Measurement of DNSSEC Validation with RSA-4096

Geoff Huston, Joao Damas
APNIC Labs
November 2021

RSA is not a “dense” algorithm

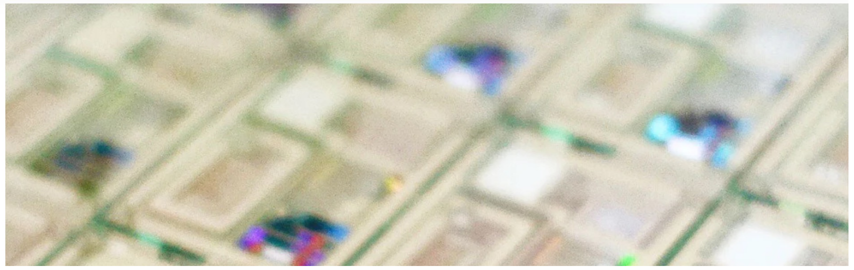
Security “level” (bits) *	RSA Key Length (bits)
80	1,024
112	2,048
128	3,072
140	4,096
192	7,680

* An *n*-bit security level implies that an attacker would need to perform 2^n operations to “solve” it.

And quantum computing is an “issue”

MIT Technology Review

Featured Topics Newsletters Events Podcasts



(using a mere 20 million qubit quantum computer!)

Computing

How a quantum computer could break 2048-bit RSA encryption in 8 hours

A new study shows that quantum technology will catch up with today’s encryption standards much sooner than expected. That should worry anybody who needs to store data securely for 25 years or so.

by **Emerging Technology from the arXiv** May 30, 2019

Many people worry that quantum computers will be able to crack certain codes used to send secure messages. The codes in question encrypt data using “trapdoor” mathematical functions that work easily in one direction but not in the other. That makes encrypting data easy but decoding it hugely difficult without the help of a special key.

These encryption systems have never been unbreakable. Instead, their security is based on the huge amount of time it would take for a classical computer to do the job. Modern encryption methods are specifically designed so that decoding them would take so long they are practically unbreakable.

What if…

You’re concerned by this because:

- You want to protect the data for longer than 8 hours
- You believe that quantum computers will develop quickly
- You believe that ECDSA represents a lower level of resistance to quantum computing techniques

- So you might want to keep using RSA, but extend its key length to defend against this risk
- To 4,096 bit keys perhaps?

Will RSA-4096 work for DNSSEC?

The major concern is the size of the data to be carried in DNS payloads

Algorithm	Private Key (bytes)	Public Key (bytes)	Signature (bytes)	Security Level
RSA-1024	1,102	438	259	80
RSA-2048	1,776	620	403	112
RSA-4096	**3,312**	**967**	**744**	**140**
ECDSA P-256	187	353	146	128
Ed25519	179	300	146	128

RSA-4096 Performance

Algorithm	Signing Time (secs)	Validation Time (secs)
None		905
RSA-1024	52	1,168
RSA-2048	126	1,173
RSA-4096	**830**	**1,176**
ECDSA P-256	159	1,036
Ed25519	205	1,008

Signing Time is the elapsed time to size a zone with 0.5M entries
Validation Time is the elapsed time to validate 50K responses

