Field Experiments on Post-Quantum DNSSEC

Carlos Aguilar Melchor[1], Jason Goertzen[2], Aydın Mercan[3], Shumon Huque[4], Peter Thomassen[5], Nils Wisiol[5]

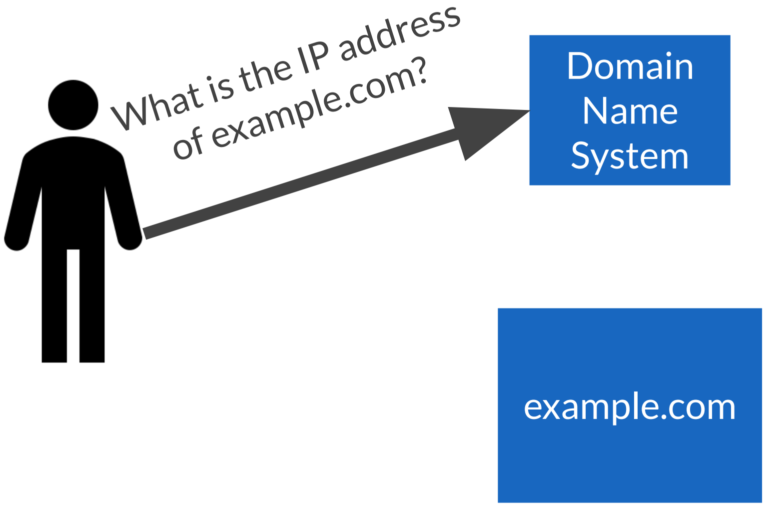
1 SandboxAQ, 2 Epsilon Cyber, 3 Internet Systems Consortium, 4 Salesforce, 5 deSEC

WHAT IS DNS?

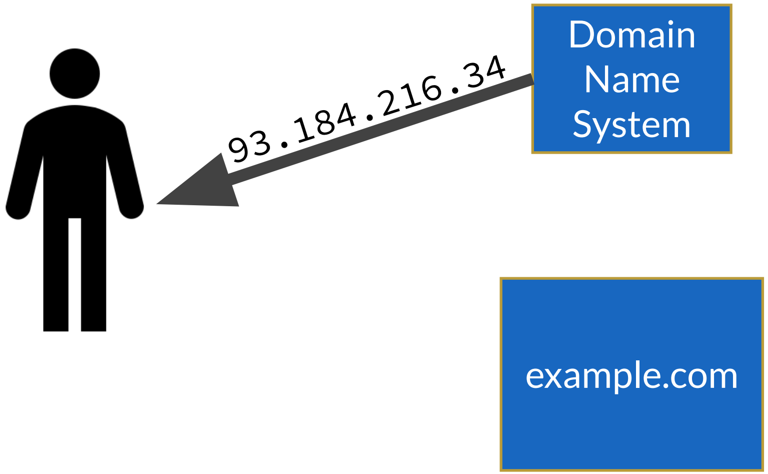
- The internet uses IP addresses to determine where to send messages
- IP addresses are difficult for people to remember!
- The Domain Name System is responsible to translating something easy for a human to remember into IP addresses

```
example.com -> 93.184.216.34
```

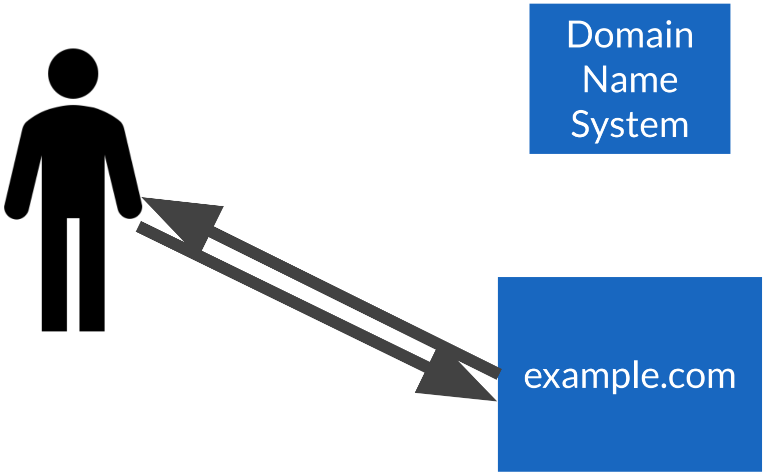
WHAT IS DNS?


What is the IP address
of example.com?
Domain
Name
System
example.com

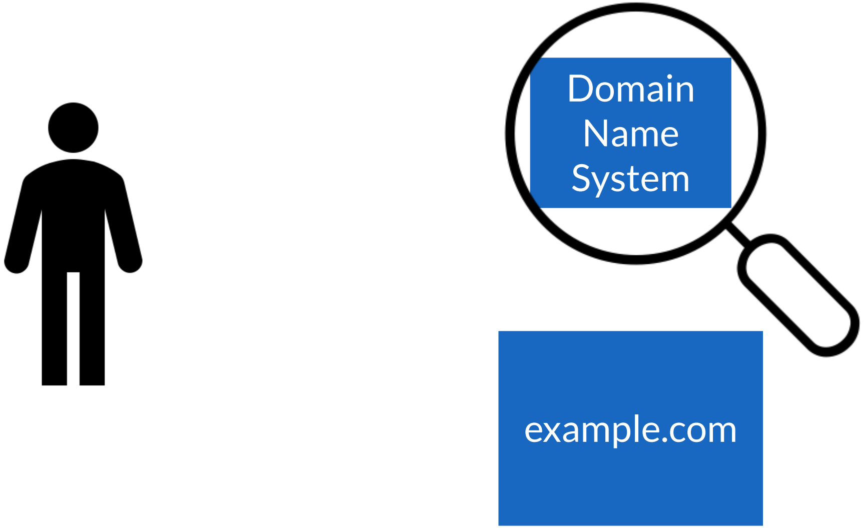
WHAT IS DNS?



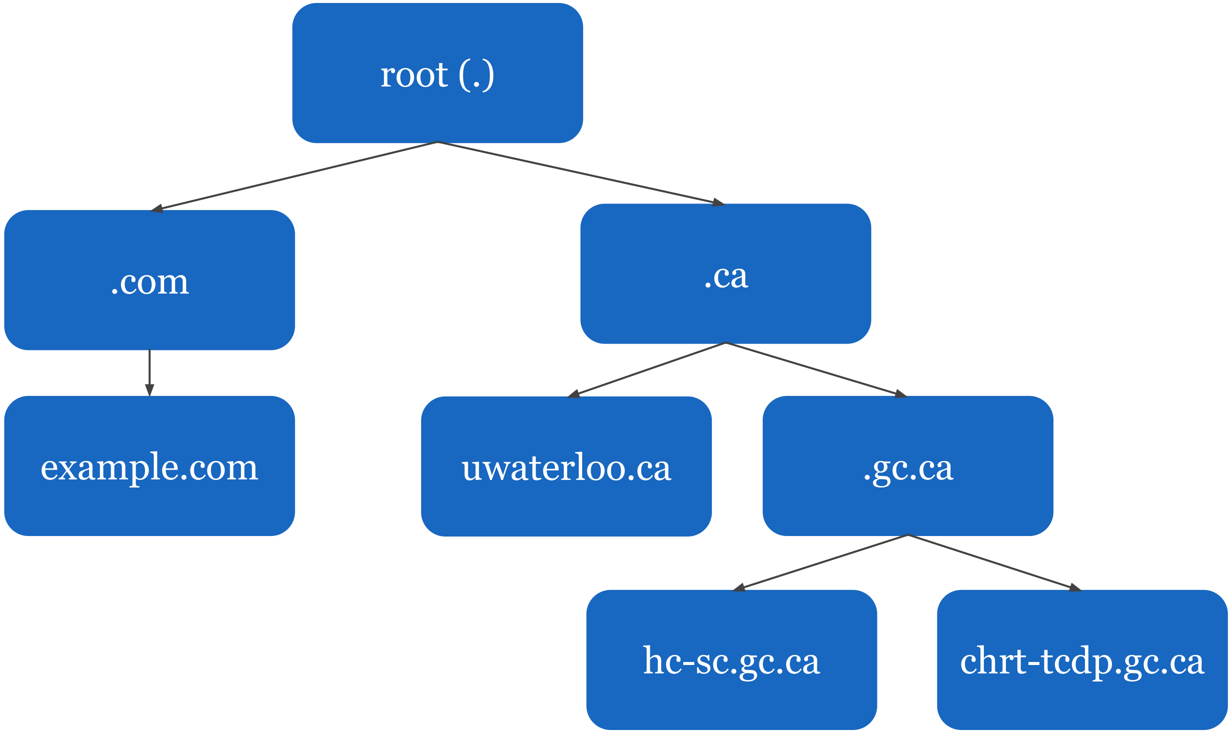
WHAT IS DNS?



WHAT IS DNS?



