ECDSA P-256 support in DNSSEC-validating Resolvers

Geoff Huston
APNIC Labs
March 2016

ECDSA

Elliptic Curve Cryptography allows for the construction of “strong” public/private key pairs with key lengths that are far shorter than equivalent strength keys using RSA

> “256-bit ECC public key should provide comparable security to a 3072-bit RSA public key” *

* http://en.wikipedia.org/wiki/Elliptic_curve_cryptography

ECDSA

And the DNS protocol has some sensitivities over size when using UDP

- UDP fragmentation has its issues in both V4 and V6

* http://en.wikipedia.org/wiki/Elliptic_curve_cryptography

ECDSA vs RSS

```
$ dig +dnssec u5221730329.s1425859199.i5075.vcf100.5a593.y.d

; <<>> DiG 9.9.6-P1 <<>> +dnssec u5221730329.s1425859199.i50
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 61126
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 4, ADI

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;u5221730329.s1425859199.i5075.vcf100.5a593.y.dotnxdomain.ne

;; ANSWER SECTION:
u5221730329.s1425859199.i5075.vcf100.5a593.y.dotnxdomain.net
u5221730329.s1425859199.i5075.vcf100.5a593.y.dotnxdomain.net

;; AUTHORITY SECTION:
ns1.5a593.y.dotnxdomain.net. 1       IN      NSEC    x.5a593.y
ns1.5a593.y.dotnxdomain.net. 1       IN      RRSIG   NSEC 13 5
5a593.y.dotnxdomain.net. 3598 IN     NS      ns1.5a593.y.dotn
5a593.y.dotnxdomain.net. 3600 IN     RRSIG   NS 13 4 3600 202

;; Query time: 1880 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Mar 12 03:59:42 UTC 2015
;; MSG SIZE  rcvd: 527
```

ECDSA signed response – 527 octets

```
$ dig +dnssec u5221730329.s1425859199.i5075.vcf100.5a593.z.dotnxdomain.n

; <<>> DiG 9.9.6-P1 <<>> +dnssec u5221730329.s1425859199.i5075.vcf100.5a
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 25461
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 4, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;u5221730329.s1425859199.i5075.vcf100.5a593.z.dotnxdomain.net. IN A

;; ANSWER SECTION:
u5221730329.s1425859199.i5075.vcf100.5a593.z.dotnxdomain.net. 1    IN A 19
u5221730329.s1425859199.i5075.vcf100.5a593.z.dotnxdomain.net. 1    IN RRSI

;; AUTHORITY SECTION:
33d23a33.3b7acf35.9bd5b553.3ad4aa35.09207c36.a095a7ae.1dc33700.103ad556.
33d23a33.3b7acf35.9bd5b553.3ad4aa35.09207c36.a095a7ae.1dc33700.103ad556.
5a593.z.dotnxdomain.net. 3599 IN     NS      nsz1.z.dotnxdomain.net.
5a593.z.dotnxdomain.net. 3600 IN     RRSIG   NS 5 4 3600 20200724235900 20

;; Query time: 1052 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Mar 12 03:59:57 UTC 2015
;; MSG SIZE  rcvd: 937
```

RSA signed response – 937 octets

So let's use ECDSA for DNSSEC

Yes!

Let’s do that right now!

So lets use ECDSA for DNSSEC

Or maybe we should look before we leap...

- Is ECDSA a “well supported” crypto protocol?
- If you signed using ECDSA would resolvers validate the signature?

The Test Environment

We use the Google Ad network in to deliver a set of DNS tests to clients to determine whether (or not) they use DNSSEC validating resolvers

We use 5 tests:

1. no DNSSEC-signature at all
2. DNSSEC signature using RSA-based algorithm
3. DNSSEC signature using broken RSA-based algorithm
4. DNSSEC signature using ECDSA P-256 algorithm
5. DNSSEC signature using broken ECDSA P-256 algorithm

