PQC FOR DNSSEC

The Good, the Bad and the Ugly

Ondřej Surý, ISC; RIPE 90 Lisbon

PQC SUITABILITY FOR DNSSEC

- Previous study by DESEC:
  - https://pq-dnssec.dedyn.io/
- My Master's Thesis (final draft done, yay!)
  - NIST Round 2 Additional DSS – bunch of new algorithms
    - https://csrc.nist.gov/Projects/post-quantum-cryptography/round-4-submissions
  - + Some interesting new research (not mine)

PQC FOR DNSSEC

- HAWK – lattice-based
- SQISign – isogeny-based
- MAYO – multivariete-based
- Antrag – Espitau, Thomas, Thi Thu Quyen Nguyen, Chao Sun, Mehdi Tibouchi, and Alexandre Wallet. 'Antrag: Annular NTRU Trapdoor Generation', 2023. https://eprint.iacr.org/2023/1335.

PQC FOR DNSSEC

ALGORITHM	NIST Level	SECRET KEY	PUBLIC KEY	SIGNATURE
FALCON-512	1	1281	897	666
HAWK-256	Challenge	96	450	249
HAWK-512	1	184	1024	555
SQIsign	1	353	65	148
MAYO	1	24	1420	454
ANTRAG-512	1	59392	768	592
RSA 2048	n/a	1232	256	256
ECDSAP256	n/a	32	64	64
ED25519	n/a	32	32	64

IMPLEMENTATION STATUS

- FALCON – PQclean implementation; embedded; needed some tweaking
- HAWK – neat little code meant for embedding 🙏
- SQIsign – not suitable for external use (yet), mashed the CMake project into building shared libraries, generic and broadwell implementation 😭
- MAYO – same story, team focused on NIST, mashed into shared library 😭
- ANTRAG – not a shared library, definitely work in progress 😭

LOCAL TESTING METHODOLOGY

- System 76 Meerkat
  - 22-core Intel(R) Core(TM) Ultra 7 155H
  - HT-disabled, Turbo-Boost disabled (so, basically using just **6 cores**)
- Using Hyperfine for measurements
- Use *tmpfs* for storing files
- Use the «ref» implementation (not the assembly versions)

IMPLEMENTATION IN BIND 9

- RSA 2048, ECDSA255, Ed25519 – *ondrej/pqc-main* branch; EDNS(0) buffer size bumped to 1452
  - BASE – *ondrej/pqc-base* branch; custom root server
    - FALCON-512 – *ondrej/pqc-falcon-512*, embedded PQclean padded variant
    - HAWK-256 – *ondrej/pqc-hawk-256*, embedded sources
    - HAWK-512 – *ondrej/pqc-hawk-512* embedded sources; max buffers bumped
    - SQIsign – *ondrej/pqc-sqisign*, embedded shared libraries (yeah, I know, awful)
    - MAYO – *ondrej/pqc-mayo*, embedded shared libraries (bleh!); max buffers bumped
    - Antrag-512 – *ondrej/pqc-antrag*, modified test suite for embedding and fixed some broken C

KEY GENERATION

ALGORITHM	MEAN	σ
FALCON-512	80.1 ms	11.7 ms
HAWK-256	46.9 ms	1.2 ms
HAWK-512	*51.5 ms*	3.5 ms
SQIsign	97.8 ms	4.4 ms
MAYO	45.1 ms	2.9 ms
ANTRAG-512	71.9 ms	2.6 ms
RSA 2048	493.7 ms	253.8 ms
ECDSAP256	45.1 ms	2.3 ms
ED25519	45.2 ms	2.3 ms

SIGNING (ROOT, 1 KSK, 1 ZSK, RAW)

ALGORITHM	MEAN	σ	SIGNATURES/S	RAW SIZE
FALCON-512	4881.9 ms	26.8 ms	589	2891700
HAWK-256	195.5 ms	4.9 ms	62001	1727793
HAWK-512	261.0 ms	9.6 ms	49821	2582375
SQIsign	*54528.1 ms*	67.9 ms	51	1445334
MAYO	1086.6 ms	48.7 ms	2746	2301478
ANTRAG-512+	5339.6 ms	111.2 ms	546	2685056
RSA 2048	845.7 ms	3.0 ms	3980	1746936
ECDSAP256	218.1 ms	10.2 ms	44286	1211056
ED25519	240.6 ms	6.3 ms	47288	1210992

+ Single threaded

DNS MESSAGE SIZES (ROOT, 1 KSK, 1 ZSK)

ALGORITHM	SOA	DNSKEY	NXDOMAIN	NODATA	Delegation
FALCON-512	797	3244	1520	1518	1023
HAWK-256	380	1237	686	684	606
HAWK-512	686	2691	1298	1296	912
SQIsign	279	366	484	482	505
MAYO	1108	3382	1096	1094	811
ANTRAG-512	723	2216	1372	1370	949
RSA 2048	387	864	700	698	613
ECDSAP256	195	280	316	314	421
ED25519	195	216	316	314	421

Doesn't fit into 1232!

Doesn't fit into 1452!

VALIDATION (ROOT, 1 KSK, 1 ZSK, RAW)

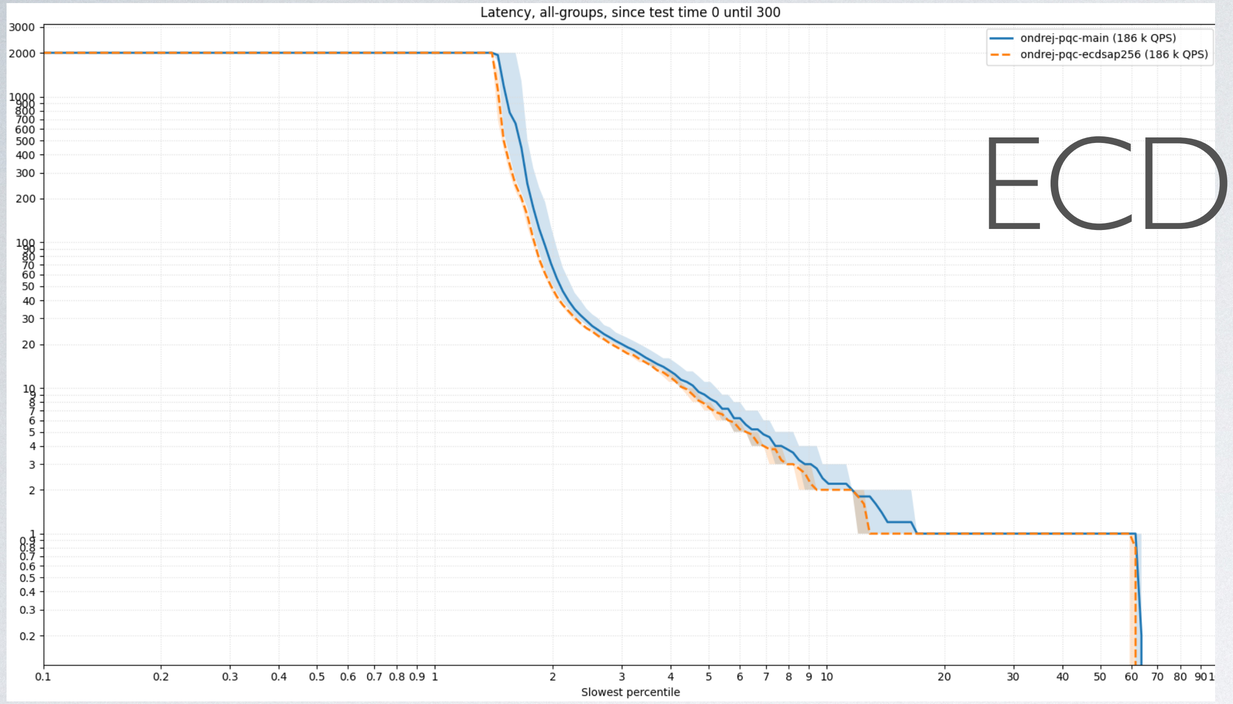
ALGORITHM	MEAN	σ
FALCON-512	403.7 ms	1.1 ms
HAWK-256	232.5 ms	1.4 ms
HAWK-512	359.4 ms	66.0 ms
SQIsign	22338.5 ms	35.0 ms
MAYO	995.8 ms	26.8 ms
ANTRAG-512	548.6 ms	1.4 ms
RSA 2048	250.2 ms	18.9 ms
ECDSAP256	610.0 ms	4.5 ms
ED25519	819.4 ms	4.5 ms

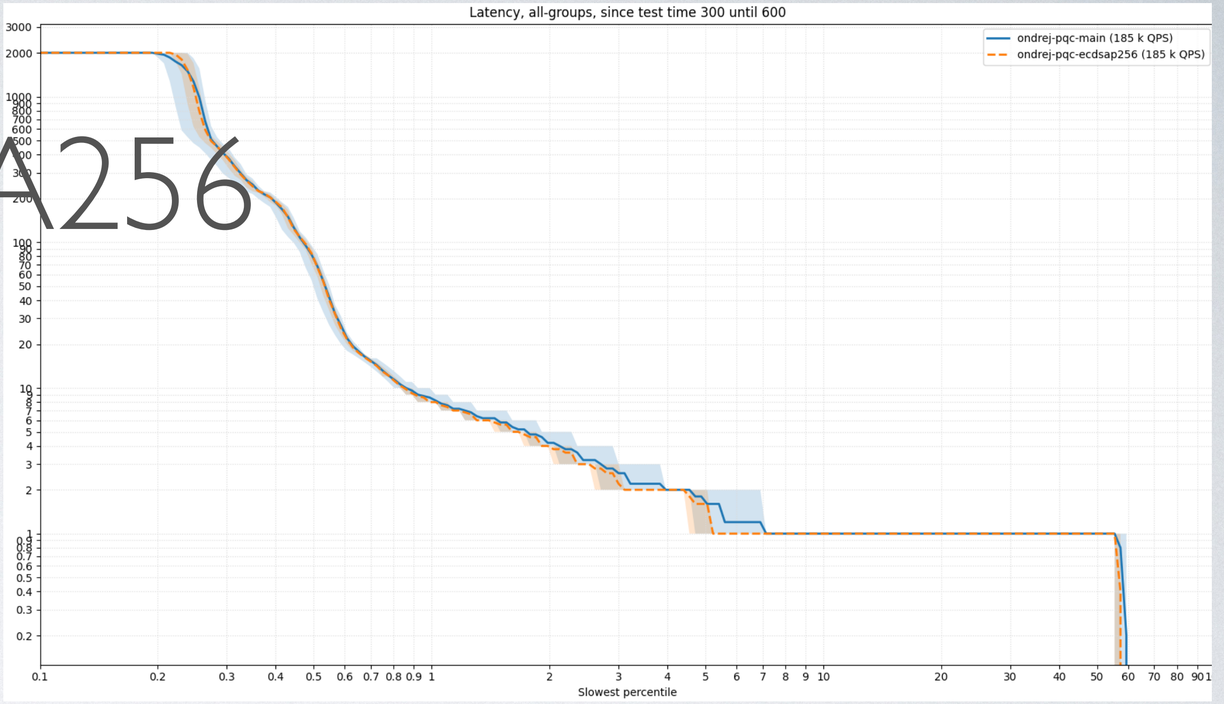
** Warning: Statistical outliers were detected.*