DNS IS BROKEN UP INTO ZONES



Root level

Top level domains

Second level domains

Third level domains

Quick Reminder: DNS Zones

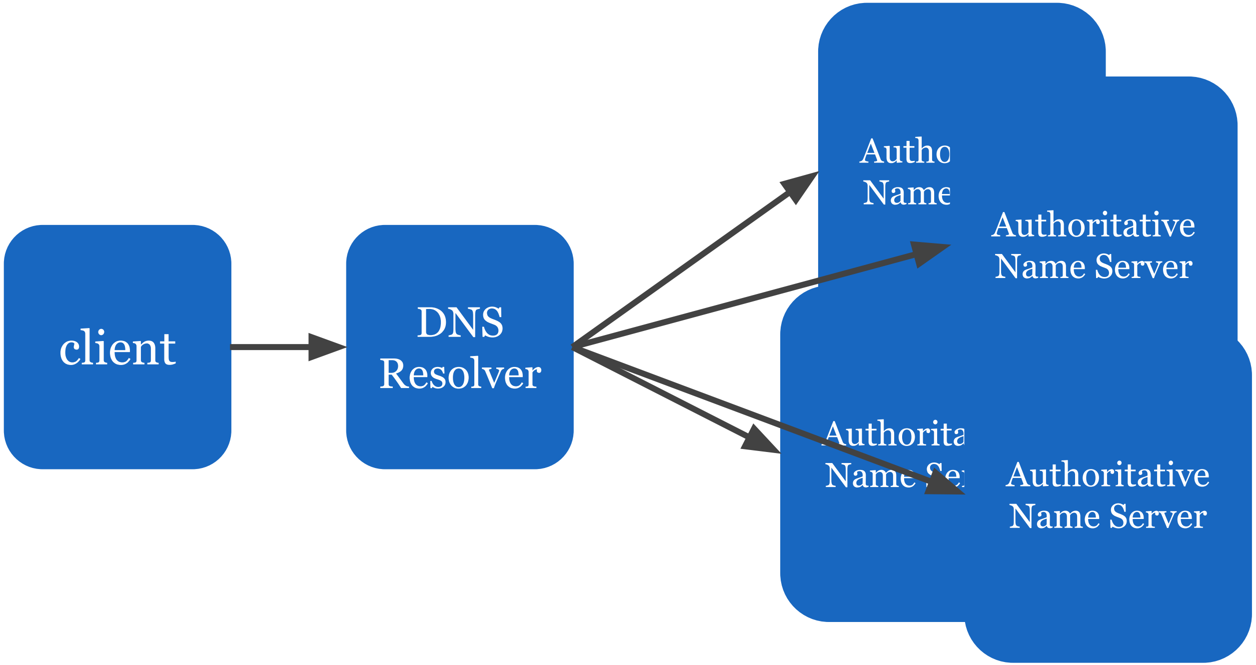
- Look-up table for a domain name, including subdomains

```
; name                     TTL       type  content
example.com.             86400   IN  SOA   ...
example.com.              3600   IN  A     192.0.2.1
example.com.              3600   IN  AAAA  2001:db8:10::1
example.com.             86400   IN  MX    10 mail.example.com.
                                           20 backup.example.com.
mail.example.com.        86400   IN  A     192.0.2.3
...
```

- Most lookups done “via UDP” (easy to manipulate), using port 53
- One type of data can be: “nothing to see here, go look there” → **delegation**

```
intranet.example.com.    86400   IN  NS    other.provider.net.
```

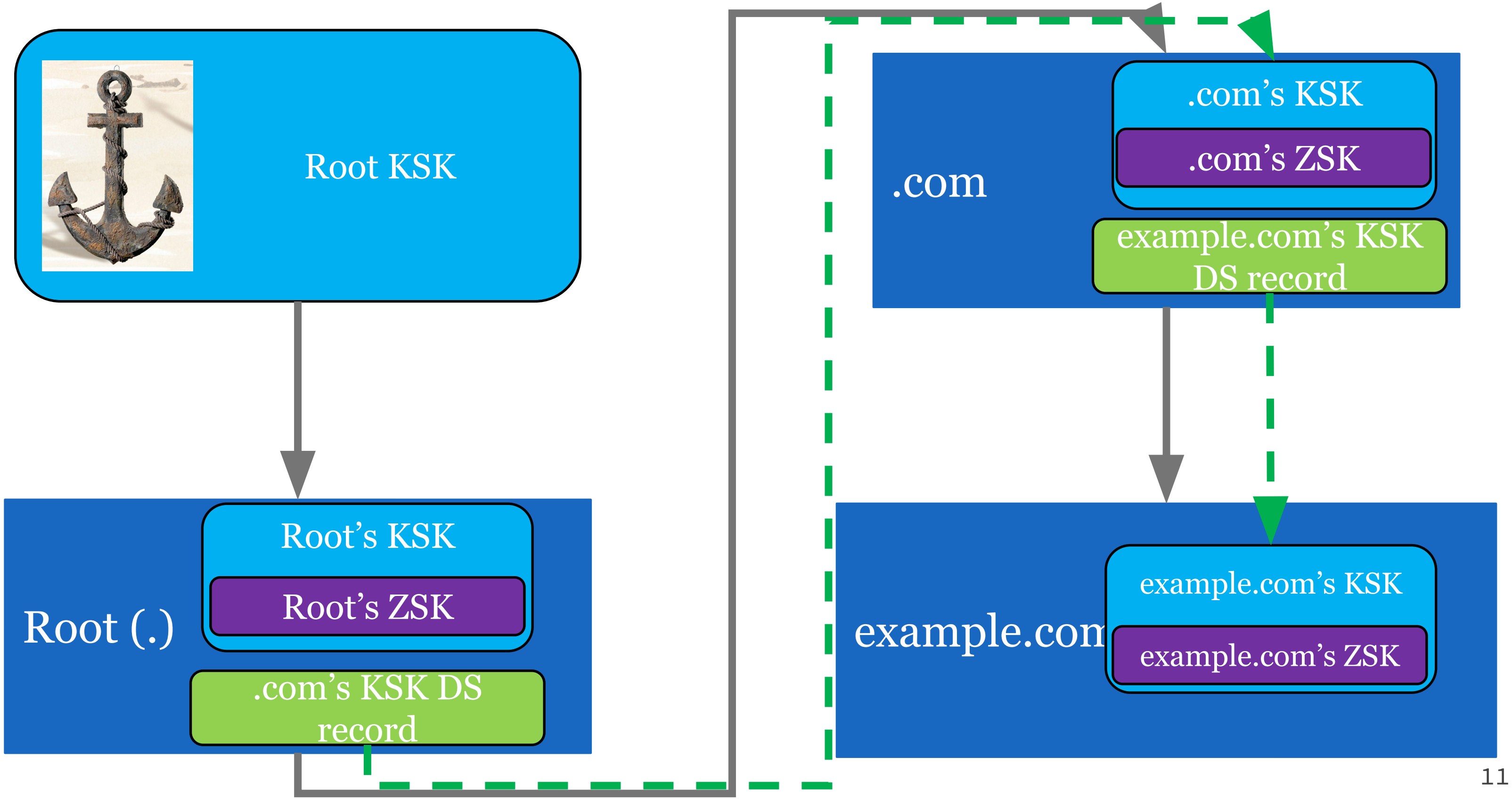
CLIENTS RARELY QUERY DIRECTLY



HOW DO WE MAINTAIN KEY INTEGRITY?

- Construct a chain of trust!
- Typical scheme: two keys with different functions
  - Zone-signing key (ZSK): signs DNS records in a zone
  - Key-signing key (KSK): signs ZSK and is linked in the parent zone via DS record ("Delegation Signer")
- The root verification KSK acts as a trust anchor
  - must be pre-configured on validating machine
- When the root ZSK is queried use the trust anchor to verify key and its signature

HOW DO WE MAINTAIN KEY INTEGRITY?



The peril of large DNS messages

DNS messages must be contained in a single UDP packet

- UDP fragmentation is fragile – long thought to cause deliverability issue
- DNS messages must be no larger than 1232 bytes

Algorithm	Public Key Size	Signature Size
RSA 2048	256	256
ECDSA P256	64	64
Falcon512 (FN-DSA)	897	666
Dilithium2 (ML-DSA)	1,312	2,420
SPHINCS+-128s (SLH-DSA)	32	7,856

Gold standard DNS: BIND 9 POWERDNS

Quantum safe Algorithms:

OPEN QUANTUM SAFE

software for the transition to quantum-resistant cryptography

FALCON

Fast-Fourier Lattice-based Compact Signatures over NTRU

DILITHIUM

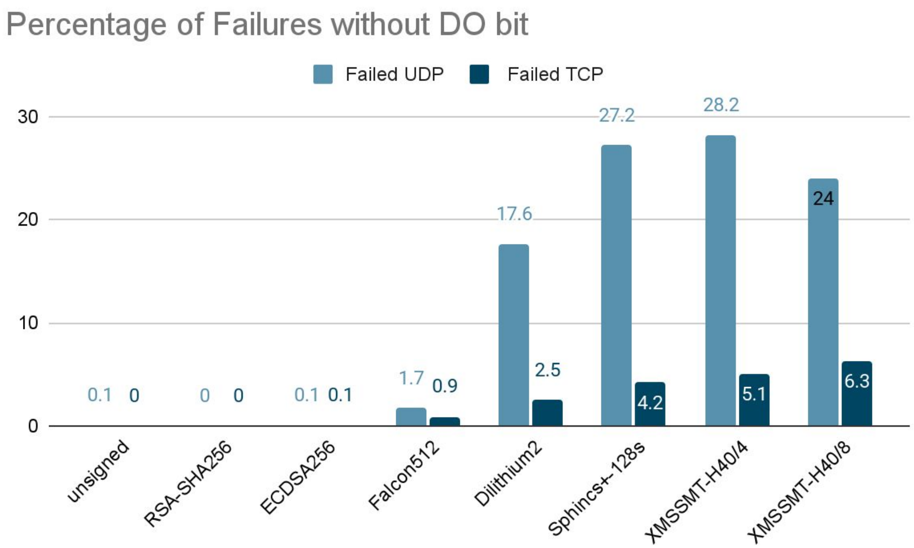
10,000 probes, 2,000,000 queries RIPE Atlas