The Test Environment

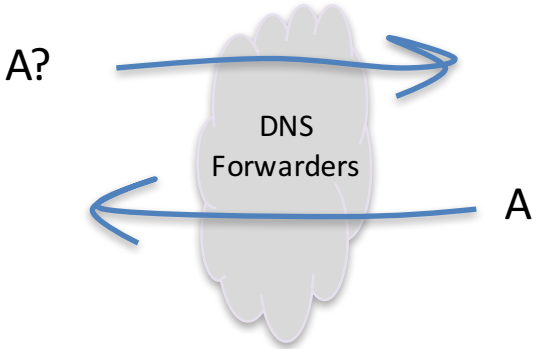
d.t10000.u2045476887.s1412035201.i5053.vne0001.4f167.z.dashnxdomain.net	Unsigned
e.t10000.u2045476887.s1412035201.i5053.vne0001.4f167.z.dotnxdomain.net	RSA Signed
f.t10000.u2045476887.s1412035201.i5053.vne0001.4f168.z.dotnxdomain.net	RSA signed (Badly)
m.t10000.u2045476887.s1412035201.i5053.vne0001.4f167.y.dotnxdomain.net	ECDSA-Signed
n.t10000.u2045476887.s1412035201.i5053.vne0001.4f168.y.dotnxdomain.net	ECDSA-Signed (bad!)

Mapped to a wildcard in the zone file

Unique Signed Zone

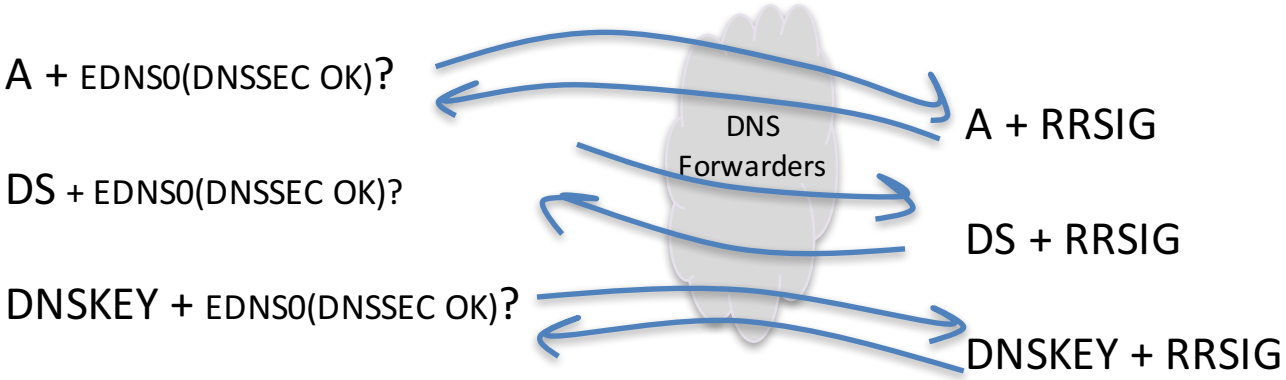
A Naive View

A non-DNSSEC-validating resolver query:



Seen: Single A Query

A DNSSEC-Validating resolver query:



Seen: A, DS, DNSKEY Queries

Theory: DNSSEC Validating Queries

e.t10000.u2045476887.s1412035201.i5053.vne0001.4f167.z.dotnxdomain.net

1. Query for the A resource record with EDNS0, DNSSEC-OK

   query: e.t10000.u204546887.s1412035201.i5053.vne0001.4f167.z.dotnxdomain.net IN A +ED