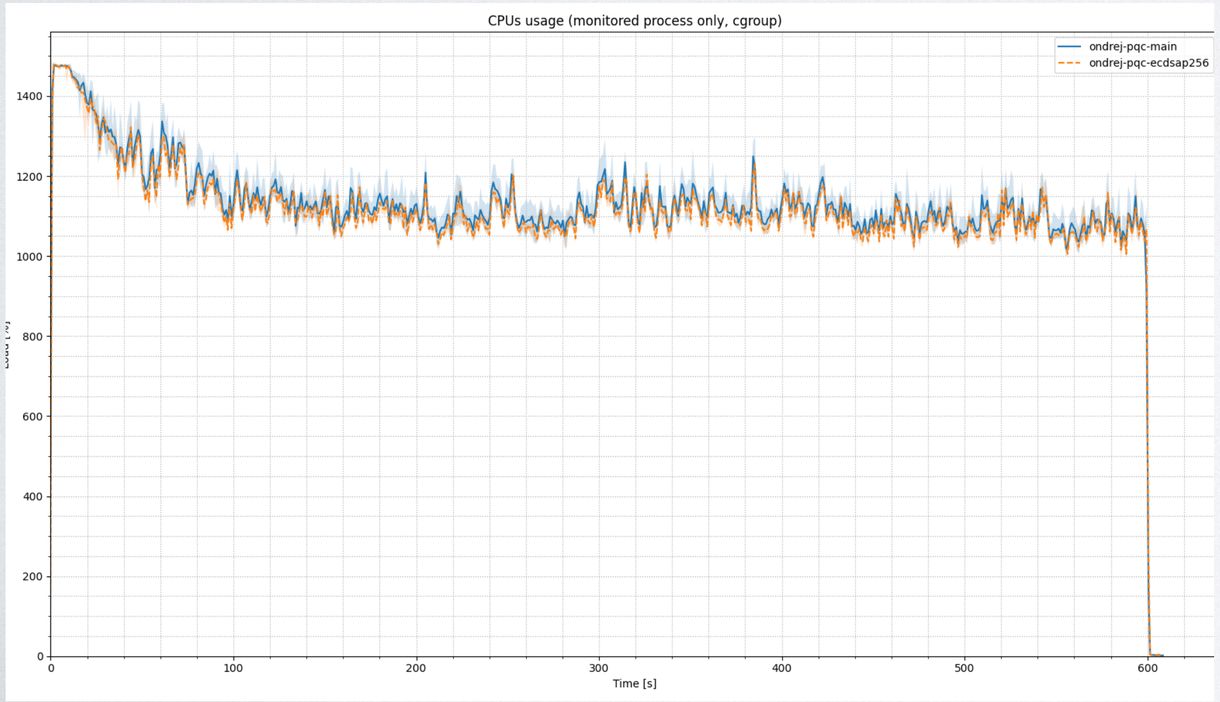
RESOLVER TESTING

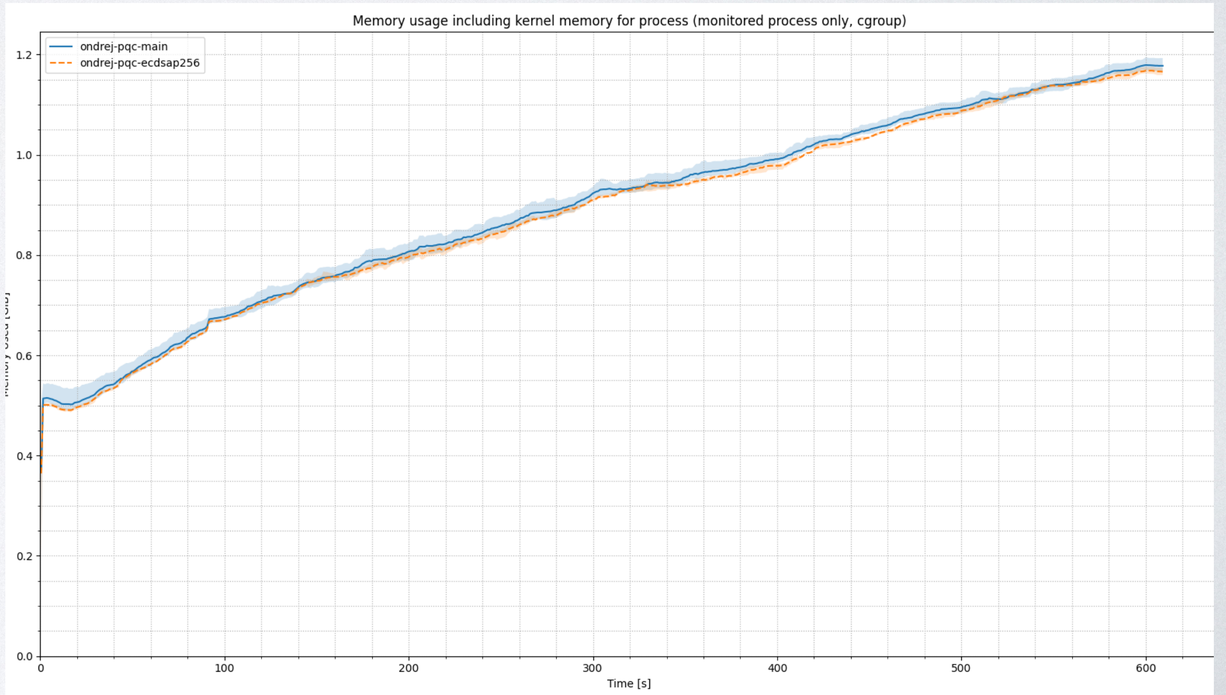
- Real-world resolver data
- Custom root server serving signed root zone
- Using DNS Shotgun – part of the regular BIND 9 testing

ECDSA256

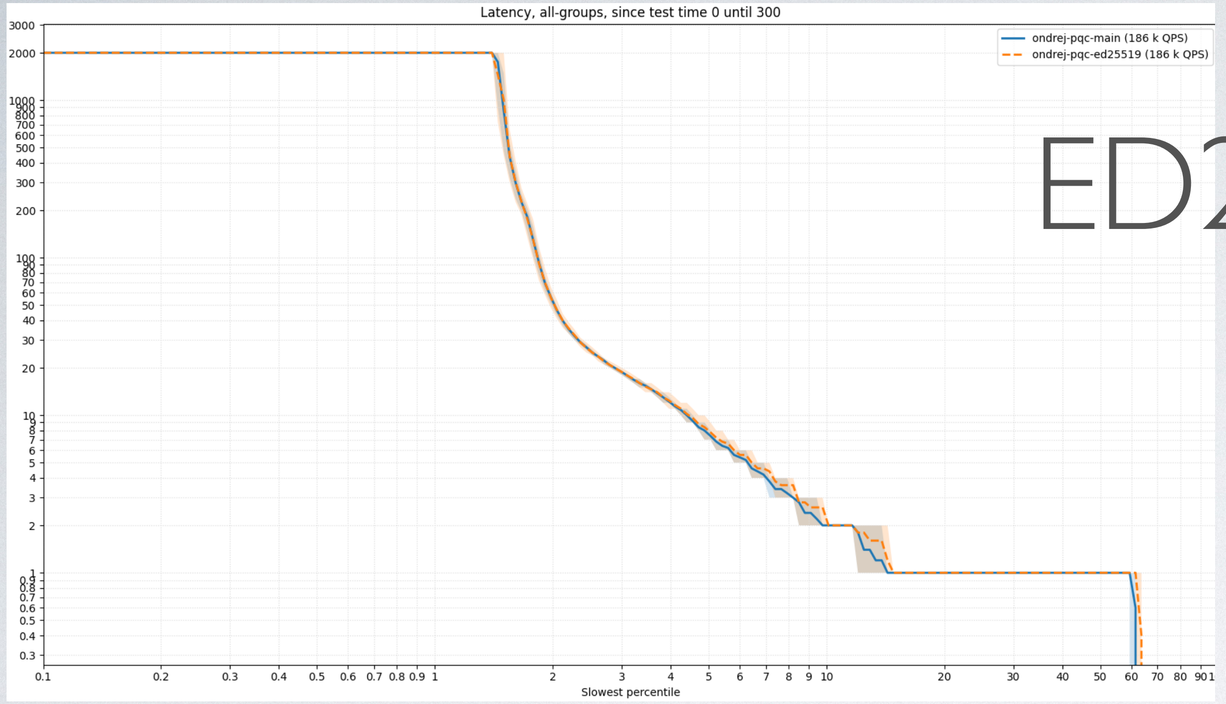

Latency, all-groups, since test time 0 until 300
ondrej-pqc-main (186 k QPS)
ondrej-pqc-ecdsap256 (186 k QPS)
Slowest percentile

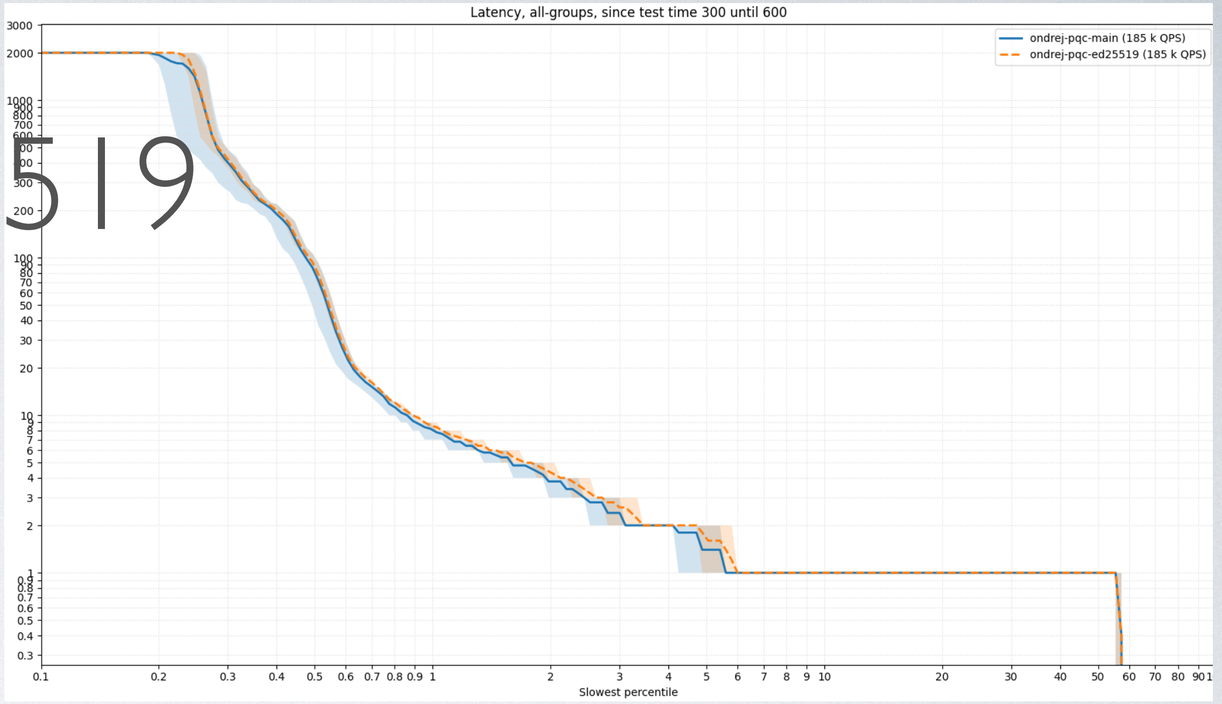

Latency, all-groups, since test time 300 until 600
ondrej-pqc-main (185 k QPS)
ondrej-pqc-ecdsap256 (185 k QPS)
Slowest percentile