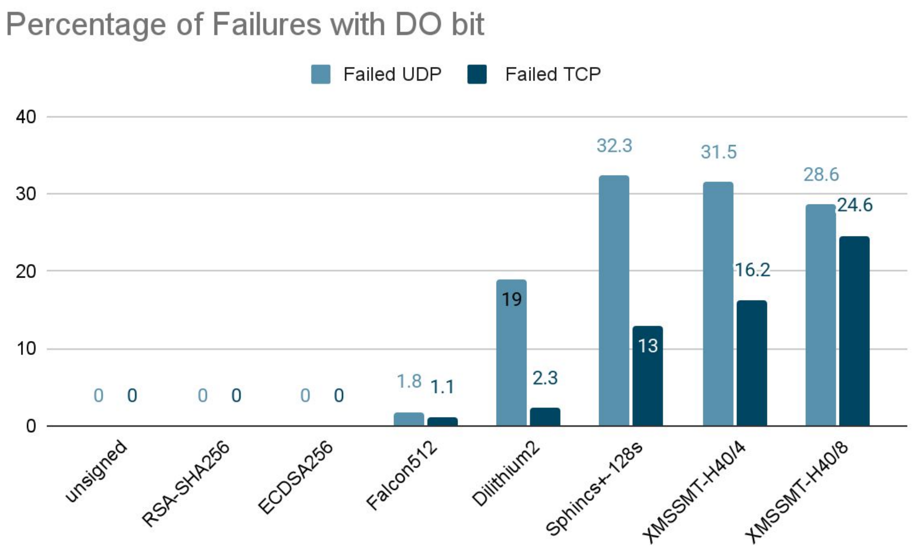
Queries Using a PQC-aware Resolver

dig +dnssec A dilithium2.pdns.pq-dnssec.dedyn.io @bind9.pq-dnssec.dedyn.io *-p 5304*

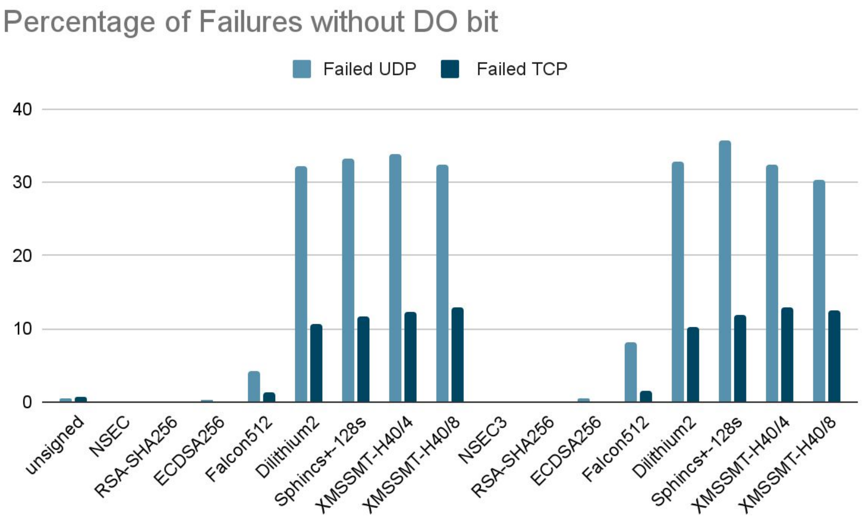
```
;; Truncated, retrying in TCP mode.

; <<>> DiG 9.18.24-0ubuntu0.22.04.1-Ubuntu <<>> +dnssec A dilithium2.pdns.pq-dnssec.dedyn.io @bind9.pq-dnssec.dedyn.io -p 5304
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 22245
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 1232
; COOKIE: 8455829f86d7fb7601000000669b5d9517dfc67dff539cac (good)
;; QUESTION SECTION:
;dilithium2.pdns.pq-dnssec.dedyn.io. IN A

;; ANSWER SECTION:
dilithium2.pdns.pq-dnssec.dedyn.io. 3599 IN A  95.217.209.184
dilithium2.pdns.pq-dnssec.dedyn.io. 3599 IN RRSIG A 18 5 3600 20240801000000 20240711000000 3978 dilithium2.pdns.pq-dnssec.ded
yn.io. 19/28JXGCgGbNtEAtUOzv1/SzP+kr6vBlglWrJ/ZfYgdC1DXZHdh+xol rnZ9uhvmADCqZzJXOyOU1Tyw2sHN32Vmcv4KLR8lI7TBwfTJq6T3nGfV oQnv9
DNvPJTyb4VonYH3fLTMYeQ3/0Wy9gbv0ngy55QqRjw+ikhS0yIp ezpZYH3ArY/xxmTgM7OBW0yBg3gXgo1G2mrX97ufqrwkO/n0Vu/xXfSI npGKq+dVu7LQQR7nM
lmM3FkbaRAFyo0FjmbzXDPtyrwJekJP8dfQ5zvc pOCRfrpjRg+ZBUofhdk1PURO539JwD[...shortened...]AA AAAAAAAAAAAAAAAAAAAAAAAAAAAAABAcJzc=

;; Query time: 56 msec
;; SERVER: 35.232.14.170#5304(bind9.pq-dnssec.dedyn.io) (TCP)
;; WHEN: Fri Jul 19 23:47:49 PDT 2024
;; MSG SIZE  rcvd: 2593
```

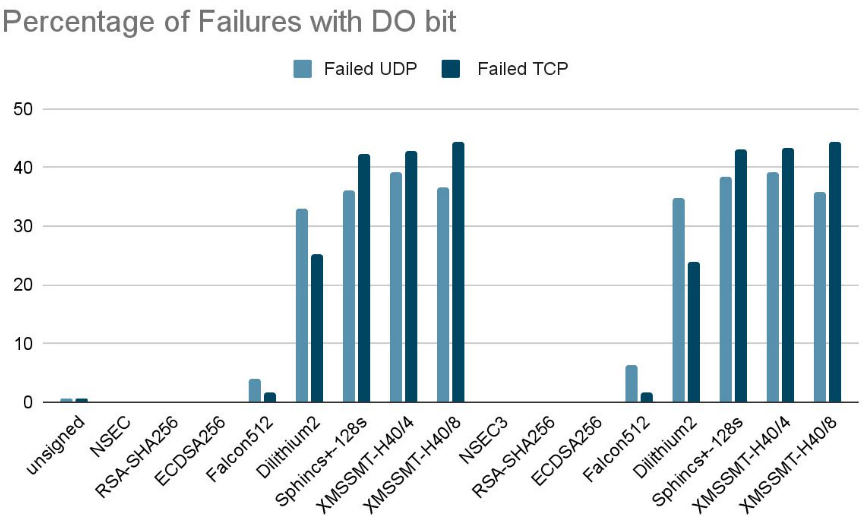
Failures for a valid label





Failures for a nonexistent label





CONTACT

Post-Quantum DNSSEC Testbed with BIND and PowerDNS

Query our PQC-enabled DNS Resolvers

Send queries to our post-quantum enabled validating resolver! You can choose from a number of post-quantum (and classical) signing schemes, NSEC or NSEC3 mode, and implementations for PowerDNS (source) and BIND (source).

Zones signed accordingly are available at `{algorithm}.{vendor}.pq-dnssec.dedyn.io`, and each has a `A` and a `TXT` record configured. To query a non-existing name, prepend the `nx` label (for example).

Queries will be sent from your browser using DNS-over-HTTPS to a BIND or PowerDNS resolvers with validation support for the selected algorithm. The resolver will talk to the corresponding BIND or PowerDNS authoritative DNS server (again, with support for the selecting signing scheme), to get your response. It will then validate the signature and send the result to your browser.

All queries are send with the `DNSSEC_OK` flag (`+dnssec` in dig), so you will see `RRSIG` and `NSEC`/`NSEC3` records the the responses.

Query type	Algorithm	Authoritative vendor	Resolver vendor
	Dilithium2	PowerDNS	PowerDNS

☐ NSEC3 ☑ non-existent name

Domain name: nx.dilithium2.pdns.pq-dnssec.dedyn.io

QUERY DNSViz

Credits: Jason Goertzen (Sandbox AQ), Nils Wisiol. Funded by NLnet Foundation.

Try it yourself!

https://pq-dnssec.dedyn.io/

(also has detailed results)

What we observed

- **Transmission issues are real**
  - PQC response delivery rates go down significantly as response sizes increase → Falcon leads
  - Gets worse depending on circumstances, like with DO bit or with NSEC3
- **UDP & DO=0:**
  ~70% KSK/ZSK responses correct
  ~80% CSK responses correct
  - Goes up by ~10% via TCP
- **UDP & DO=1:**
  ~50% responses correct
  - Goes up by ~20–40% via TCP

Investigations into Falcon AD bit

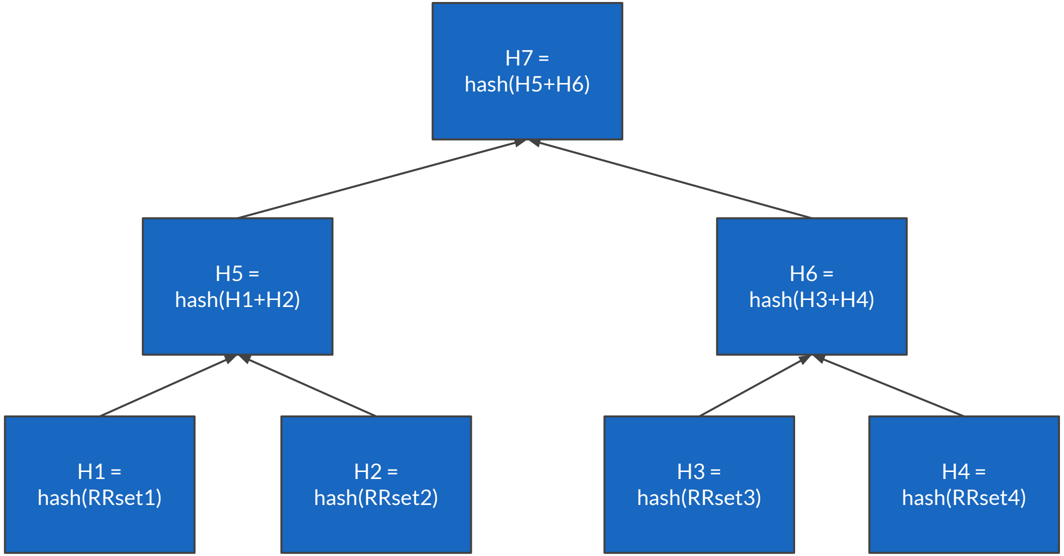
- In initial tests, 8.5% of probe-resolver pairs claim successfully validating Falcon
  - Implausible: resolvers in the wild are not expected to validate (our flavor of) Falcon signatures
- Re-measured select RIPE ATLAS probes showing this behavior
  - Selection: probes whose operators likely can be contacted for debugging (universities)
- Analysis of re-measurement did not reproduce AD bit behavior
  - Except for one university, but they did not respond to our reachout
    → root cause remains unknown
    → shows systematic error of RIPE ATLAS
- Note: Sporadically, AD bits were also observed for some probes which were configured to use Google's or Cloudflare's public resolvers (8.8.8.8 / 1.1.1.1)
  → Indicates some degree of network spoofing

Want to use PQC while keeping messages small.