2. Query the parent domain for the DS resource record

   query: 4f167.z.dotnxdomain.net IN DS +ED

3. Query for the DNSKEY resource record

   query: 4f167.z.dotnxdomain.net IN DNSKEY +ED

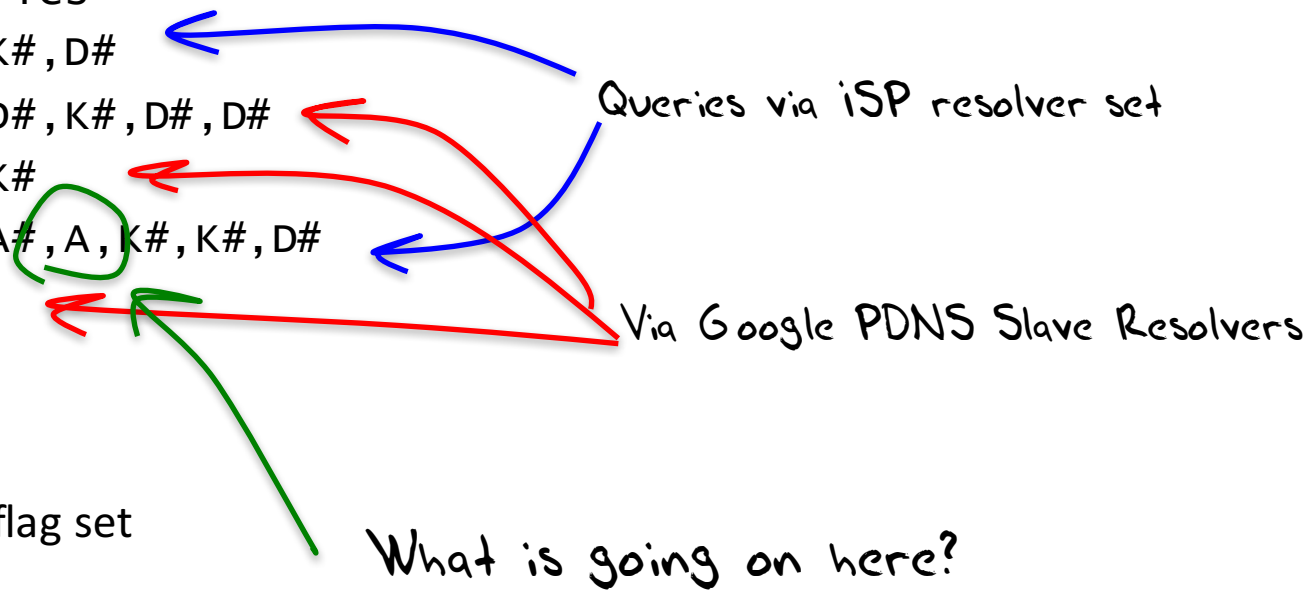
Practice: The DNS is "messy"

- Clients typically use multiple resolvers, and use local timeouts to repeat the query across these resolvers
- Resolvers may use slave farms, so that queries from a common logical resolution process may be presented to the authoritative name server from multiple resolvers, and each slave resolver that directs queries to servers may present only a partial set of validation queries
- Resolvers may use forwarding resolvers, and may explicitly request checking disabled to disable the forwarding resolver from performing validation itself
- Clients and resolvers have their own independent retry and abandon timers

DNS Mess!

Queries for a single badly signed (RSA) name:

```
Resolver        Queries
200.55.224.68:  A#,K#,D#
74.125.19.147:  A#,D#,K#,D#,D#
74.125.19.145:  K#,K#
200.55.224.67:  A#,A#,A,K#,K#,D#
74.125.19.148:  D#
```



#: EDNS(0) DNSSEC OK flag set

DNS Mess!

Queries for a single badly signed (RSA) name:

Resolver	Queries
200.55.224.68:	A#,K#,D#
74.125.19.147:	A#,D#,K#,D#,D#
74.125.19.145:	K#,K#
200.55.224.67:	A#,A#,A,K#,K#,D#
74.125.19.148:	D#

Failed validation (SERVFAIL) from the initial query to ISP resolver causes client to ask Google PDNS resolver

Failed validation appears to cause client to repeat the query to Google PDNS 2 further times

Failed validation appears to cause client to repeat the query to ISP's resolver 2 (or 3?) further times

No clue why this is an orphan DS query!

#: EDNS(0) DNSSEC OK flag set

First Approach to answering the ECDSA question - Statistical Inference

- A DNSSEC-aware resolver encountering a RR with an attached RRSIG that uses a known algorithm will query for DS and DNSKEY RRs (in either order!)

- A DNSSEC-aware resolver encountering a RR with an attached RRSIG that uses an unknown/unsupported crypto algorithm appears to query only for DS RR (and NOT the DNSKEY RR)

Results

Over 45 days in December 2015 – January 2016 we saw:

765,257,019 completed experiments

208,980,333 experiments queried for the DNSKEY RR of a validly signed (RSA) domain **(27.3%**)

183,240,945 experiments queried for the DNSKEY RR of a validly signed (ECDSA) domain (**23.9%**)

Results

Over 45 days in December 2015 – January 2016 we saw:

765,257,019 completed experiments

208,980,333 experiments queried for the DNSKEY RR of a validly signed (RSA) domain **(27.3%**)

183,240,945 experiments queried for the DNSKEY RR of a validly signed (ECDSA) domain (**23.9%**)

If we assume that the DNSKEY query indicates that the resolver "recognises" the crypto algorithm, then it appears that there is a fall by 19.5% in validation when using ECDSA

1 in 5 experiments that fetched the RSA-signed DNSKEY did not fetch the ECDA-signed DNSKEY

That's better than it was…

Over 22 days in September 2014 we saw:

- 3,773,420 experiments
- 937,166 experiments queried for the DNSKEY RR of a validly signed (RSA) domain (**24.8%**)
- 629,726 experiments queried for the DNSKEY RR of a validly signed (ECC) domain (**16.6%**)

1 in 3 experiments that fetched the DNSKEY in RSA did not fetch the ECDSA-signed DNSKEY

Protocol Recognition

- When does the resolver "recognise" the signing protocol?
  - RRSIG field?
  - DS RR?
  - DNSKEY RR?