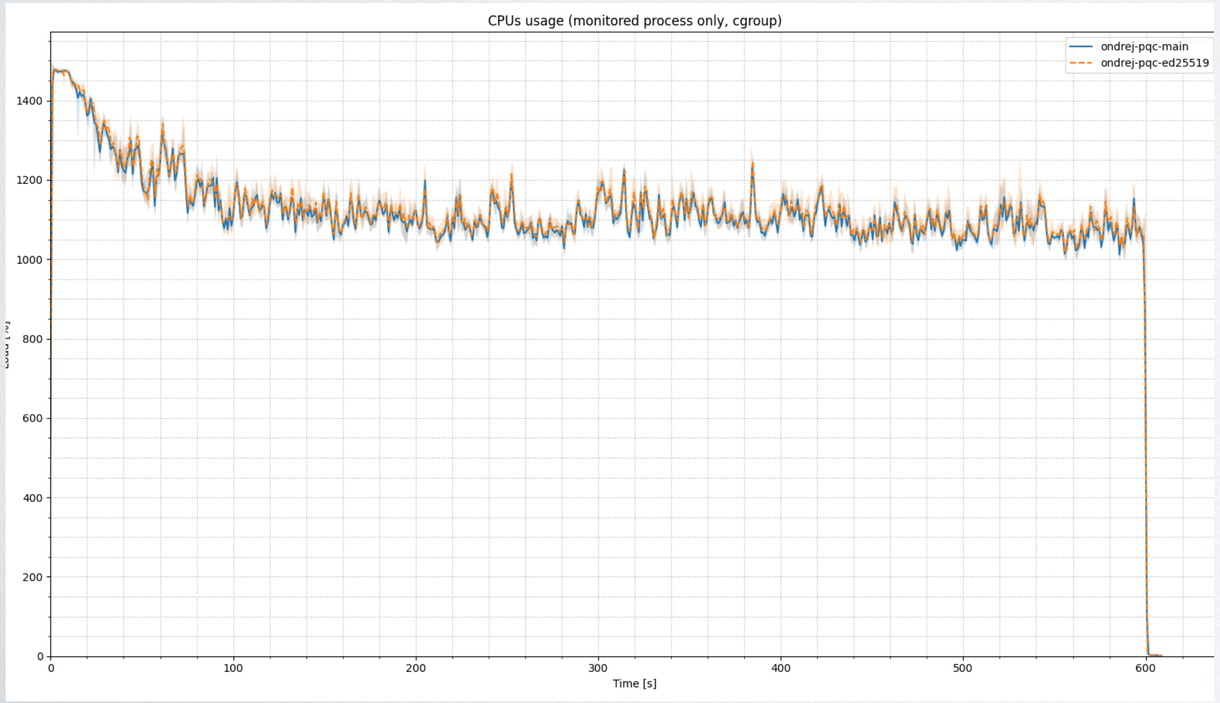

CPUs usage (monitored process only, cgroup)
ondrej-pqc-main
ondrej-pqc-ecdsap256
Time [s]

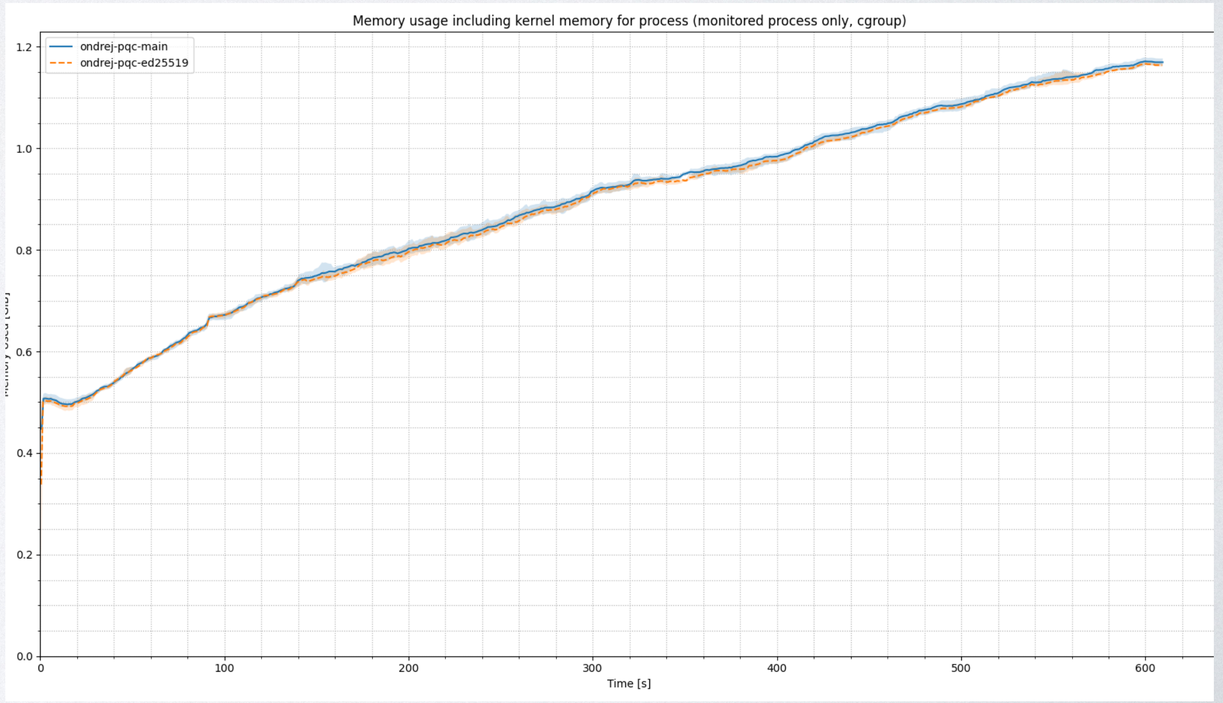

Memory usage including kernel memory for process (monitored process only, cgroup)
ondrej-pqc-main
ondrej-pqc-ecdsap256
Time [s]

ED25519

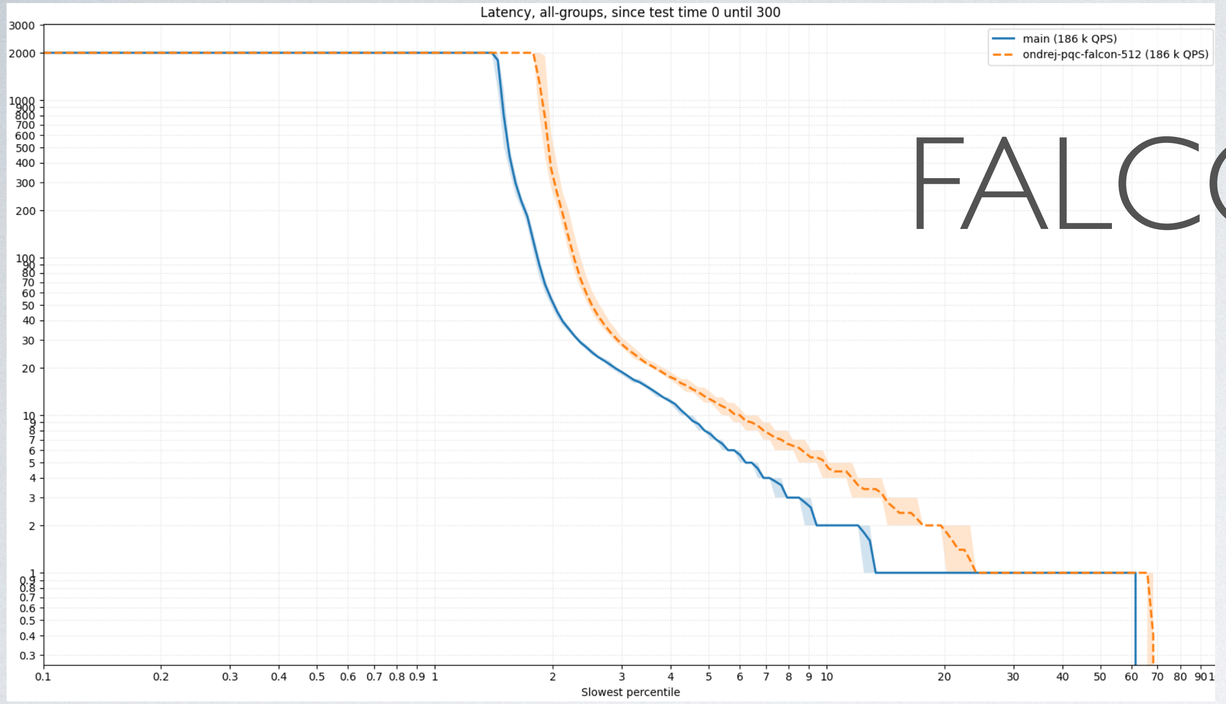

Latency, all-groups, since test time 0 until 300
ondrej-pqc-main (186 k QPS)
ondrej-pqc-ed25519 (186 k QPS)
Slowest percentile

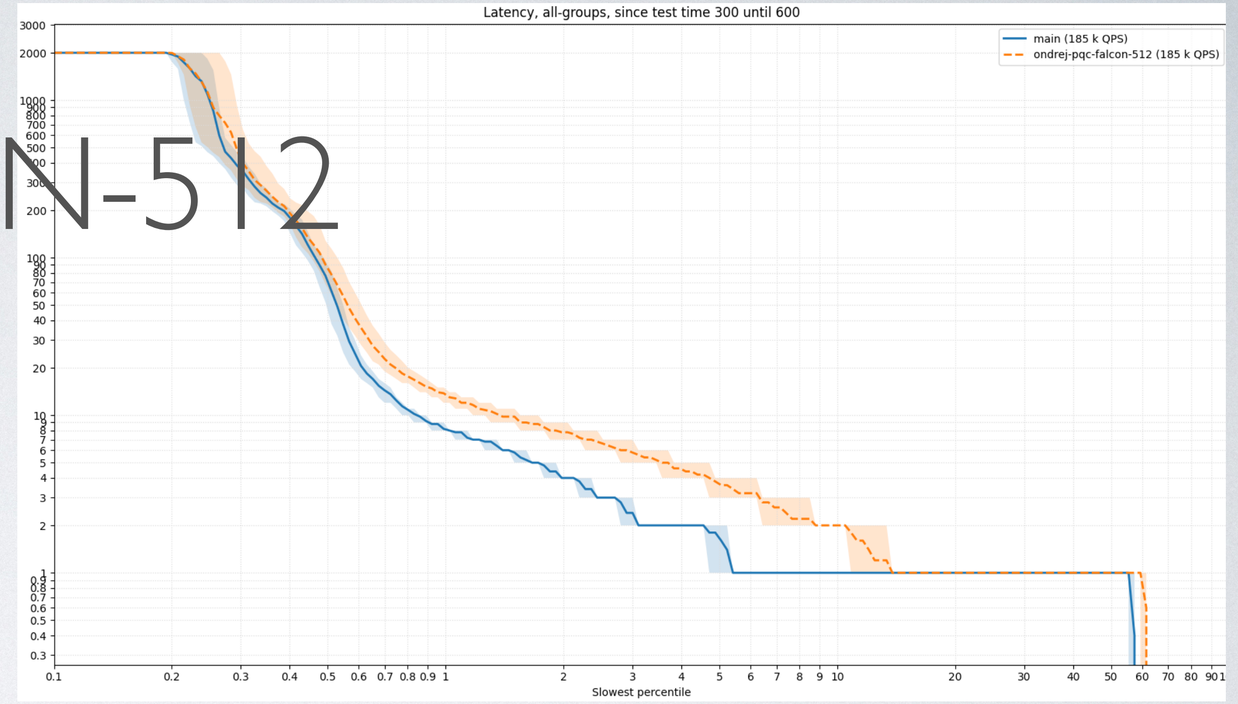

Latency, all-groups, since test time 300 until 600
ondrej-pqc-main (185 k QPS)
ondrej-pqc-ed25519 (185 k QPS)
Slowest percentile