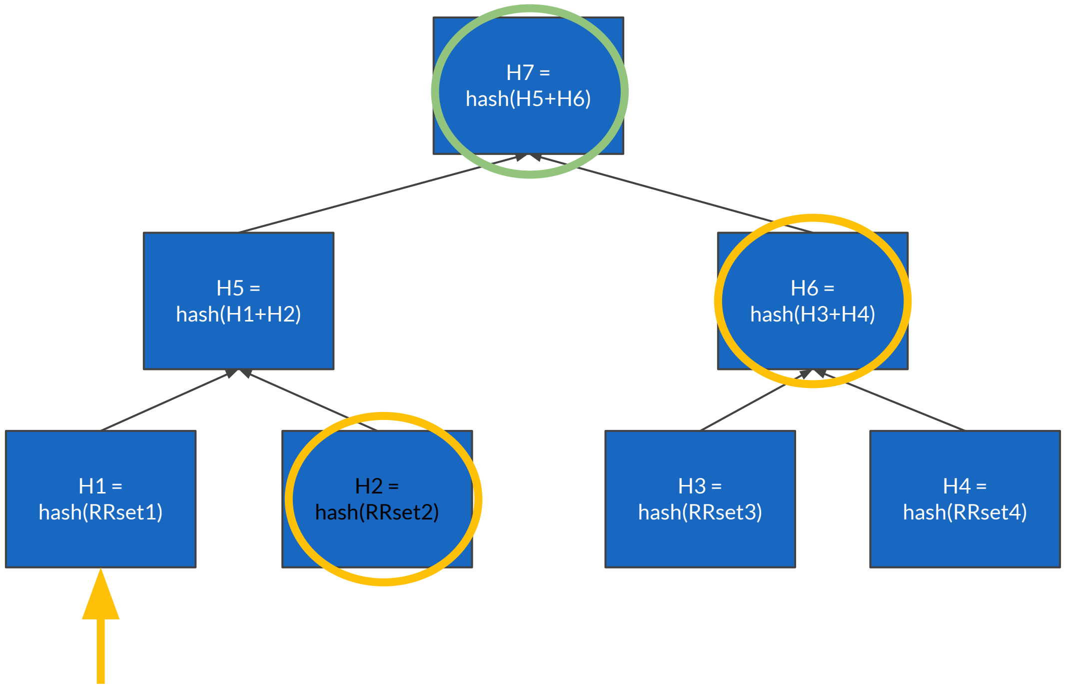
Want to use PQC while keeping messages small.

We can use Merkle Trees to make DNSSEC messages smaller!

What is a Merkle Tree?



What is a Merkle Tree?



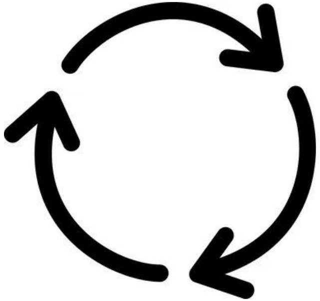
Can we apply this to DNS?

Sure!

- Use a Standardized DNSSEC Algorithm for our KSK
  - Provides Authenticity and Integrity
- Define a new "Merkle Tree" algorithm
  - Store the root hash in the ZSK's record
  - Provides Integrity via proof of inclusion + gets Authenticity from being signed by KSK
- Record "signatures" simply contain authenticating path of the Merkle tree
  - Grow logarithmically with the number of record sets in a zone
- We can combine the work from Batched Signatures Revisited [1] to reduce hash size without reducing security (Second Preimage Resistance)

[1] https://eprint.iacr.org/2023/492

We need to change some things about DNS first...



Circular signing is an issue

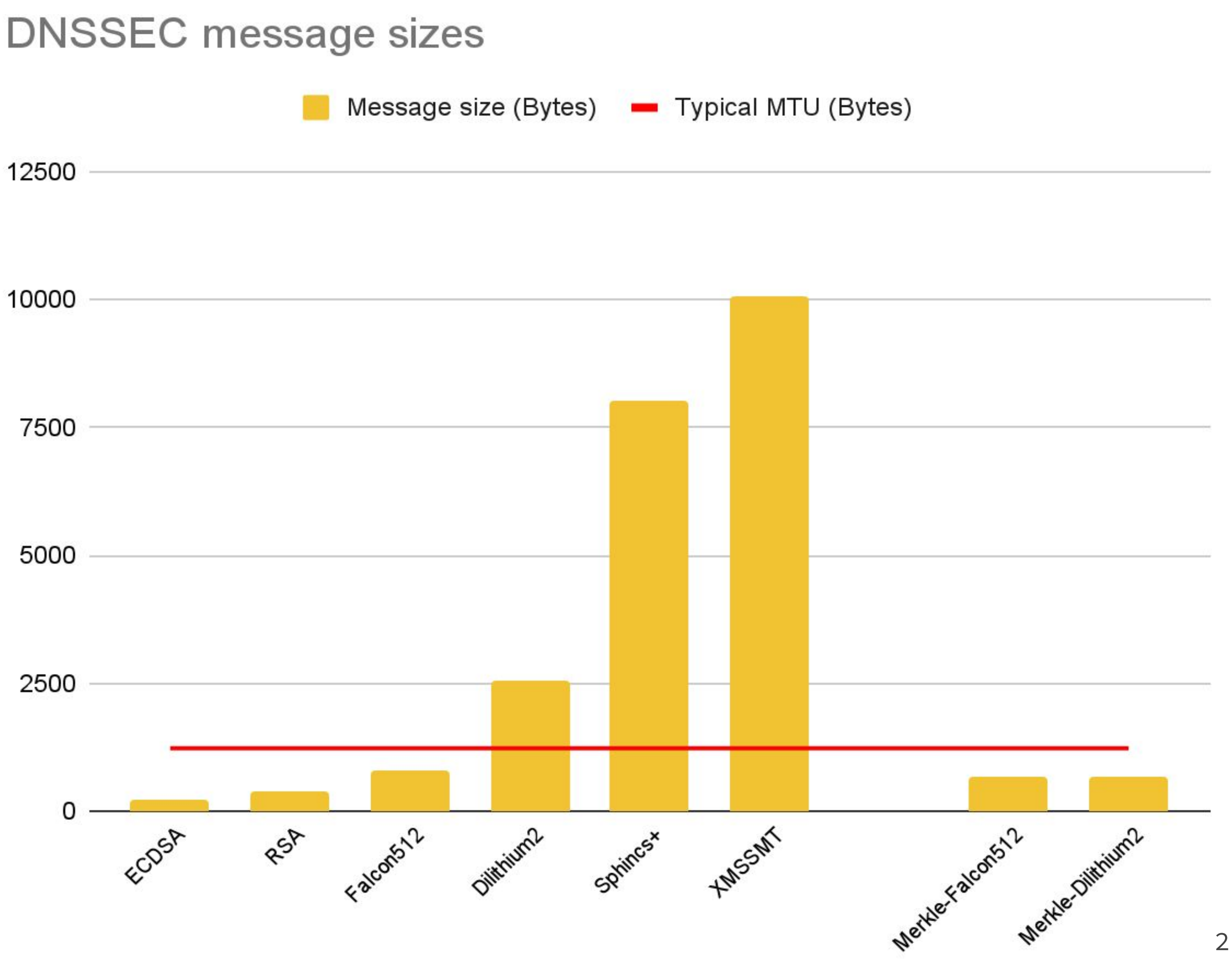
- ZSK changes each time we sign something
- Everytime the root node changes:
    - the keytag changes -> sign/verify input changes

Sign/Verify needs to change for Merkle Trees

- Exclude keytag as part of the data being signed

We get two nice wins

DNS messages without DNSKEY set stay below line of peril!



Merkle trees make zone files smaller and faster

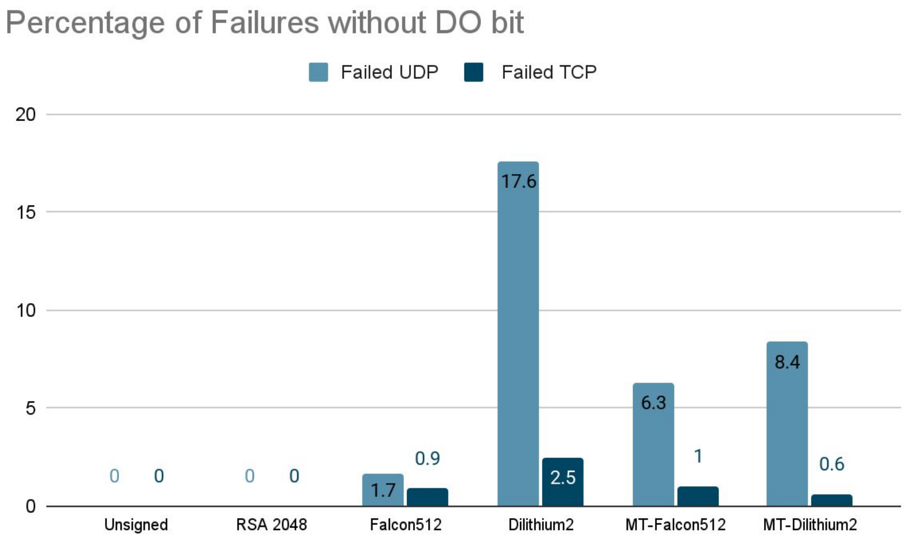
- The signatures logarithmically grow based on the number of signatures
  - In our largest real-world zone (6.2M signatures) we save 7GB by using merkle trees compared to using Falcon on its own (23GB when comparing to Dilithium)
- In general signing times are less than all other signing algorithms except for ECDSA
  - Fully implemented: slightly slower than ECDSA
  - Partially implemented: twice faster (and seems vastly improvable)