September 2014 data

Experiments	ECDSA DS	ECDSA DNSKEY	RSA DS	RSA DNSKEY
11,988,195	2,957,855	2,391,298	2,963,888	2,970,902

Protocol Recognition

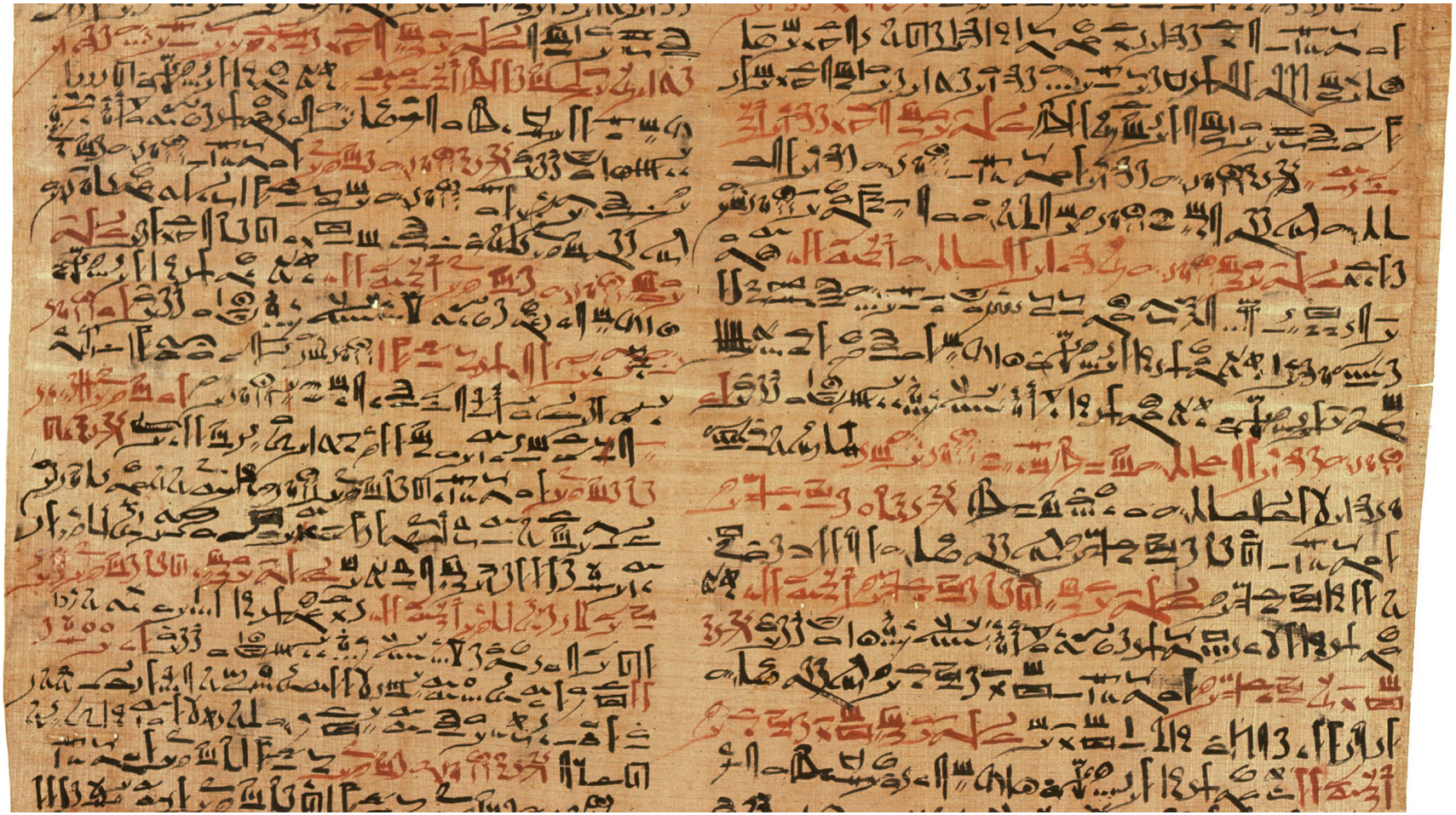
- When does the resolver "recognise" the signing protocol?
  - RRSIG field? ✗
  - DS RR? ✓
  - DNSKEY RR? ✗