CPUs usage (monitored process only, cgroup)
ondrej-pqc-main
ondrej-pqc-ed25519
Time [s]

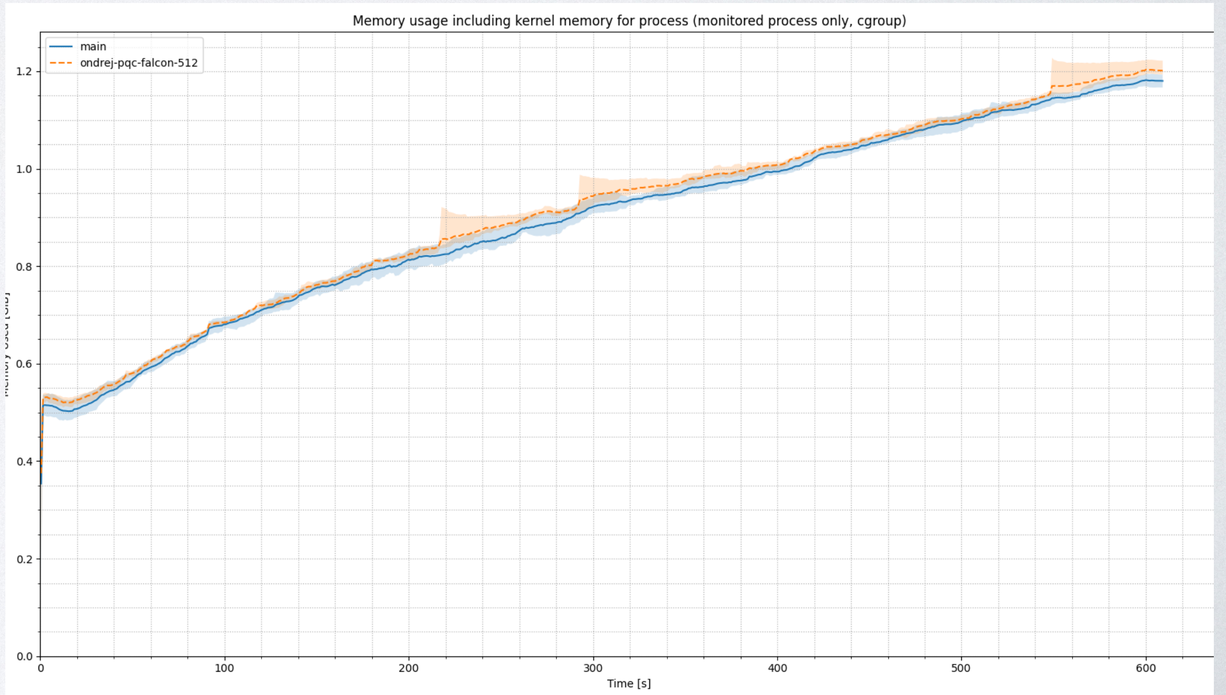

Memory usage including kernel memory for process (monitored process only, cgroup)
ondrej-pqc-main
ondrej-pqc-ed25519
Time [s]

FALCON-512

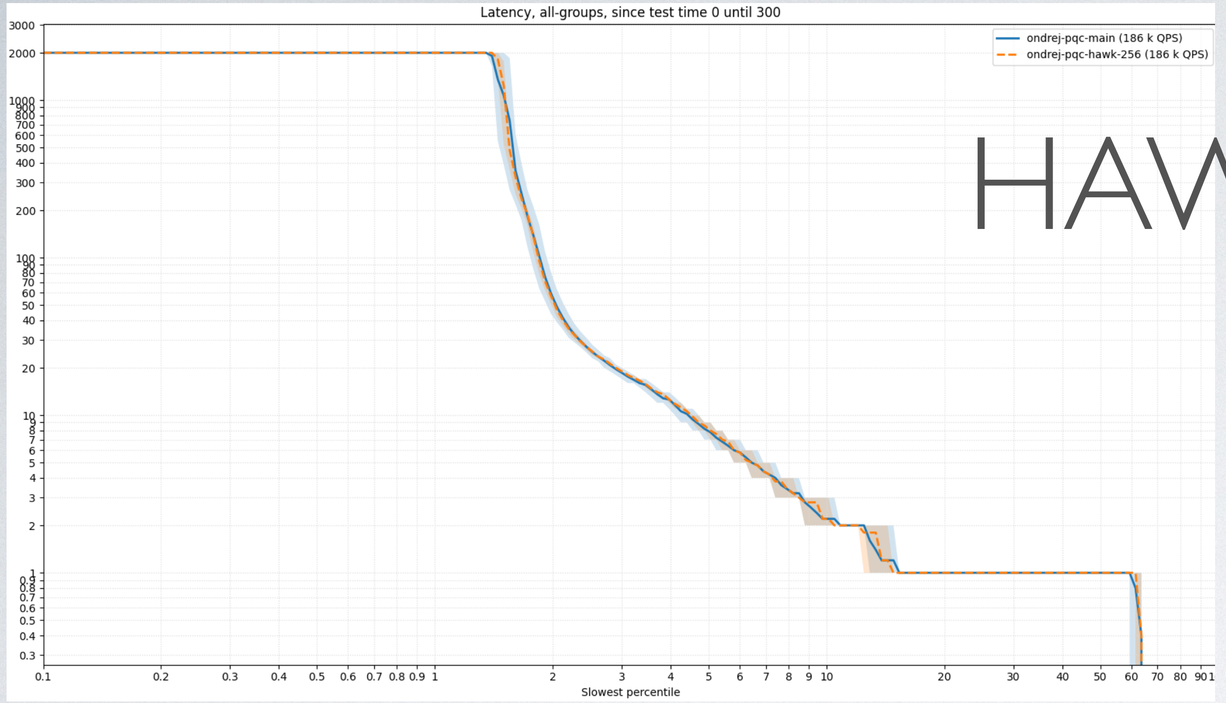

Latency, all-groups, since test time 0 until 300
main (186 k QPS)
ondrej-pqc-falcon-512 (186 k QPS)
Slowest percentile

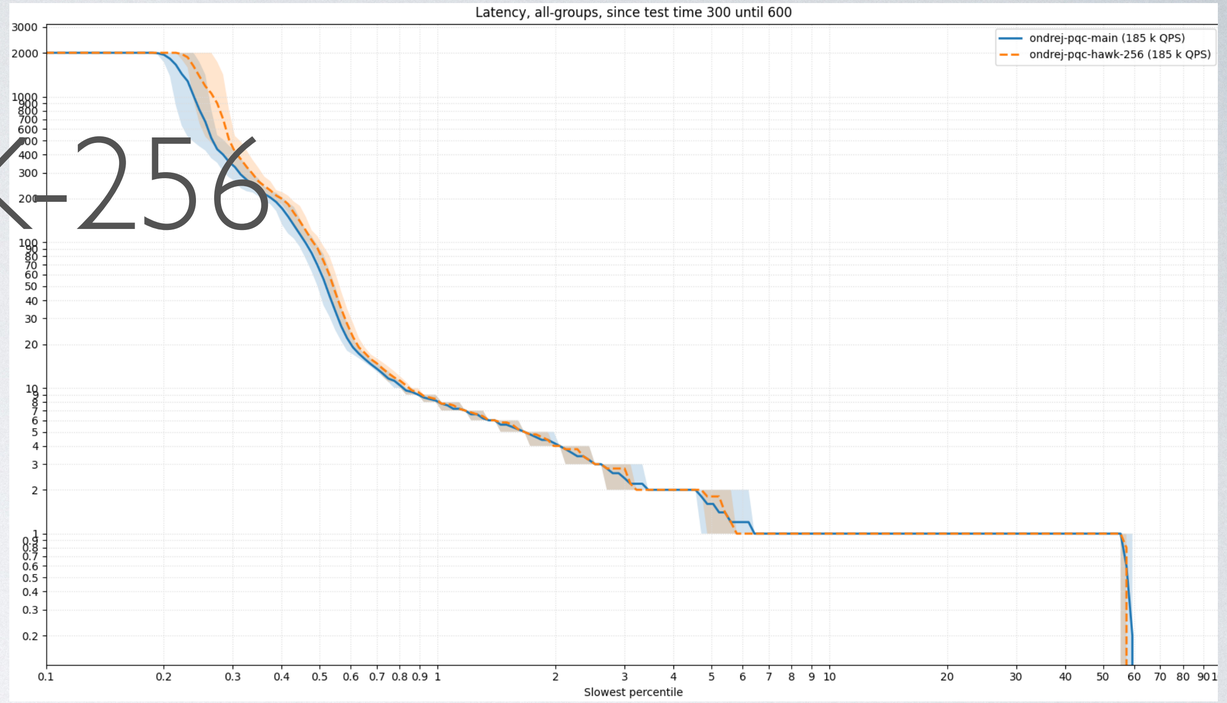

Latency, all-groups, since test time 300 until 600
main (185 k QPS)
ondrej-pqc-falcon-512 (185 k QPS)
Slowest percentile

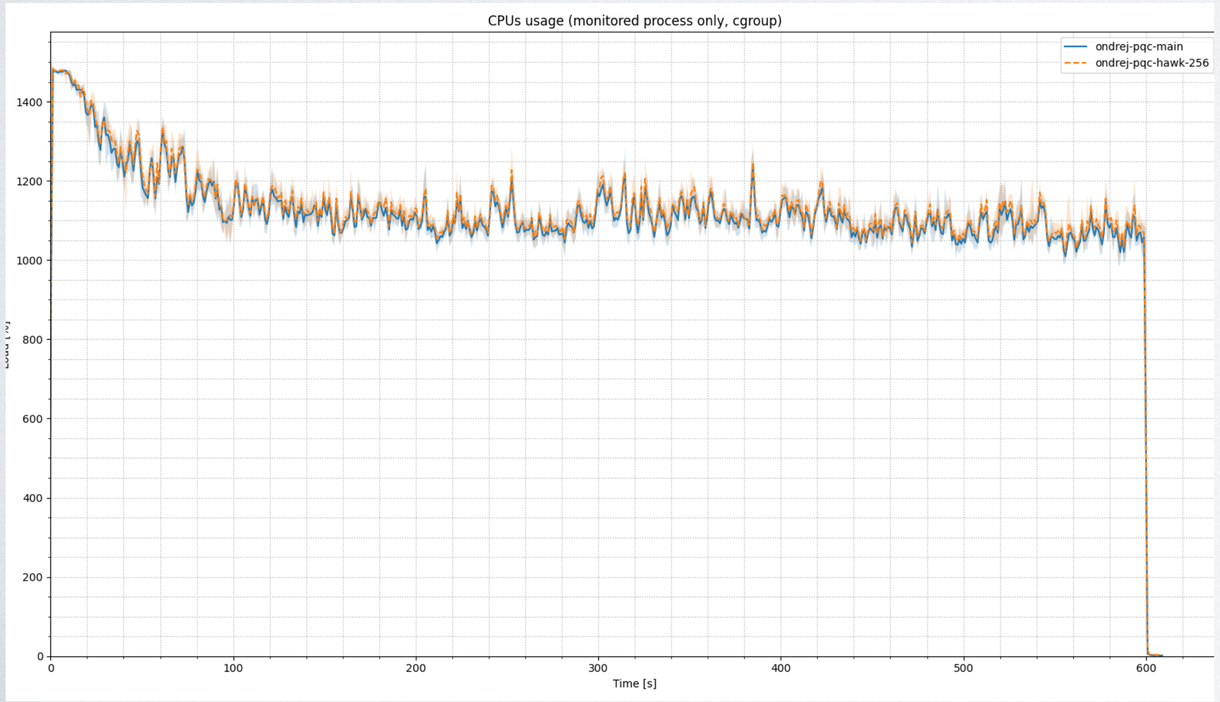

CPUs usage (monitored process only, cgroup)
main
ondrej-pqc-falcon-512
Time [s]