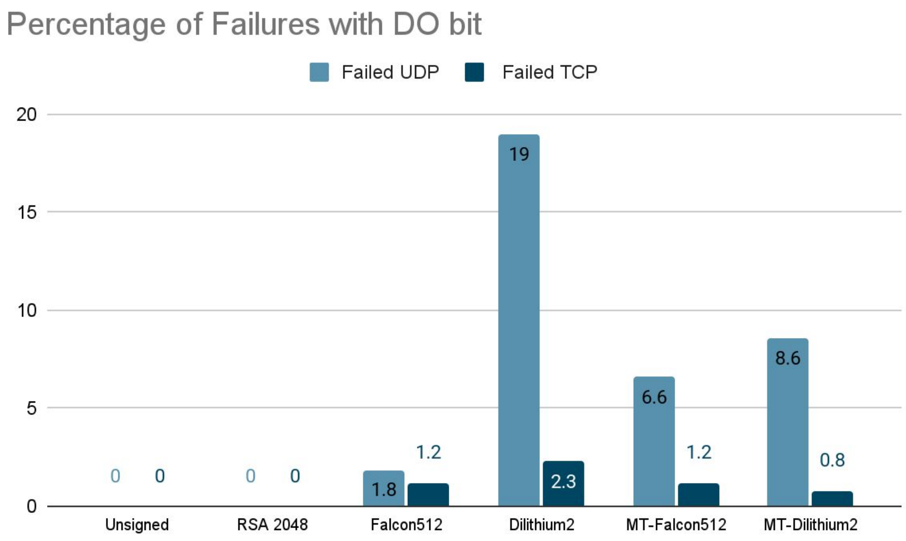
Tiny zone transfers

- DNS infrastructures usually usehave a central primary, and many secondary servers answering clients' queries (kept in sync via zone transfers)
- Since a private key isn't involved, all secondary servers can rebuild the tree and authenticating paths
- Interesting trade-off: We can transmit empty signatures during zone transfers greatly reducing the size of the zone
    - Only one signature in zone transfer (for DNSKEY RRset)
- (We don't have a full implementation for this at the moment)

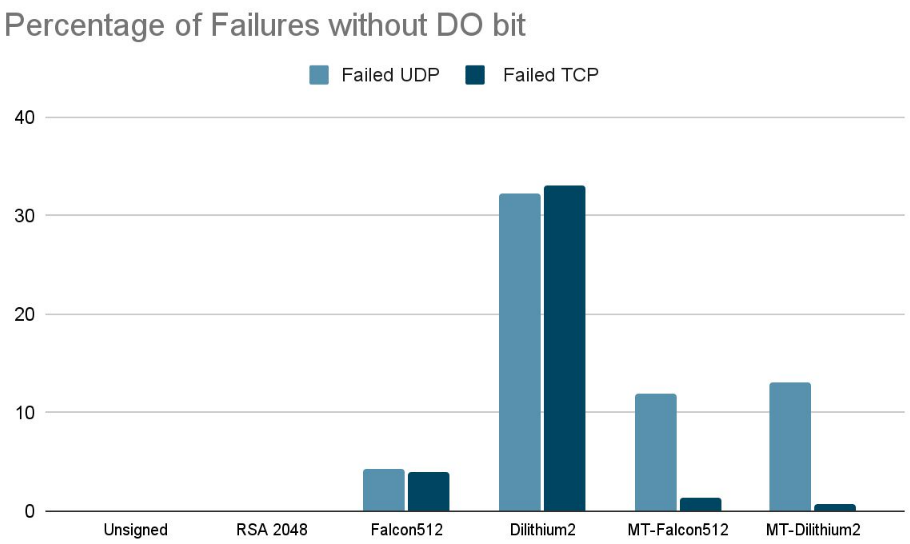
Was there a difference in ATLAS tests?

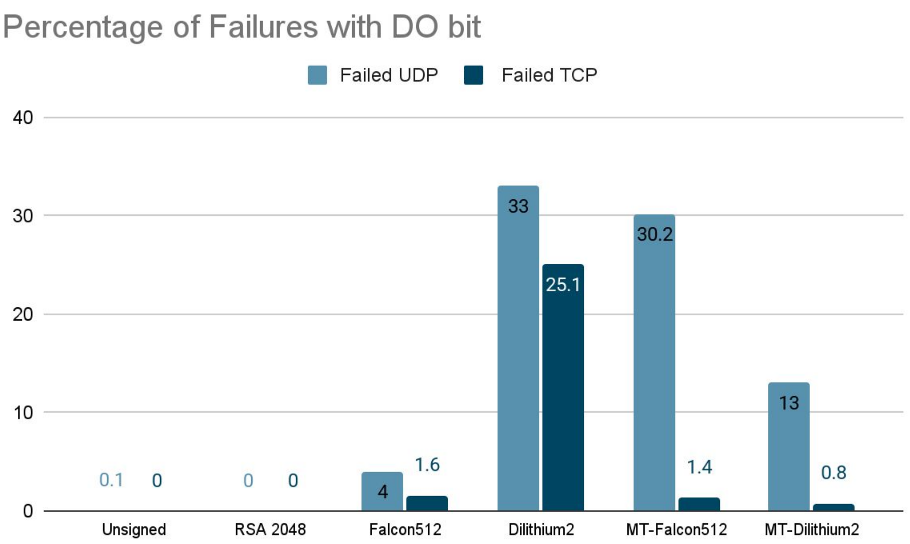
Failures for a valid label





Failures for a nonexistent label





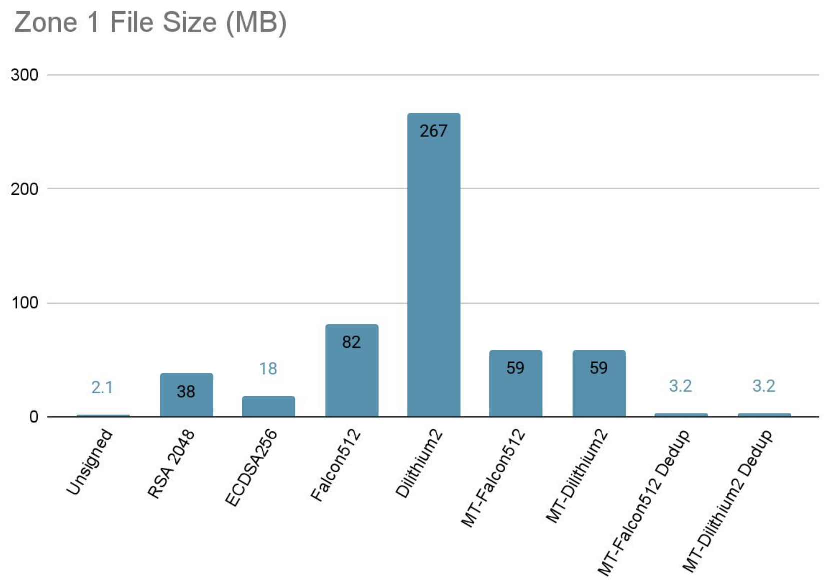
DNS – A Real World Example of a large enterprise

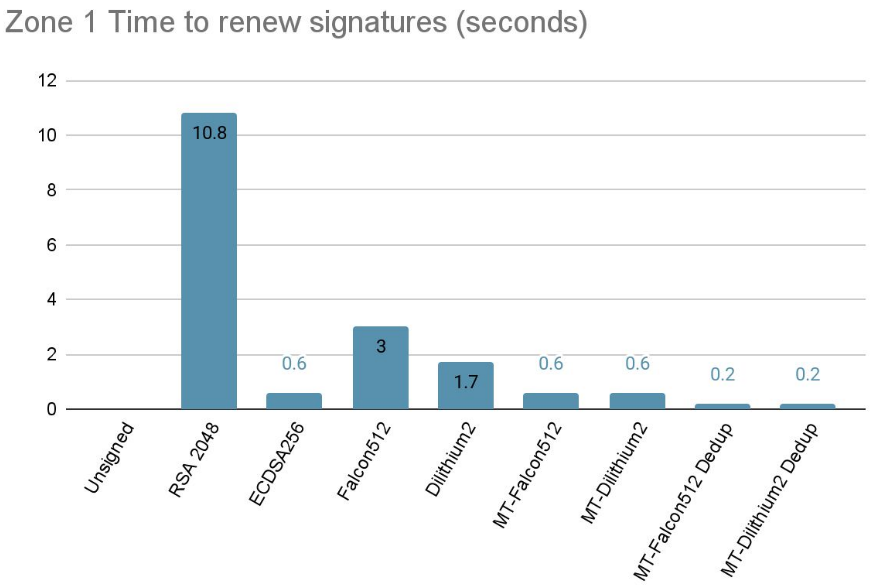
- ~5000 zones in total
  - ~1000 in active use; many of the rest are defensively registered and parked
- Size: very small (a few records) to ~20 zones with millions of records each
  - Wide variety of DNS record types in use
- Update rates vary considerably
  - Many zones changing very seldom, but a few hundred changing rapidly
  - Most active ones support a change rate of a few hundred per minute
- Using pre-computed signatures or online signing
  - depending on the feature set needed and DNS provider involved
- Mix of inhouse DNS services and 3rd party managed DNS providers are in use