September 2014 data

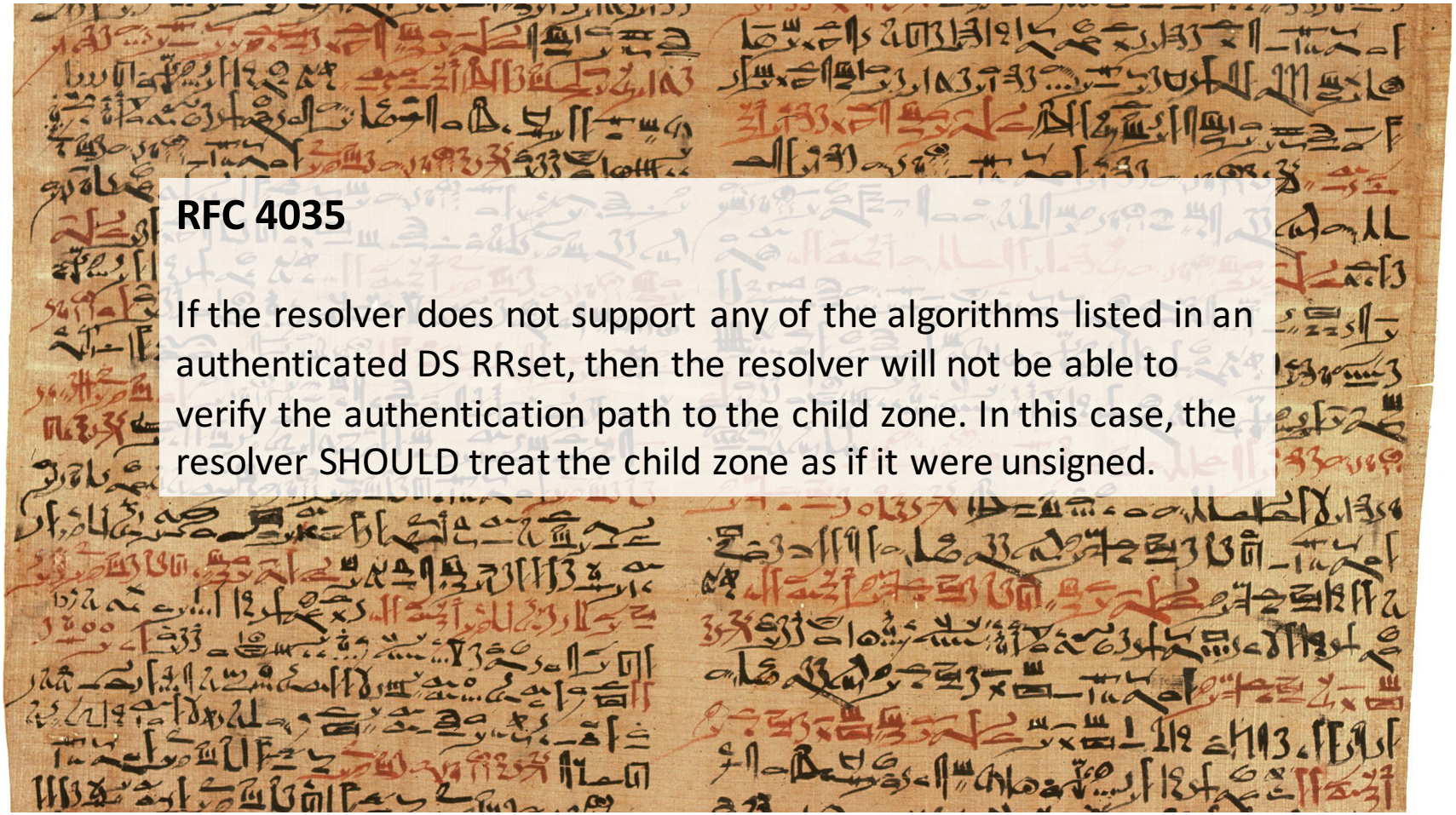
Experiments	ECDSA DS	ECDSA DNSKEY	RSA DS	RSA DNSKEY
11,988,195	2,957,855	2,391,298	2,963,888	2,970,902

This indicates that a validating resolver appears to fetch the DS RR irrespective of the signing protocol, and only fetches the DNSKEY RR if it recognizes the zone signing protocol.