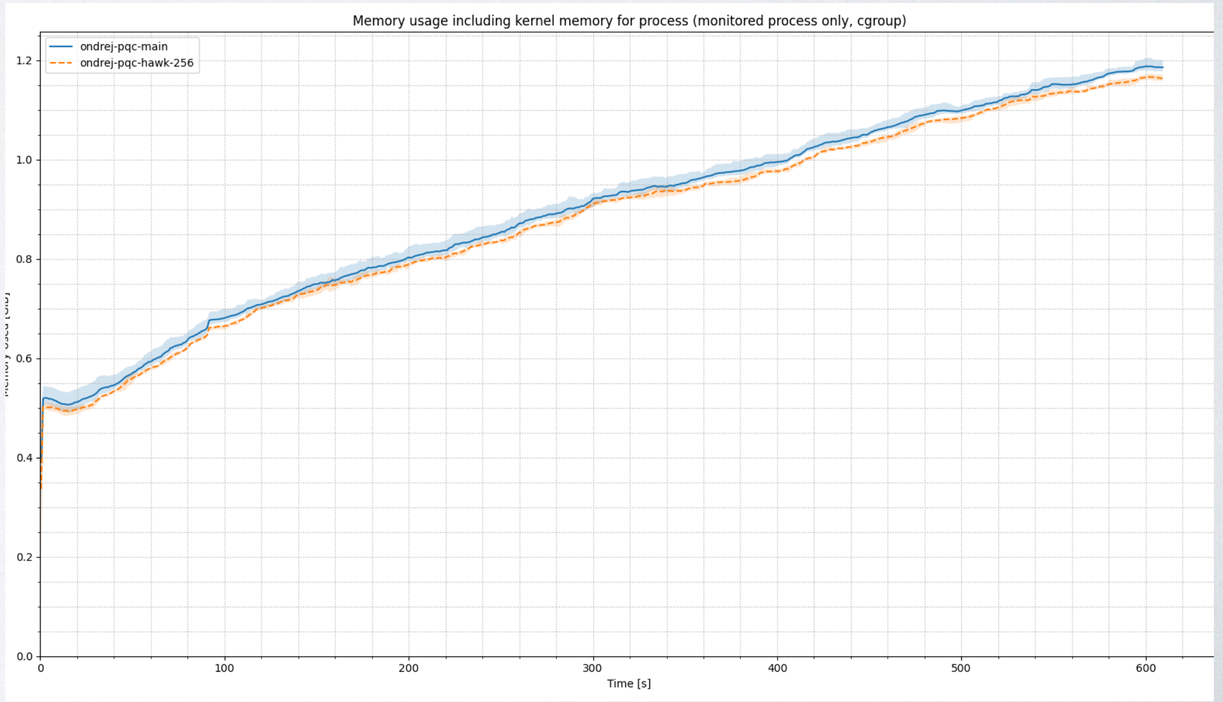

Memory usage including kernel memory for process (monitored process only, cgroup)
main
ondrej-pqc-falcon-512
Time [s]

HAWK-256

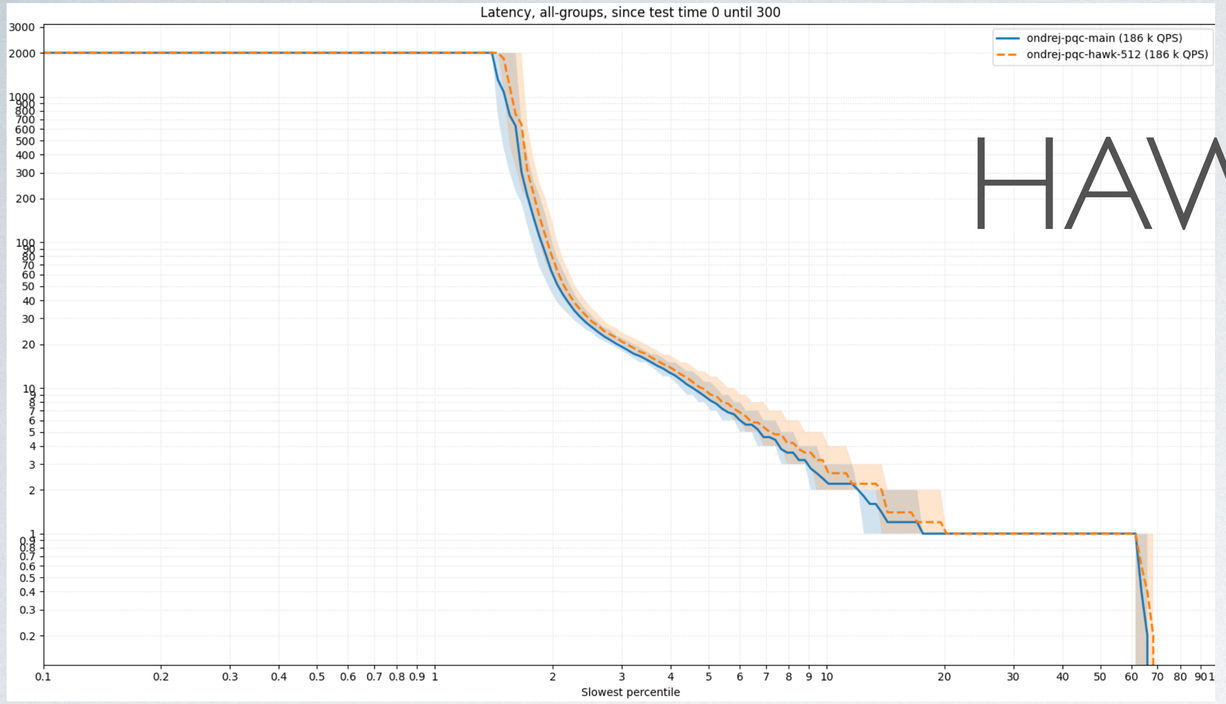

Latency, all-groups, since test time 0 until 300
ondrej-pqc-main (186 k QPS)
ondrej-pqc-hawk-256 (186 k QPS)
Slowest percentile

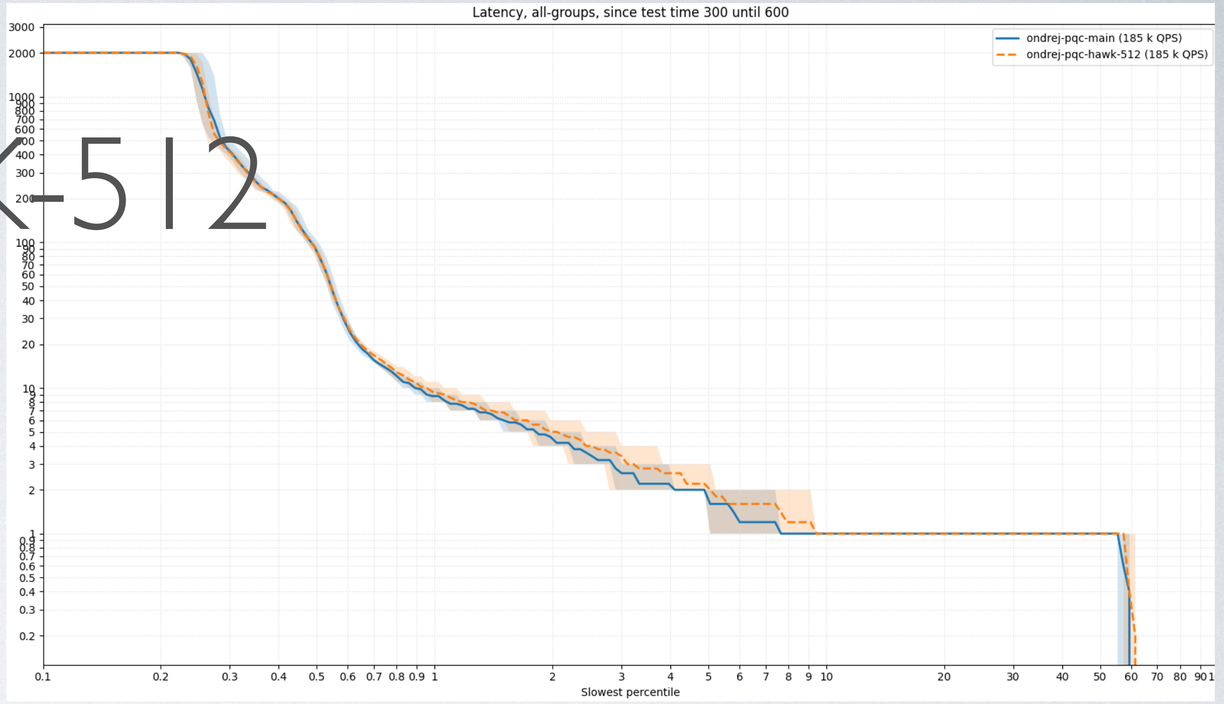

Latency, all-groups, since test time 300 until 600
ondrej-pqc-main (185 k QPS)
ondrej-pqc-hawk-256 (185 k QPS)
Slowest percentile