Descriptions of Zones tested

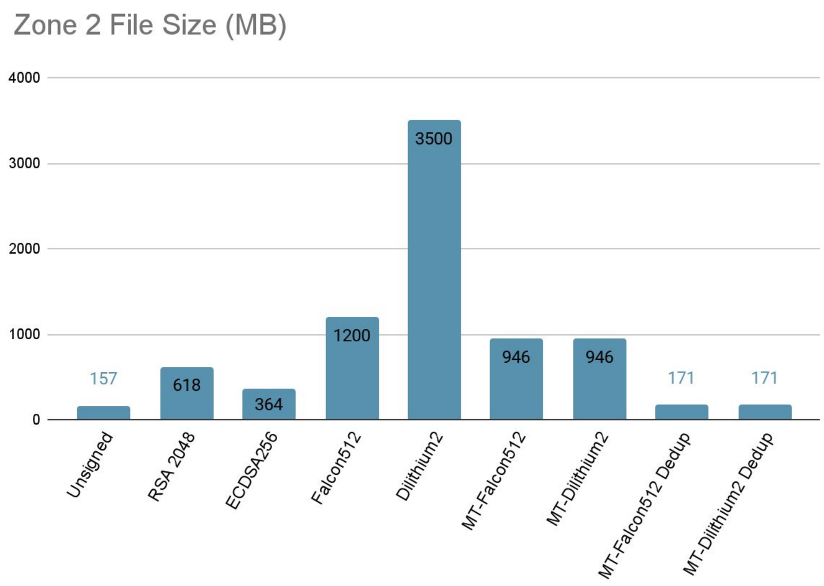
- Copies of zones with some level of data redaction
  - Zone 1 - Cloud infrastructure services
  - Zone 2 - Email infrastructure services
  - Zone 3 - Corporate website services
  - Zone 4 - Smaller zone of customer and tenant specific names (redacted)
  - Zone 5 - Realtime services
  - Zone 6 - Very large zone of customer and tenant specific endpoint names (redacted)
- Focusing in this presentation on Zones 1, 2, and 6

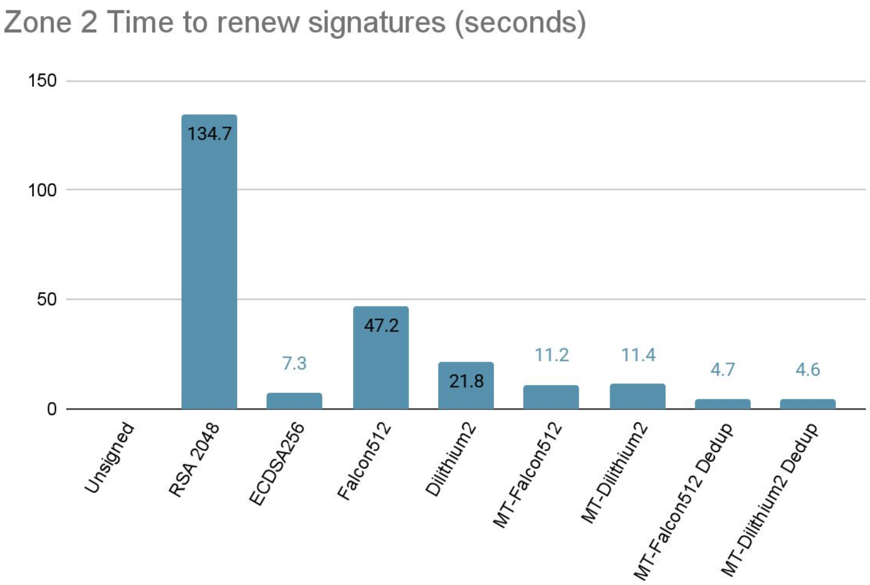
Zone 1: Cloud infrastructure services (71380 signatures)



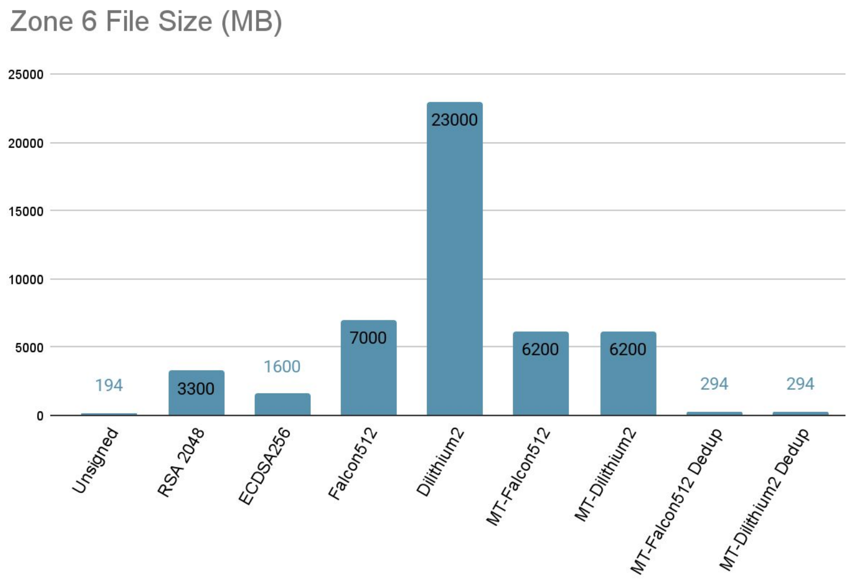


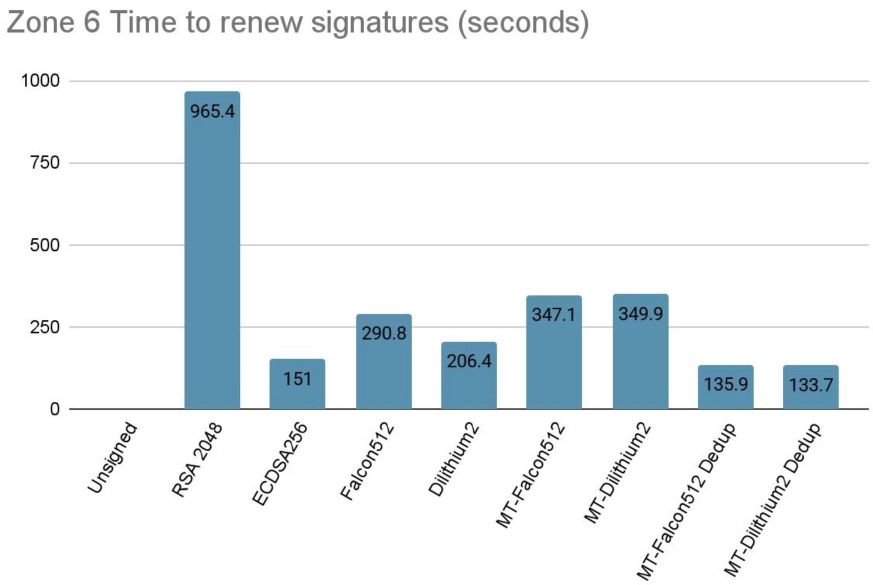
Zone 2: Email infrastructure services (897922 signatures)





Zone 6: Customer and tenant endpoints (6.2M signatures)





Observations

- Zone size explodes when using PQC
  - Falcon is over 4x
  - Dilithium is over 14x
- Attempted to sign zones with sphincs+-128s and XMSSMT_H40_4
  - Two results happened:
    - Took forever to sign the smaller zones
    - Took forever to sign and got killed by the oom killer

Zone	Algorithm	Signatures Generated	Signatures per second	Signing time in seconds	Rough zonefile size: (ls -lh)
zone1	unsigned	0		-	2.1M
zone1	RSA 2048	71380	6610.336	10.798	38M
zone1	ECDSAP256SHA256	71380	127010.676	0.562	18M
zone1	Falcon512	71380	23793.333	3	82M
zone1	Dilithium2	71380	41548.311	1.718	267M
zone1	Merkle-Falcon512	71380	58787.356	1.218	59M
zone1	Merkle-Dilithium2	71380	55036.894	1.301	59M
zone2	unsigned	0			157M
zone2	RSA 2048	897922	6667.346	134.674	618M
zone2	ECDSAP256SHA256	897922	122232.779	7.346	364M
zone2	Falcon512	897922	19016.094	43.882	1.2G
zone2	Dilithium2	897922	41164.534	21.813	3.5G
zone2	Merkle-Falcon512	897921	51298.046	17.504	946M
zone2	Merkle-Dilithium2	897921	80274.254	11.185	946M

Zone	Algorithm	Signatures Generated	Signatures per second	Signing time in seconds	Rough zonefile size: (ls -lh)
zone3	unsigned	0			3.1M
zone3	RSA 2048	65627	6635.021	9.891	36M
zone3	ECDSAP256SHA256	65627	114532.286	0.573	17M
zone3	Falcon512	65627	23977.712	2.737	76M
zone3	Dilithium2	65627	42068.589	1.56	246M
zone3	Merkle-Falcon512	65626	117609.318	0.558	52M
zone3	Merkle-Dilithium2	65626	121981.412	0.538	52M
zone4	unsigned	0			110M
zone4	RSA 2048	2627256	6601.574	397.974	1.5G
zone4	ECDSAP256SHA256	2627256	70370.823	37.334	710M
zone4	Falcon512	2627256	22753.695	115.465	3.0G
zone4	Dilithium2	2627256	36816.426	71.36	9.7G
zone4	Merkle-Falcon512	2627255	32023.629	82.041	2.6G
zone4	Merkle-Dilithium2	2627255	31453.402	83.528	2.6G

Zone	Algorithm	Signatures Generated	Signatures per second	Signing time in seconds	Rough zonefile size: (ls -lh)
zone5	unsigned	0			84M
zone5	RSA 2048	2415918	6650.713	363.256	1.3G
zone5	ECDSAP256SHA256	2415918	94668.108	25.519	618M
zone5	Falcon512	2415918	23229.767	104	2.7G
zone5	Dilithium2	2415918	35760.878	67.557	8.9G
zone5	Merkle-Falcon512	2415917	44760.77	53.973	2.4G
zone5	Merkle-Dilithium2	2415917	45353.45	53.268	2.4G
zone6	unsigned	0			194M
zone6	RSA 2048	6285416	6511.006	965.352	3.3G
zone6	ECDSAP256SHA256	6285416	41631.825	150.976	1.6G
zone6	Falcon512	6285416	21613.365	290.811	7.0G
zone6	Dilithium2	6285416	30449.105	206.423	23G
zone6	Merkle-Falcon512	6285415	18110.71	347.055	6.2G
zone6	Merkle-Dilithium2	6285415	17962.181	349.924	6.2G

dnssec-signzone

- Currently only supports offline signing
- Heavy modifications to BIND's dnssec-signzone
  - Iterate through all RRSets and add them to the Merkle tree
  - Finalize the Merkle tree and update keytag
  - Iterate over all RRSIGs and insert the correct authenticating path and keytag
  - Takes about half the time of signing the same zone with ECDSA
    - Heavily unoptimized code

Some takeaways for Merkle trees

- (Minor) DNSSEC protocol changes would need to be made
- By defining it with its own algorithm number you can use Merkle trees with any other DNSSEC algorithm
- Zone updates are limited by the root node's (ZSK) TTL
    - Verisign's MTL might help with this?
- DNSKEY messages are not compressed

What's next?

