
```

```
Starting validation of answer to test.foo.example|TXT for 127.0.0.1:42959
Answer to test.foo.example|TXT for 127.0.0.1:42959 validates as Insecure
```

[5] Trust anchors are ignored

$ dig test.foo.example TXT

trust-anchor:
foo.example. DNSKEY 257 3 1

example. SOA …

…
foo.example. IN NS ns.foo.example.
foo.example. IN DS 303 1 2 …

foo.example. SOA …
foo.example. IN DNSKEY 257 3 1 …
…
test.foo.example. IN TXT …
test.foo.example. IN RRSIG TXT 1 3 …

[5] Trust anchors are ignored

```
skipping trusted key for 'foo.example.': algorithm is unsupported
skipping managed key for 'foo.example.': algorithm is unsupported
```

```
warning: unsupported algorithm for trust anchor foo.example. DNSKEY IN
warning: trust anchor foo.example. has no supported algorithms, the
anchor is ignored (check if you need to upgrade unbound and openssl)
```

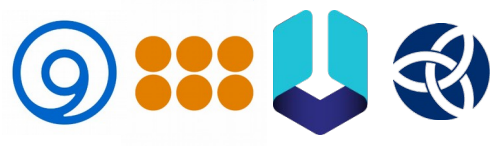
```
> trust_anchors.add_file('trustanchors.conf')
nil

> PANIC: unprotected error in call to Lua API
(/usr/local/lib/kdns_modules/trust_anchors.lua:169: invalid RR:
foo.example. 60   DNSKEY   257 3 1 ...: invalid key algorithm)
```

```
$ rec_control add-ta foo.example 1822 1 1 ...
Added Trust Anchor for foo.example with data 1822 1 1 ...
```

[6] Algorithm rollover

$ dig test.foo.example TXT



managed-key:
foo.example. DNSKEY 257 3 13

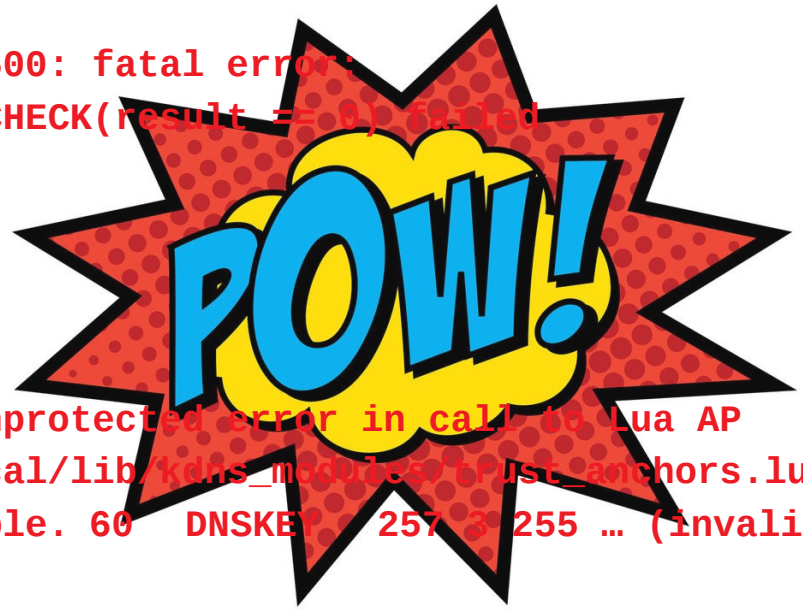
example. SOA ...
...
foo.example. IN NS ns.foo.example.
foo.example. IN DS 303 13 2 ...

foo.example. SOA ...
foo.example. IN DNSKEY 257 3 13 ...
foo.example. IN DNSKEY 257 3 255 ...
...
test.foo.example. IN TXT ...
test.foo.example. IN RRSIG TXT 13 3 ...

[6] Algorithm rollover

```
info: resolving foo.example. DNSKEY IN
info: trust point has unsupported algorithm at foo.example. DNSKEY IN
…
info: trust point was revoked foo.example. DNSKEY IN
```

zone.c:9600: fatal error:
RUNTIME_CHECK(result == 0) failed



PANIC: unprotected error in call to Lua AP
(/usr/local/lib/kdns_modules/trust_anchors.lua:208: invalid RR:
foo.example. 60 DNSKEY 257 3 255 … (invalid key algorithm)

[6] Algorithm rollover

```
info: resolving foo.example. DNSKEY IN
info: trust point has unsupported algorithm at foo.example. DNSKEY IN
…
info: trust point was revoked foo.example. DNSKEY IN
```

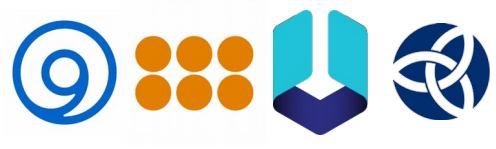
zone.c:9600: fatal error:
RUNTIME_CHECK(result == 0) failed