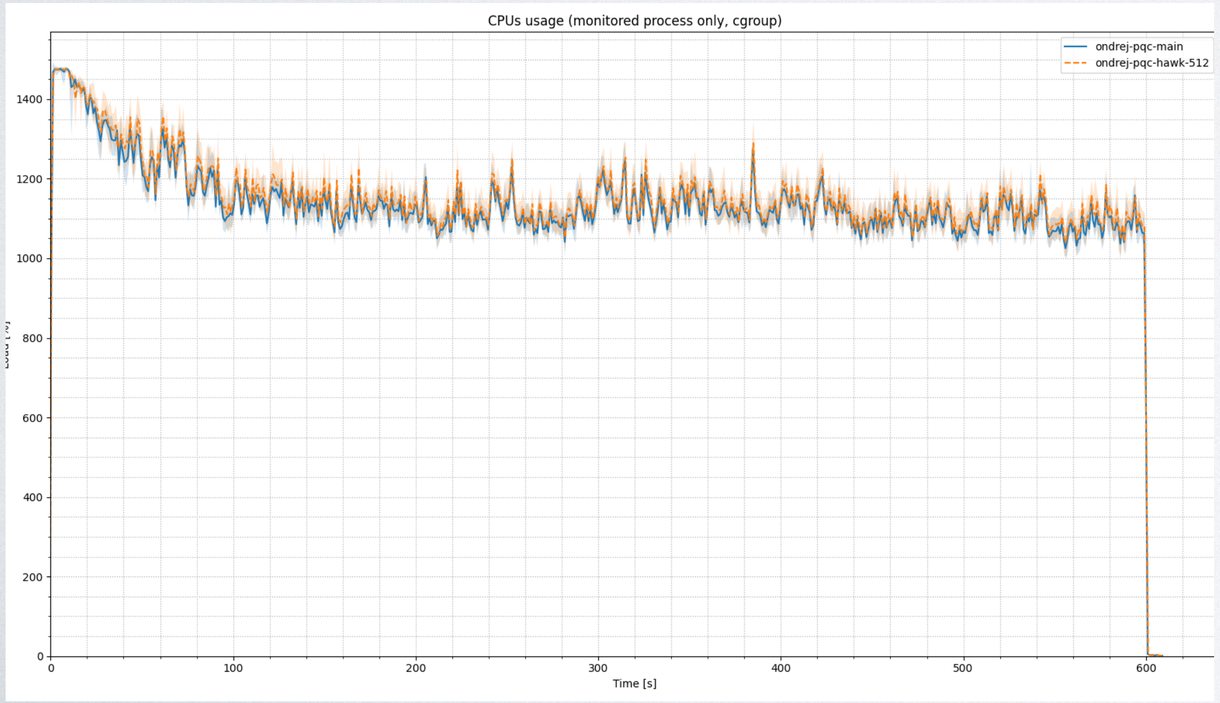

CPUs usage (monitored process only, cgroup)
ondrej-pqc-main
ondrej-pqc-hawk-256
Time [s]

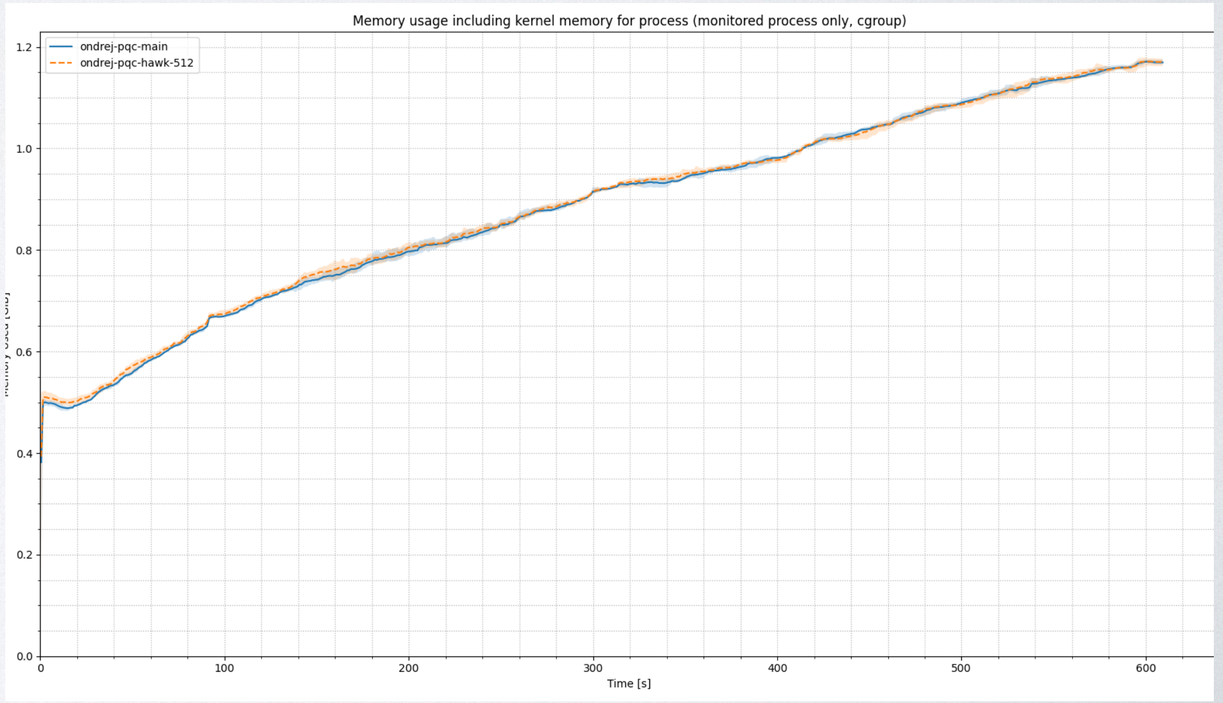

Memory usage including kernel memory for process (monitored process only, cgroup)
ondrej-pqc-main
ondrej-pqc-hawk-256
Time [s]

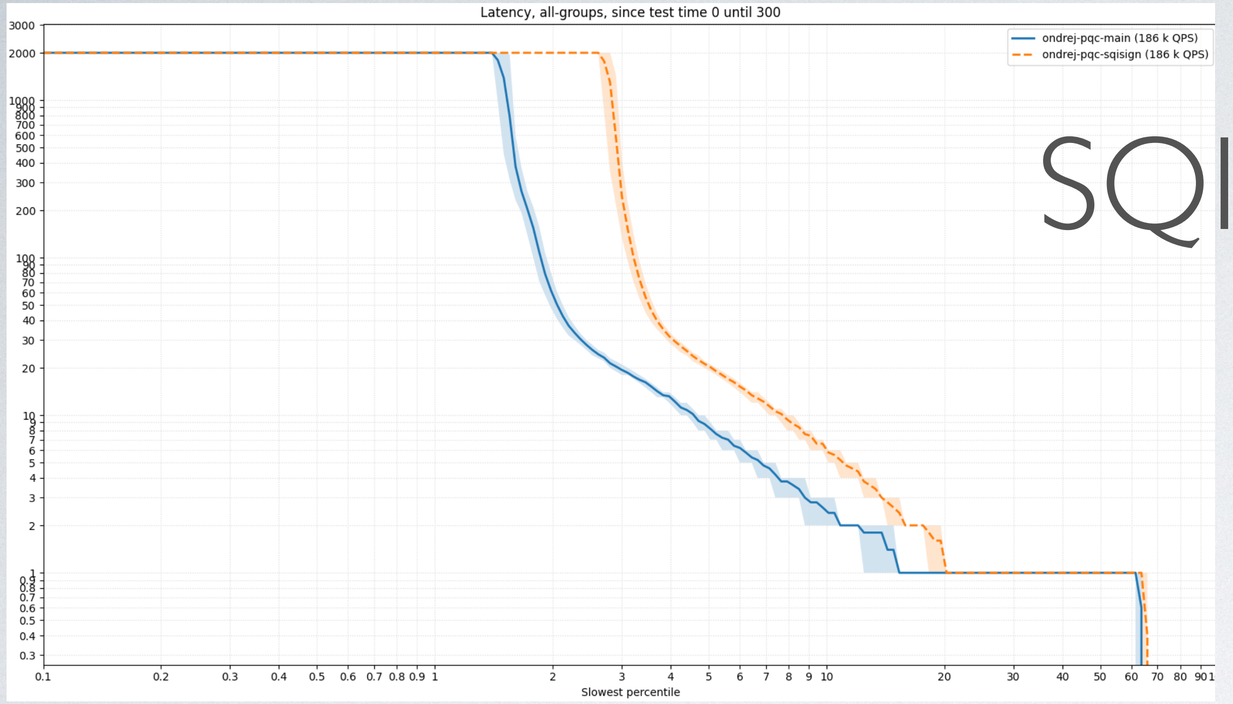
HAWK-512


Latency, all-groups, since test time 0 until 300
ondrej-pqc-main (186 k QPS)
ondrej-pqc-hawk-512 (186 k QPS)
Slowest percentile

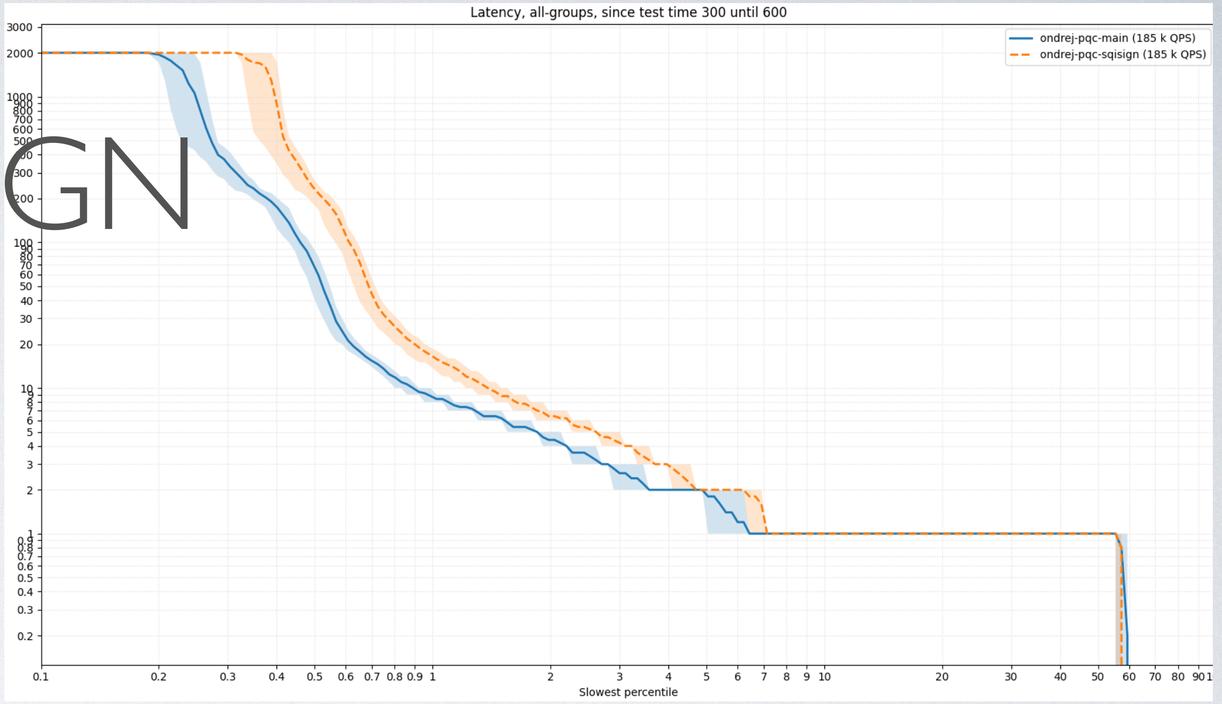

Latency, all-groups, since test time 300 until 600
ondrej-pqc-main (185 k QPS)
ondrej-pqc-hawk-512 (185 k QPS)
Slowest percentile