- What about ATLAS measures instability?
  - The study was motivational, there clearly seems to be a problem with PQC
  - Consequences:
    - Regular IETF side meetings on the subject
    - A number of groups are doing studies now
- Fixing may require **revamping signature representation** in DNS
  - Does not necessarily involve a wire format / spec change
  - Or will more robust DoT/DoH/DoQ gain enough traction?
- To transition, **any scalable solution will require DS provisioning automation**
- Future work needed!
  → Research agenda
  → Mailing list: pq-dnssec@ietf.org

Thank you!

Web app and results:
https://pq-dnssec.dedyn.io

Acknowledgments:

nlnet FOUNDATION   SANDBOXAQ

SSE   deSEC

Carlos Aguilar Melchor[1], Jason Goertzen[2], Aydın Mercan[3], Shumon Huque[4], Peter Thomassen[5], Nils Wisiol[5]

1 SandboxAQ, 2 Epsilon Cyber, 3 Internet Systems Consortium, 4 Salesforce, 5 deSEC

Context & Motivation

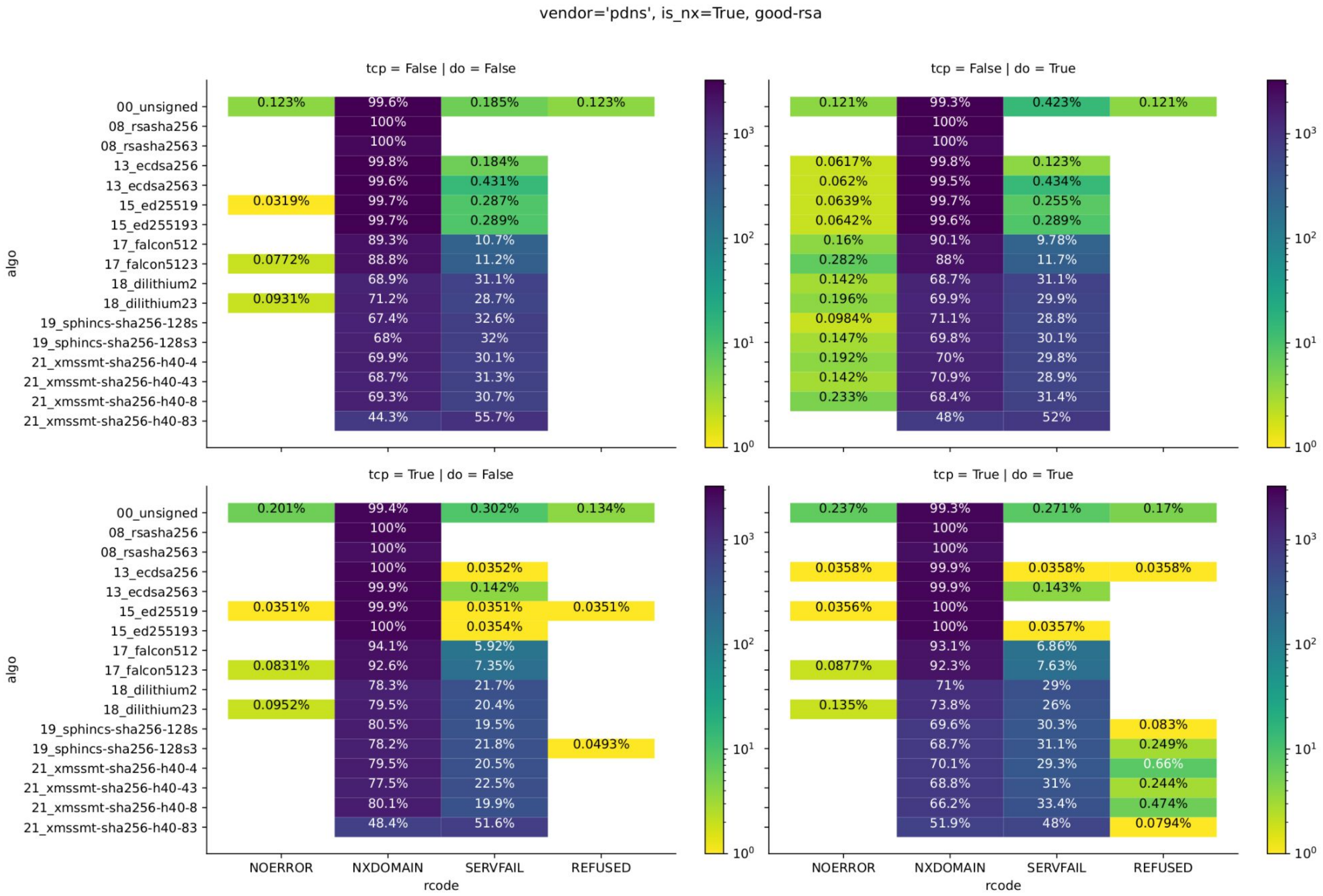
- In 2022, performed (local-only) DNSSEC study with **Falcon** in PowerDNS
  - Results: https://blog.powerdns.com/2022/04/07/falcon-512-in-powerdns
- Now: Broader experiments with **multiple PQC algorithms**
  - fast validation, short signatures, short-ish keys
- Goal: **Public deployment** on the Internet, to investigate …
  - **behavior of non-PQC-aware resolvers** typically used by clients
  - **behavior of PQC-aware resolvers**
- Parameters:
  - KSK/ZSK (BIND) vs. CSK (PowerDNS)
  - Name existence and NSEC vs. [NSEC3 conventional (BIND) vs. minimal (PowerDNS)]
  - UDP vs. TCP
  - DO bit

Algorithm Considerations

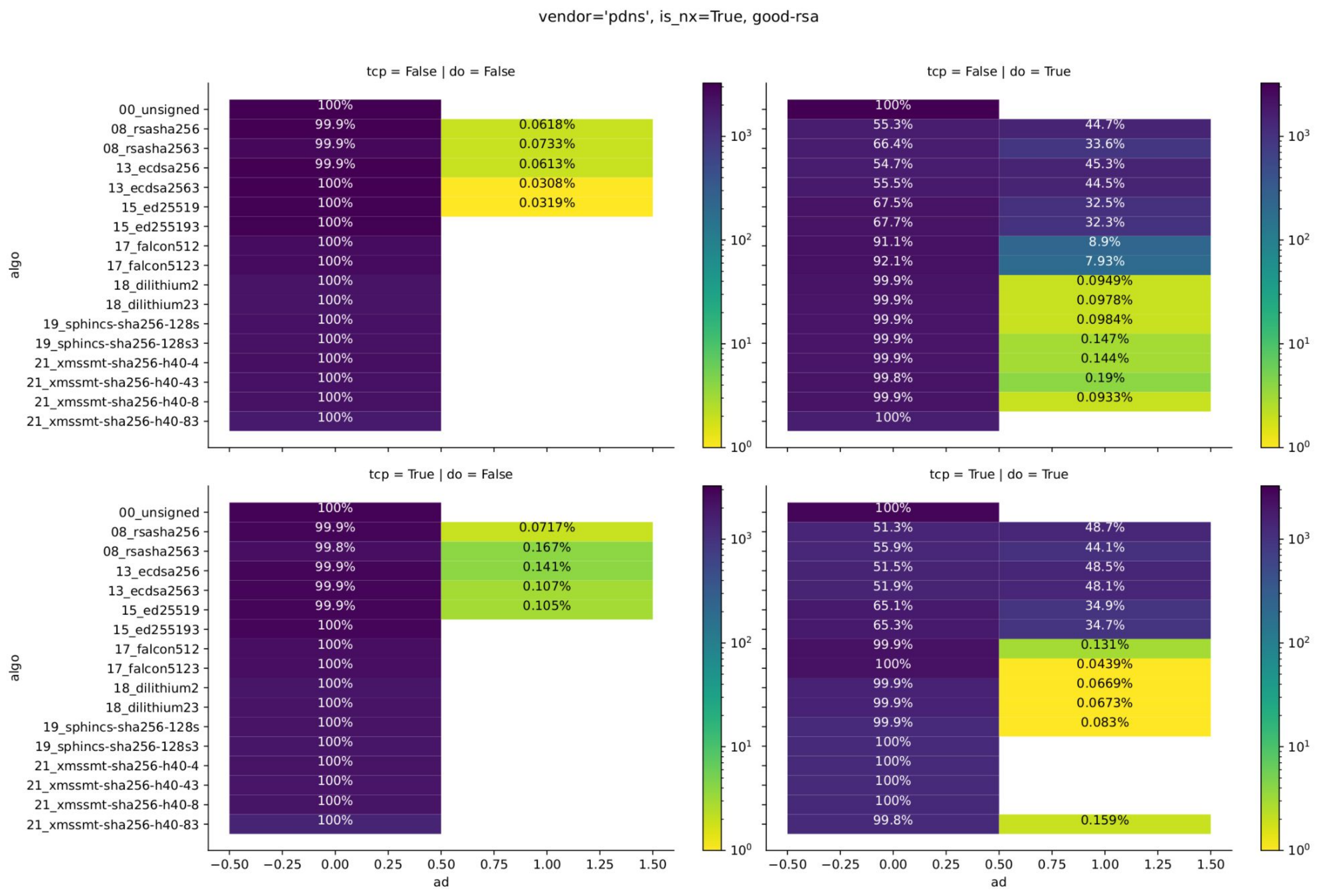
Algorithm	NIST Verdict	Approach	Private key	Public key	Signature	Sign/s	Verify/s
Crystals-Dilithium-II [29]	Finalist	Lattice	2.8kB	1.2kB	2.0kB		
Falcon-512 [31]	Finalist	Lattice	57kB	0.9kB	0.7kB	3,307	20,228
Rainbow-I_a [56]	Finalist	Multivariate	101kB	158kB	66B	8,332	11,065
RedGeMSS128 [16]	Candidate	Multivariate	16B	375kB	35B	545	10,365
Sphincs$^+$-Haraka-128s [11]	Candidate	Hash	64B	32B	8kB		
Picnic-L1-FS [17]	Candidate	Hash	16B	32B	34kB		
Picnic2-L1-FS [17]	Candidate	Hash	16B	32B	14kB		
EdDSA-Ed22519 [12]		Elliptic curve	64B	32B	64B	25,935	7,954
ECDSA-P256 [12]		Elliptic curve	96B	64B	64B	40,509	13,078
RSA-2048 [12]		Prime	2kB	0.3kB	0.3kB	1,485	49,367