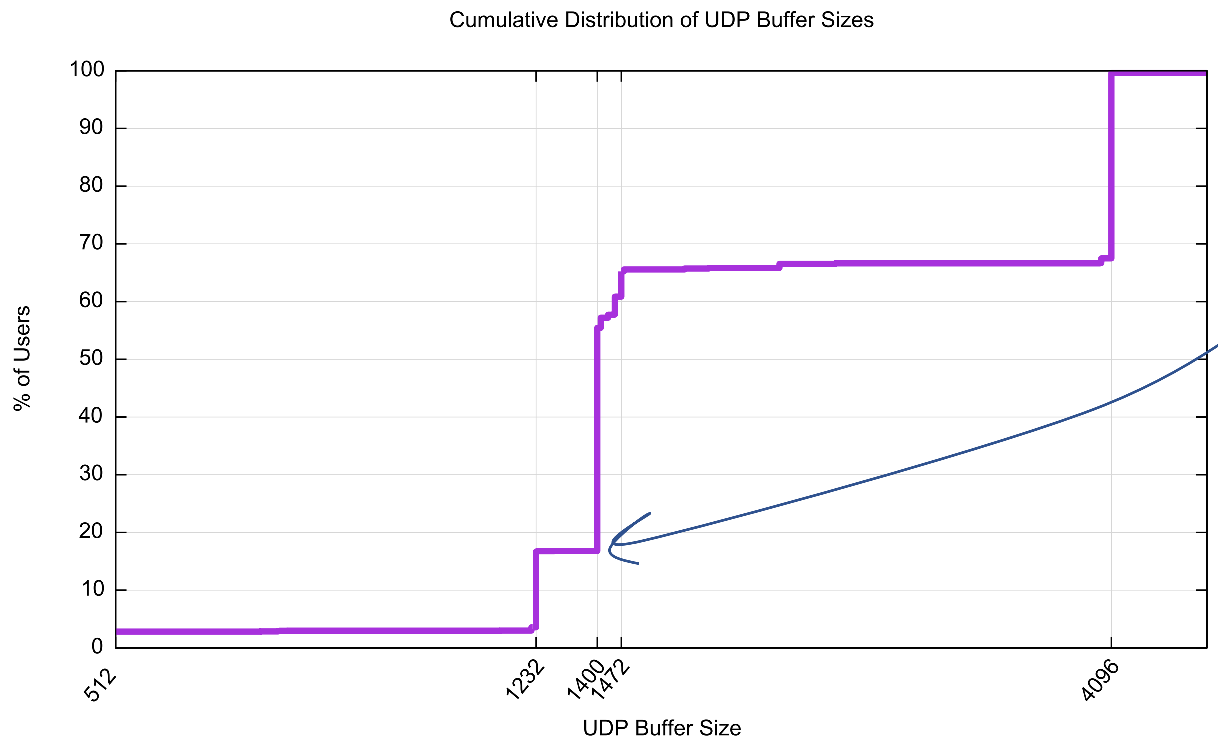
RSA-4096 signed DNS Response sizes by Query Type

RR Type	RSA-4096 (bytes)	ECDSA P-256 (bytes)
A	747	273
DS	721	245
DNSKEY	1,245	347

DNS Flag Day 2020 proposed a maximum DNS payload of 1,232 bytes when using UDP transport

Will it fit?

If we are looking at the point of UDP truncation then we need to look at the distribution of EDNS(0) UDP Buffer size settings on these DNSSEC validation queries



17÷ of users query via recursive resolvers that use a UDP buffer size of less than 1,245 bytes

Test Rig

- Configure an Ad campaign to ask for 2 URLs:
  - One has a domain name that is signed using an RSA-4096 key
  - One uses an invalidly signed domain name, again using a RSA-4096 key
- The Domain names are dynamically generated with unique sub-labels

https://0ds-udeb9087f-c13-a1283-s1632968189-i00000000.ape.dotnxdomain.net/1x1.png
https://0di-udeb9087f-c13-a1283-s1632968189-i00000000.ape.dotnxdomain.net/1x1.png

- We perform a DNS packet capture at the authoritative server
- We are looking for experiments that use resolvers that ask for DS and DNSKEY records

Experiment result classes

1. **“Validating”**

   The A / AAAA queries have the DO bit set

   DS / DNSKEY queries are made

   The user fetches the validly signed web object, but **not** the invalidly signed object

2. **“Mixed Validating”**

   The A / AAAA queries have the DO bit set

   DS / DNSKEY queries are made, but not necessarily by every resolver

   The user fetches both the validly-signed and invalidly signed web objects

Single RSA-4096 Key

- The zone uses a single yet as both the KSK and the ZSK
- The DNSKEY records contains a single key

Algorithm	Validating (% of tests)	Mixed (% of tests)
RSA-1024 *	29.7%	9.0%
RSA-4096	29.4%	9.1%

* We use a RSA-1024 signed zone as a control, where the DNSKEY response is 491 octets

DNSKEY query processing

- 74% of experiments received the DNSKEY response of 1,245 bytes over UDP
- 26% of experiments had a smaller UDP buffer size and were sent a truncated UDP response
  - 23,5% of experiments followed up using TCP
  - And 2.5% did not!
    - 2% then re-queried with a different resolver that used a larger UDP buffer size
    - 0.5% failed DNS resolution

But maybe this is not a “real” test

- Perhaps a more realistic scenario for a DNSSEC-signed zone is to use a ZSK and a separate KSK, which means we should look at the DNSKEY record with 2 x RSA-4096 keys
- And we should also consider a key roll scenario, which means we should also look at the DNSKEY record with 3 x RSA-4096 keys
- So let’s go there!