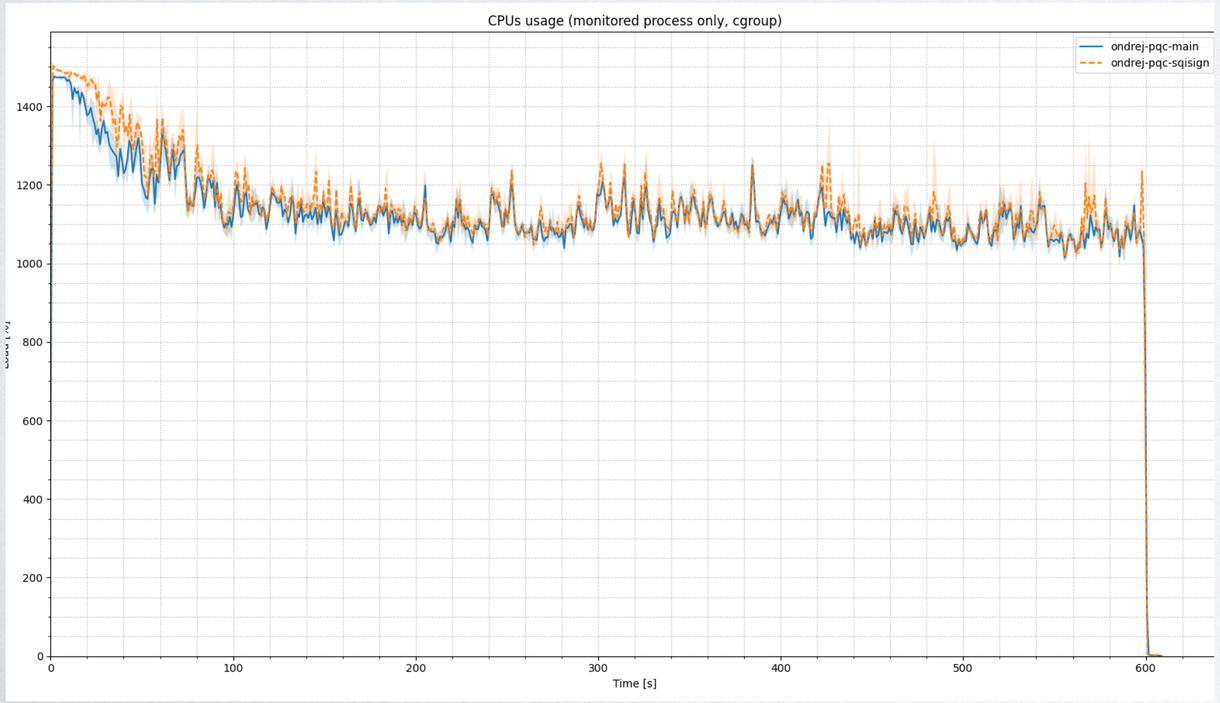

CPUs usage (monitored process only, cgroup)
ondrej-pqc-main
ondrej-pqc-hawk-512
Time [s]

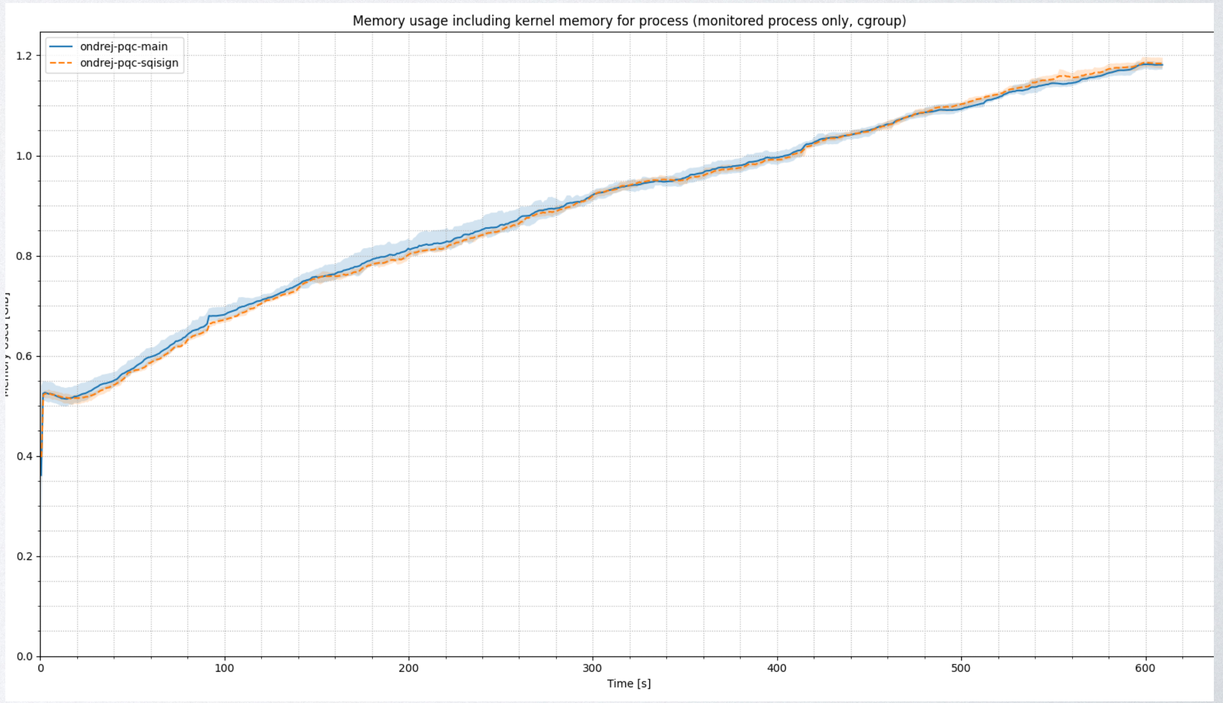

Memory usage including kernel memory for process (monitored process only, cgroup)
ondrej-pqc-main
ondrej-pqc-hawk-512
Time [s]

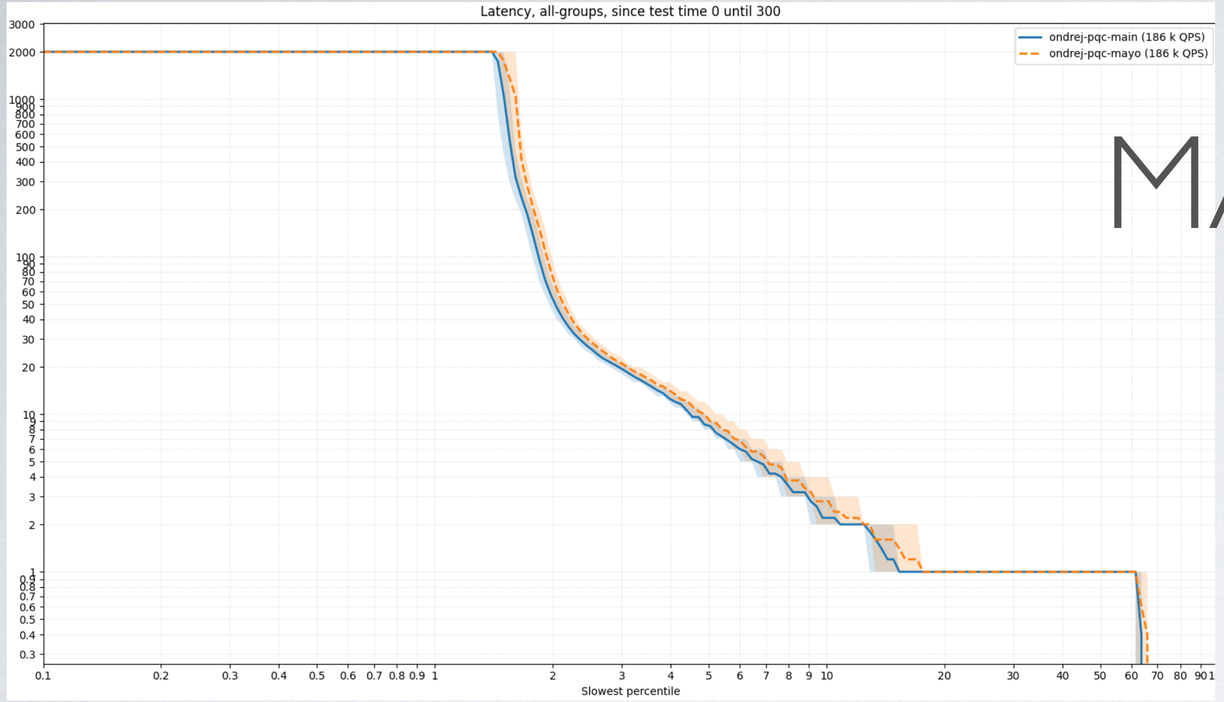

Latency, all-groups, since test time 0 until 300
ondrej-pqc-main (186 k QPS)
ondrej-pqc-sqisign (186 k QPS)
Slowest percentile