The Words of the Ancients



The Words of the Ancients



RFC 4035

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.

DNS resolver failure modes for an unknown signing algorithm

If a DNSSEC-Validating resolver receives a response DS with an unknown crypto algorithm does it:

- ❑ Immediately stop resolution and return a status code of SERVFAIL?
- ❑ Fetch the DNSKEY RR and then return a status code of SERVFAIL?
- ❑ Abandon validation and just return the unvalidated query result?

DNS resolver failure modes for an unknown signing algorithm

If a DNSSEC-Validating resolver receives a response DS with an unknown crypto algorithm does it:

- ❑ Immediately stop resolution and return a status code of SERVFAIL?
- ❑ Fetch the DNSKEY RR and then return a status code of SERVFAIL?
- ☒ Abandon validation and just return the unvalidated query result?

If the resolver doesn't recognize the protocol in the DS record then there is no point in pulling the DNSKEY record

Second Approach to answering the ECDSA question - DNS + WEB

Data collection: 1/1/16 – 16/2/16

64,948,234 clients who appear to be exclusively using RSA DNSSEC-Validating resolvers

ECC Results:

Success:	82%	53,514,518	Saw fetches of the ECC DNSSEC RRs and the well-signed named URL, but not the badly signed named URL
Failure (fetched both URLs):			
Mixed Resolvers	1.9%	1,218,240	Used both ECDSA-Validating and non-validating resolvers
NO ECC	13.0%	8,461,551	Saw A, DS, no DNSKEY, fetched both URLs
Mixed	0.5%	352,914	Saw some DNSSEC queries, fetched both URLs
No Validation	2.2%	1,401,011	Did not fetch any DNSSEC RRs
Apparent Fail:	**17.6%**	**11,433,716**	

1 in 6 clients that use resolvers that perform DNSSEC validation with RSA fail to validate with ECDSA

Results

- These results show that 82% of clients who appeared to exclusively use RSA DNSSEC-Validating resolvers were also seen to perform validation using ECDSA
- Two thirds of the the remaining clients fetched both objects (13% of the total), but did not fetch any DNSKEY RRs.
- Of the remainder (5%), most were using a validating resolver (which returned SERVFAIL for the badly signed object), and then the client failed over to a non-validating resolver *

* This is curious, because these clients did not failover to a non-validating resolver on a badly signed RSA structure

Is ECDSA a viable crypto algorithm for DNSSEC?

If the aim is to detect efforts to compromise the DNS for the signed zone, then signing a zone with ECDSA limits the number of DNS resolvers who will validate the signature

Which is a shame, because the shorter key lengths could be attractive for DNS over UDP

ECDSA in the (semi-)wild

```
$ dig +dnssec www.cloudflare-dnssec-auth.com

; <<>> DiG 9.9.6-P1 <<>> +dnssec www.cloudflare-dnssec-auth.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 7049
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 6, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;www.cloudflare-dnssec-auth.com.  IN    A

;; ANSWER SECTION:
www.cloudflare-dnssec-auth.com.   300 IN      A     104.20.23.140
www.cloudflare-dnssec-auth.com.   300 IN      A     104.20.21.140
www.cloudflare-dnssec-auth.com.   300 IN      A     104.20.19.140
www.cloudflare-dnssec-auth.com.   300 IN      A     104.20.22.140
www.cloudflare-dnssec-auth.com.   300 IN      A     104.20.20.140
www.cloudflare-dnssec-auth.com.   300 IN      RRSIG A 13 3 300 20150317021923 20150315001923 35273
cloudflare-dnssec-auth.com. pgBvfQkU4Il8ted2hGL9o8NspvKksDT8/jvQ+4o4h4tGmAX0fDBEoorb
tLiW7mcdOWYLoOnjovzYh3QOOdu0Xw==

;; Query time: 237 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Mon Mar 16 01:19:24 UTC 2015
;; MSG SIZE  rcvd: 261
```

Thanks!