Multiple RSA-4096 Keys

Key Count	DNSKEY Response Size (bytes)
1 x RSA-4096	1,245
2 x RSA-4096	1,755
3 x RSA-4096	2,237

Validation Outcomes with larger DNSKEY records

Algorithm / Key Count	Validating	Mixed
RSA-1024	29.7%	9.0%
RSA-4096 x 1	29.4%	9.1%
RSA-4096 x 2	27.9%	9.1%
RSA-4096 x 3	24.0%	7.8%

The comparison between the 2-key and 3-key cases shows that there are more issues than just UDP fragmentation and/or TCP re-query when the DNS response grows from 1,755 to 2,237 bytes

Are we touching upon internal implementation issues? Or perhaps selective network responses to pre-empt potential DNS amplification attacks? Perhaps other causes are at work here to create this difference between the outcomes of these two cases.

What’s going on?

- Maybe the resolver’s UDP Buffer size in queries is being too optimistic and doesn’t reflect the resolver’s ability and the local network’s ability to admit fragmented UDP responses and reassemble the responses
  - Path MTU mismatch, security policies, receiver buffer limits?

Or

- The resolver is unable to perform a TCP fetch after receiving a truncated UDP response
  - Over-enthusiastic local security rules, or borked DNS implementations?

How often is failure to complete a TCP re-query a problem here?

Key Count	TCP Failure Rate
RSA-4096 x 1 (1,245 bytes)	0.1%
RSA-4096 x 2 (1,755 bytes)	2.5%
RSA-4096 x 3 (2,237 bytes)	7.2%

Here we are looking for a query sequence where we respond to the original query with a truncated UDP DNS response, but there is no subsequent TCP completed re-query

Where is this a problem?

	RSA-1024	RSA-4096x2	Difference
Portugal	68%	40%	-28%
Morocco	59%	31%	-27%
Iceland	95%	72%	-23%
Guyana	41%	28%	-13%
USA	60%	48%	-12%
Ireland	27%	18%	-9%
Switzerland	81%	73%	-9%
Brunei	31%	23%	-8%
Singapore	71%	64%	-8%
Sweden	91%	84%	-7%

These are all % of users who complete DNSSEC Validation

Which ISPs?

	RSA-1024	**RSA-4096 x 2**	**Difference**	**AS Name**
AS39603	93%	47%	-45%	P4 UMTS, Poland
AS5466	94%	51%	-44%	Eircom, Ireland
AS23688	93%	54%	-39%	Link3, Bangladesh
AS45543	73%	34%	-39%	SCTV, Vietnam
AS36903	77%	41%	-36%	MT-MPLS, Morocco
AS34779	91%	56%	-35%	T-2, Slovenia
AS35819	93%	65%	-28%	Etihad Etisalat, Saudi Arabia
AS28573	63%	37%	-26%	Claro, Brazil
AS3243	92%	67%	-25%	Meo Residencial, Portugal
AS4818	91%	69%	-22%	Digi Telecom, Malaysia

Does RSA-4096 have a future in DNSSEC?

- It’s not looking good!

- But does it matter?
- Should we be turning to using 140-bit security level in DNSSEC in 2021 in any case?
  (i.e. is RSA-4096 over-achieving for DNSSEC?)

It’s DNSSEC!

- What is the “secure lifetime” of a signed item of DNS data?
  - It’s not hiding a “secret”!
  - It‘s protecting the integrity of the DNS data
  - And the integrity of the digital signature depends on the key lifetime
  - So the anticipated secure lifetime of a DNSSEC key needs to be greater than the key lifetime, but not that much longer!
- So the longer the lifetime of a DNSSEC key, then the greater your requirement for a longer secure lifetime, which implies the greater your need a higher security level of your algorithm and key
  - For example, if your roll keys every 6 months then your secure lifetime requirement is less than 1 year
- While RSA-1024 is probably incapable of providing a 10 year secure lifetime for encrypted messages, it is likely to still be useful in DNSSEC as long as you roll your keys more frequently than every decade!

It’s Quantum Computing!

- In our current understanding of the quantum computing environment the concept of “security level” does not transcribe from conventional to quantum computing cleanly
- While some algorithms and key profiles have the same “security level” they have different levels of *quantum resilience*
- For example, RSA with longer keys lengths is thought to be more *quantum resilient* than an equivalent security level elliptical curve profile

Questions?