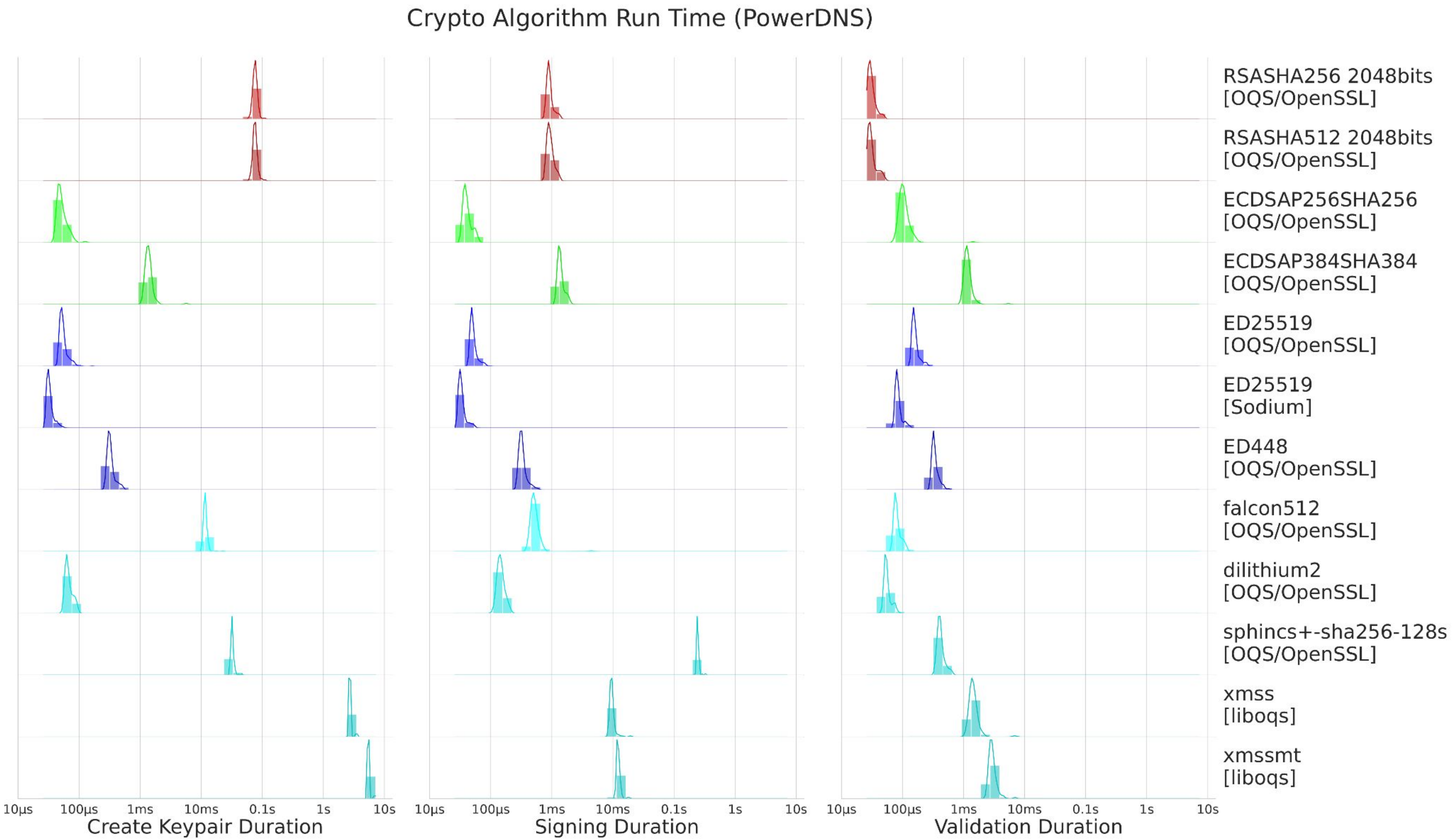
Müller, M. et al.: Retrofitting post-quantum cryptography in internet protocols: a case study of DNSSEC. SIGCOMM Comput. Commun. Rev. 50, 49–57 (2020)

- Selected algorithms with public keys and signatures < 10 KB
- Plus: a stateful hash-based algorithm (XMSS)



vendor='pdns', is_nx=True, good-rsa
tcp = False | do = False
tcp = False | do = True
tcp = True | do = False
tcp = True | do = True
algo
rcode
NOERROR
NXDOMAIN
SERVFAIL
REFUSED
00_unsigned
08_rsasha256
08_rsasha2563
13_ecdsa256
13_ecdsa2563
15_ed25519
15_ed255193
17_falcon512
17_falcon5123
18_dilithium2
18_dilithium23
19_sphincs-sha256-128s
19_sphincs-sha256-128s3
21_xmssmt-sha256-h40-4
21_xmssmt-sha256-h40-43
21_xmssmt-sha256-h40-8
21_xmssmt-sha256-h40-83
0.123% 99.6% 0.185% 0.123%
100%
100%
99.8% 0.184%
99.6% 0.431%
0.0319% 99.7% 0.287%
99.7% 0.289%
89.3% 10.7%
0.0772% 88.8% 11.2%
68.9% 31.1%
0.0931% 71.2% 28.7%
67.4% 32.6%
68% 32%
69.9% 30.1%
68.7% 31.3%
69.3% 30.7%
44.3% 55.7%
0.121% 99.3% 0.423% 0.121%
100%
100%
0.0617% 99.8% 0.123%
0.062% 99.5% 0.434%
0.0639% 99.7% 0.255%
0.0642% 99.6% 0.289%
0.16% 90.1% 9.78%
0.282% 88% 11.7%
0.142% 68.7% 31.1%
0.196% 69.9% 29.9%
0.0984% 71.1% 28.8%
0.147% 69.8% 30.1%
0.192% 70% 29.8%
0.142% 70.9% 28.9%
0.233% 68.4% 31.4%
48% 52%
0.201% 99.4% 0.302% 0.134%
100%
100%
100% 0.0352%
99.9% 0.142%
0.0351% 99.9% 0.0351% 0.0351%
100% 0.0354%
94.1% 5.92%
0.0831% 92.6% 7.35%
78.3% 21.7%
0.0952% 79.5% 20.4%
80.5% 19.5%
78.2% 21.8% 0.0493%
79.5% 20.5%
77.5% 22.5%
80.1% 19.9%
48.4% 51.6%
0.237% 99.3% 0.271% 0.17%
100%
100%
0.0358% 99.9% 0.0358% 0.0358%
99.9% 0.143%
0.0356% 100%
100% 0.0357%
93.1% 6.86%
0.0877% 92.3% 7.63%
71% 29%
0.135% 73.8% 26%
69.6% 30.3% 0.083%
68.7% 31.1% 0.249%
70.1% 29.3% 0.66%
68.8% 31% 0.244%
66.2% 33.4% 0.474%
51.9% 48% 0.0794%
10^0
10^1
10^2
10^3


vendor='pdns', is_nx=True, good-rsa
tcp = False | do = False
tcp = False | do = True
tcp = True | do = False
tcp = True | do = True
algo
ad
00_unsigned
08_rsasha256
08_rsasha2563
13_ecdsa256
13_ecdsa2563
15_ed25519
15_ed255193
17_falcon512
17_falcon5123
18_dilithium2
18_dilithium23
19_sphincs-sha256-128s
19_sphincs-sha256-128s3
21_xmssmt-sha256-h40-4
21_xmssmt-sha256-h40-43
21_xmssmt-sha256-h40-8
21_xmssmt-sha256-h40-83



Crypto Algorithm Run Time (PowerDNS)
RSASHA256 2048bits [OQS/OpenSSL]
RSASHA512 2048bits [OQS/OpenSSL]
ECDSAP256SHA256 [OQS/OpenSSL]
ECDSAP384SHA384 [OQS/OpenSSL]
ED25519 [OQS/OpenSSL]
ED25519 [Sodium]
ED448 [OQS/OpenSSL]
falcon512 [OQS/OpenSSL]
dilithium2 [OQS/OpenSSL]
sphincs+-sha256-128s [OQS/OpenSSL]
xmss [liboqs]
xmssmt [liboqs]
10µs 100µs 1ms 10ms 0.1s 1s 10s
Create Keypair Duration
Signing Duration
Validation Duration