```
result = compute_tag(keyname, &dnskey, mctx, &keytag);
RUNTIME_CHECK(result == ISC_R_SUCCESS);
```

PANIC: unprotected error in call to Lua AP
(/usr/local/lib/kdns_modules/trust_anchors.lua:208: invalid RR:
foo.example. 60 DNSKEY 257 3 255 … (invalid key algorithm)

[7] DLV records are ignored

$ dig test.foo.example TXT



dlv-anchor:
dlv.example. DS 257 3 13

example. SOA ...
...
foo.example. IN NS ns.foo.example.
foo.example. IN DS 303 1 2 ...

foo.example. SOA ...
foo.example. IN DNSKEY 257 3 1 ...
...
test.foo.example. IN TXT ...
test.foo.example. IN RRSIG TXT 1 3 ...

dlv.example. SOA ...
...
foo.example.dlv.example IN DLV 303 1 2 ...

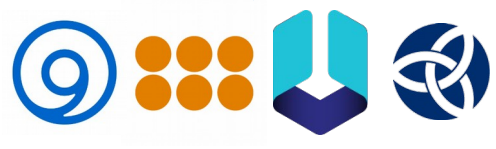
[7] DLV records are ignored

```
validating test.foo.example/TXT: looking for DLV foo.example.dlv.example
validating test.foo.example/TXT: DLV  found with no supported algorithms
validating test.foo.example/TXT: marking as answer (dlvfetched (2))
```

```
info: resolving foo.example.dlv.example. DLV IN
info: resolving foo.example. DNSKEY IN
info: Verified that response is INSECURE
```

[8] dlv trust anchors?

$ dig test.foo.example TXT



dlv-anchor:
dlv.example. DS 257 3 1

example. SOA ...
...
foo.example. IN NS ns.foo.example.
foo.example. IN DS 303 13 2 ...

foo.example. SOA ...
foo.example. IN DNSKEY 257 3 13 ...
...
test.foo.example. IN TXT ...
test.foo.example. IN RRSIG TXT 13 3 ...

dlv.example. SOA ...
...
foo.example.dlv.example IN DLV 303 13 2 ...

[8] DLV trust anchors?

```
ignoring trusted key for 'dlv.example.': algorithm is unsupported
…
validating test.foo.example/TXT: keyvalidated: got broken trust chain
client @0x7f135c0109c0 10.53.0.1#34299 (test.foo.example): query failed
(broken trust chain) for test.foo.example/IN/TXT at query.c:6784
fetch completed at resolver.c:5492 for test.foo.example/TXT in 0.018430:
broken trust chain/broken trust chain [domain:foo.example,referral:0...]
```

```
info: warning: unsupported algorithm for trust anchor dlv.example. DNSKEY IN
warning: trust anchor dlv.example. has no supported algorithms, the anchor
is ignored (check if you need to upgrade unbound and openssl)
…
info: NSEC3s for the referral proved no DS.
info: Verified that response is INSECURE
```

[8] dlv trust anchors?

- BIND 9
  - `dnssec-lookaside "foo.example." \`
    `trust-anchor "dlv.example";`
  - “foo.example” is an explicit trust point
  - Unsupported algorithm is ignored, trust point is empty
- Unbound
  - “The DLV configured is used as a root trusted DLV, this means that it is a lookaside for the root.”

Test results

Unsupported algorithm	BIND 9	Knot	PowerDNS	Unbound	OpenDNSSEC
[1] Key creation should fail	V	V	V	-	V*
[2] Signing should fail	V	V	V	-	V
[3] Publication is allowed	V	V	V	-	V
[4] Validation status insecure	V	V	V	V	-
[5] Trust anchors are ignored	V	X**	X***	V	-
[6] 5011 rollover go to insecure	X*	X**	-	V	-
[7] DLV records are ignored	V	-	-	V	-
[8] DLV anchors?	?	-	-	?	-

** BIND 9: Fixed in 9.11.5-P4, 9.12.3-P4, 9.13.6*

*** Knot Resolver: Fixed in 4.0.0*

**** PowerDNS Recursor: Issue reported*

** OpenDNSSEC: Issue reported*

Why this story?

- Improve BIND 9 tests
- Improve the code
- Learn about weird cases
- Show boring results
- Share thoughts & experiences

Black Honey, Merleyn

Implementation status

	BIND 9	Knot	PowerDNS	Unbound	OpenDNSSEC
RSAMD5	NO	NO	NO	NO	NO
*DSA**	NO	NO	NO	YES	YES
RSASHA1	YES	YES	YES	YES	YES
*DSA-NSEC3-SHA1**	NO	NO	NO	YES	YES
RSASHA1-NSEC3-SHA1	YES	YES	YES	YES	YES
RSASHA256	YES	YES	YES	YES	YES
RSASHA512	YES	YES	YES	YES	YES
*ECC-GOST**	NO	NO	NO	YES	YES
ECDSAP256SHA256	YES	YES	YES	YES	YES
ECDSAP384SHA384	YES	YES	YES	YES	YES
ED25519	NO	YES	YES (4.2)	YES	YES
ED448	NO	NO	YES (4.2)	YES	YES

Recap

- Unsupported algorithms handling
  - Overall quite good
  - A bit of quirkiness
  - Hard to exploit crashes
- Some undefined cases
  - Clarifications needed?
- Implementation guidelines change
  - Bye bye DSA, GOST

A Dyn dog

Thank you for your attention

- Plans for BIND 9.15
  - DNSSEC Made Easy
    - Sane algorithms
    - Fully automated
    - Offline KSK
    - CSK support
    - …
  - Come talk to me :)
    - matthijs@isc.org

Terschelling