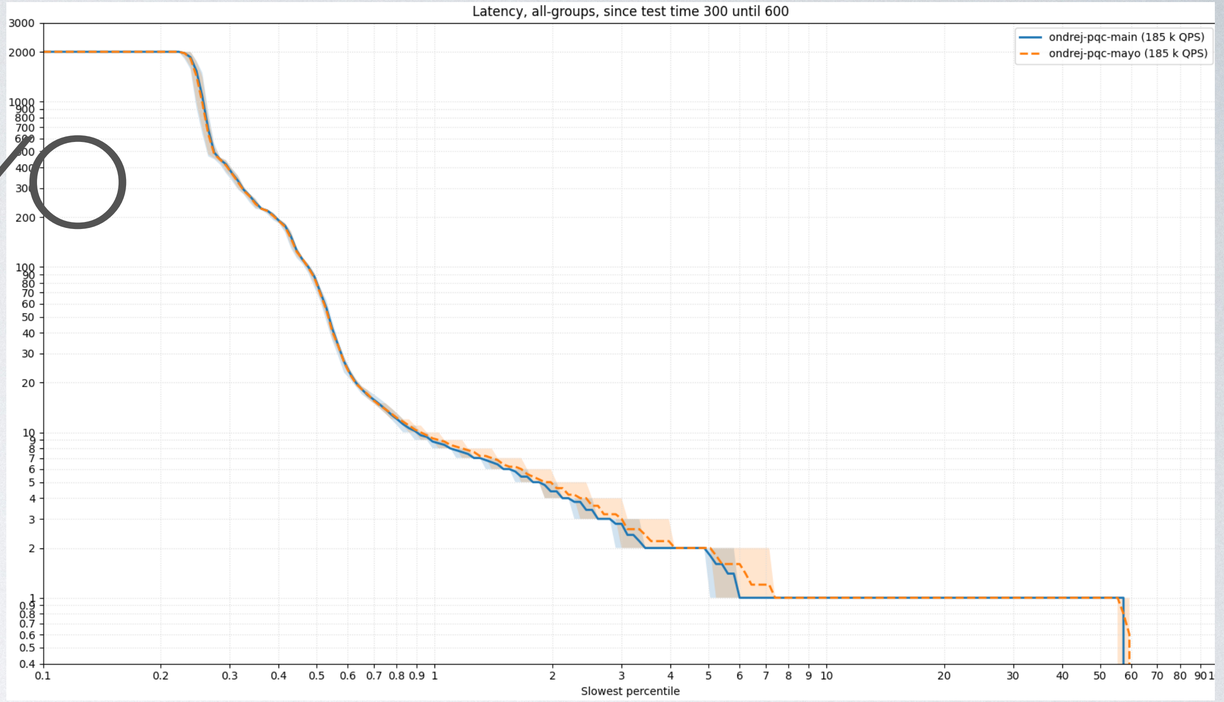
SQISIGN

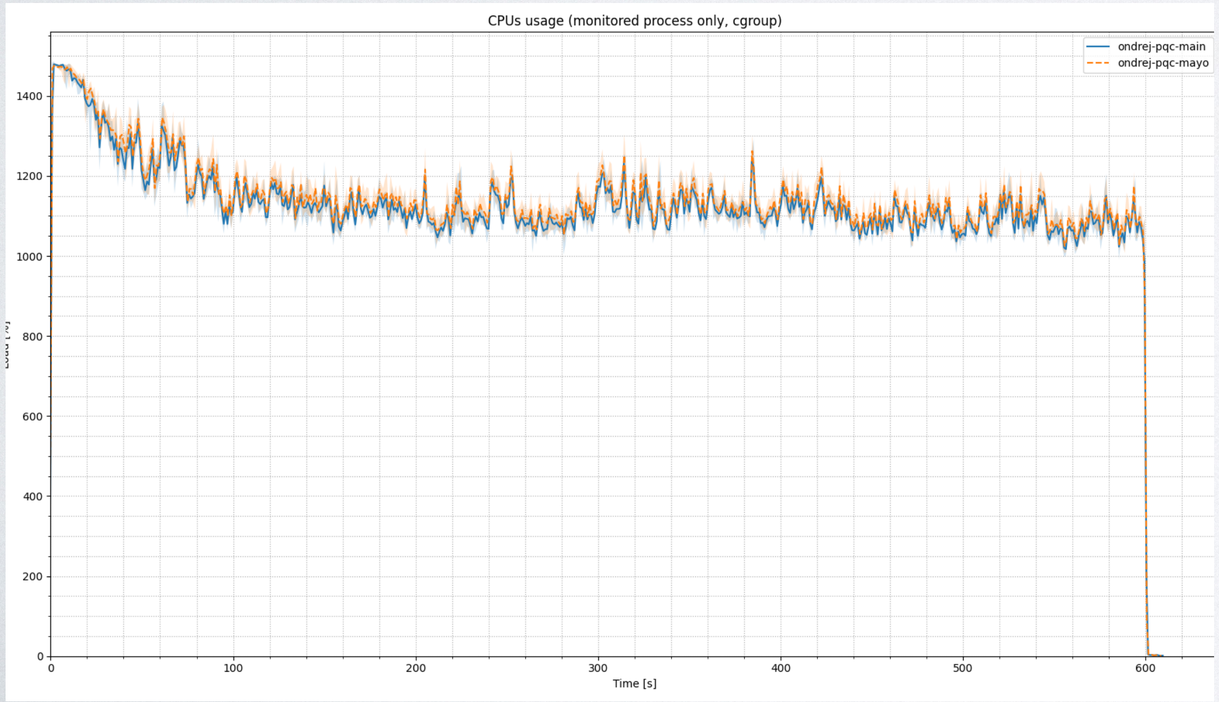

Latency, all-groups, since test time 300 until 600
ondrej-pqc-main (185 k QPS)
ondrej-pqc-sqisign (185 k QPS)
Slowest percentile

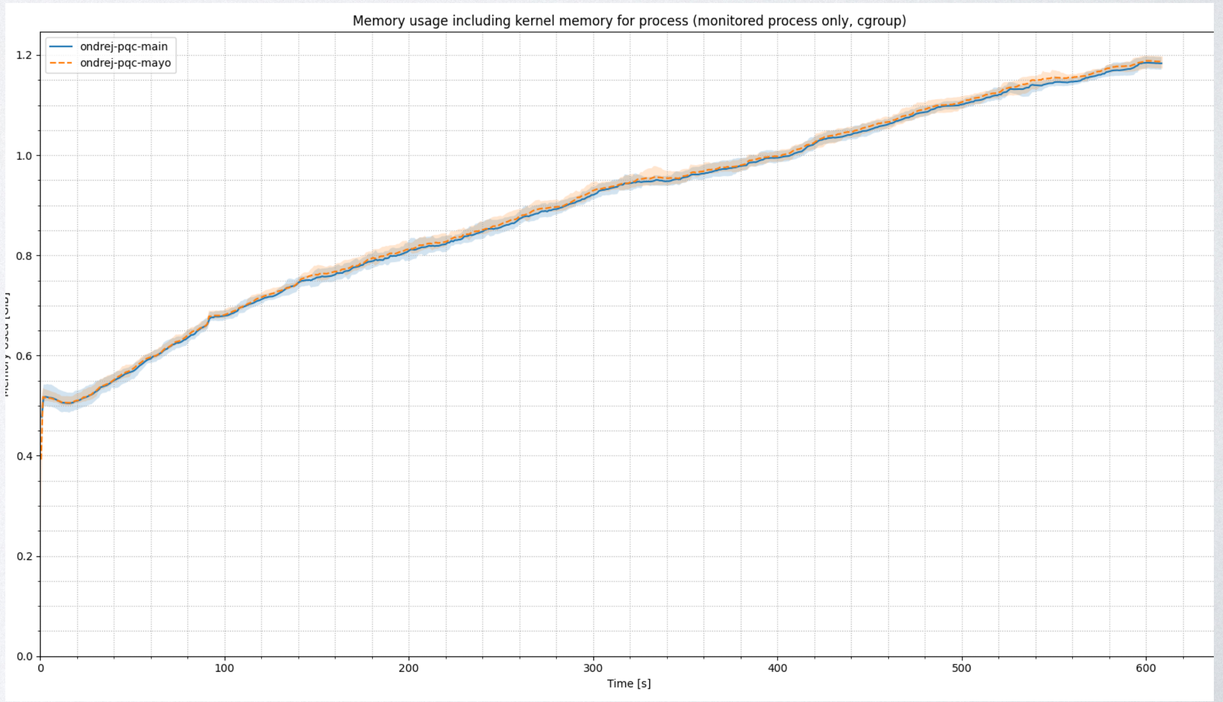

CPUs usage (monitored process only, cgroup)
ondrej-pqc-main
ondrej-pqc-sqisign
Time [s]


Memory usage including kernel memory for process (monitored process only, cgroup)
ondrej-pqc-main
ondrej-pqc-sqisign
Time [s]

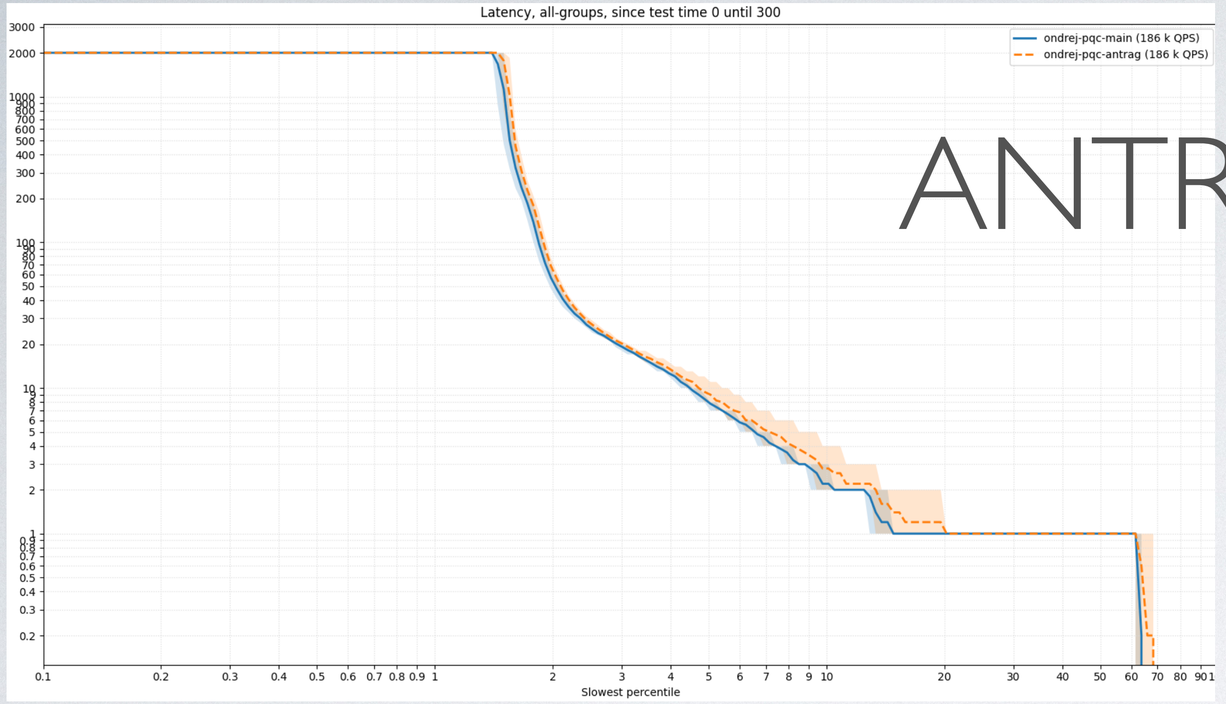

Latency, all-groups, since test time 0 until 300
ondrej-pqc-main (186 k QPS)
ondrej-pqc-mayo (186 k QPS)
Slowest percentile

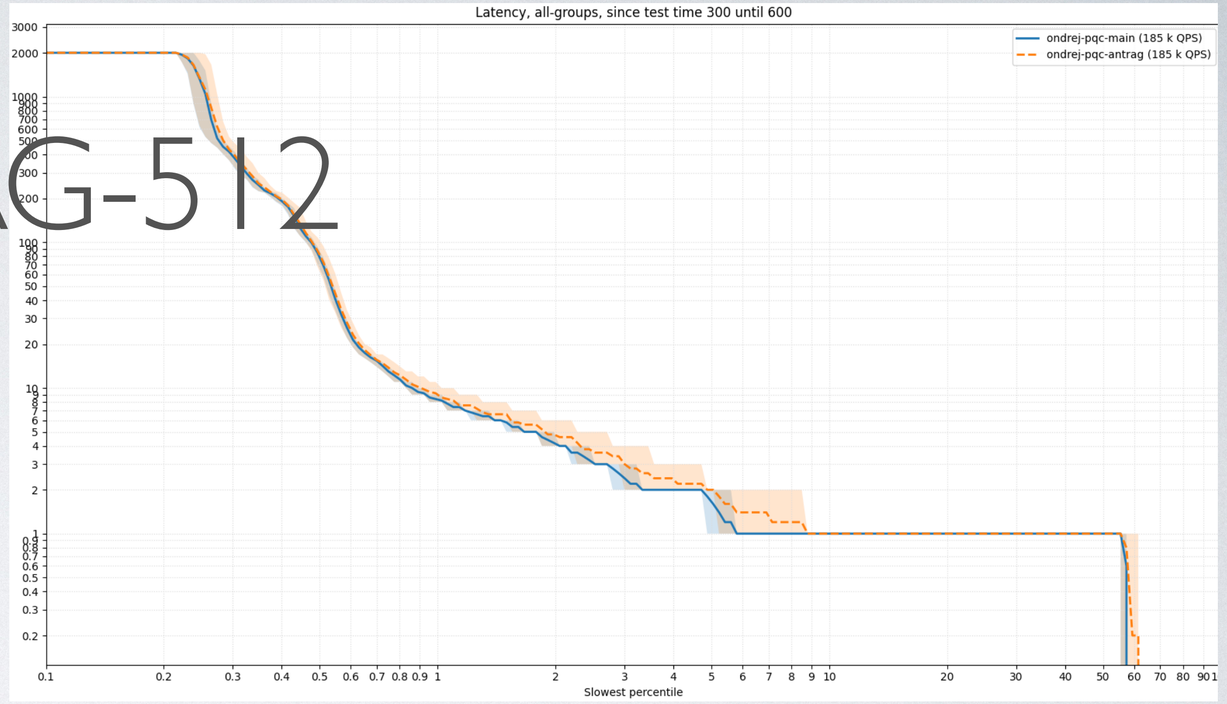

Latency, all-groups, since test time 300 until 600
ondrej-pqc-main (185 k QPS)
ondrej-pqc-mayo (185 k QPS)
Slowest percentile

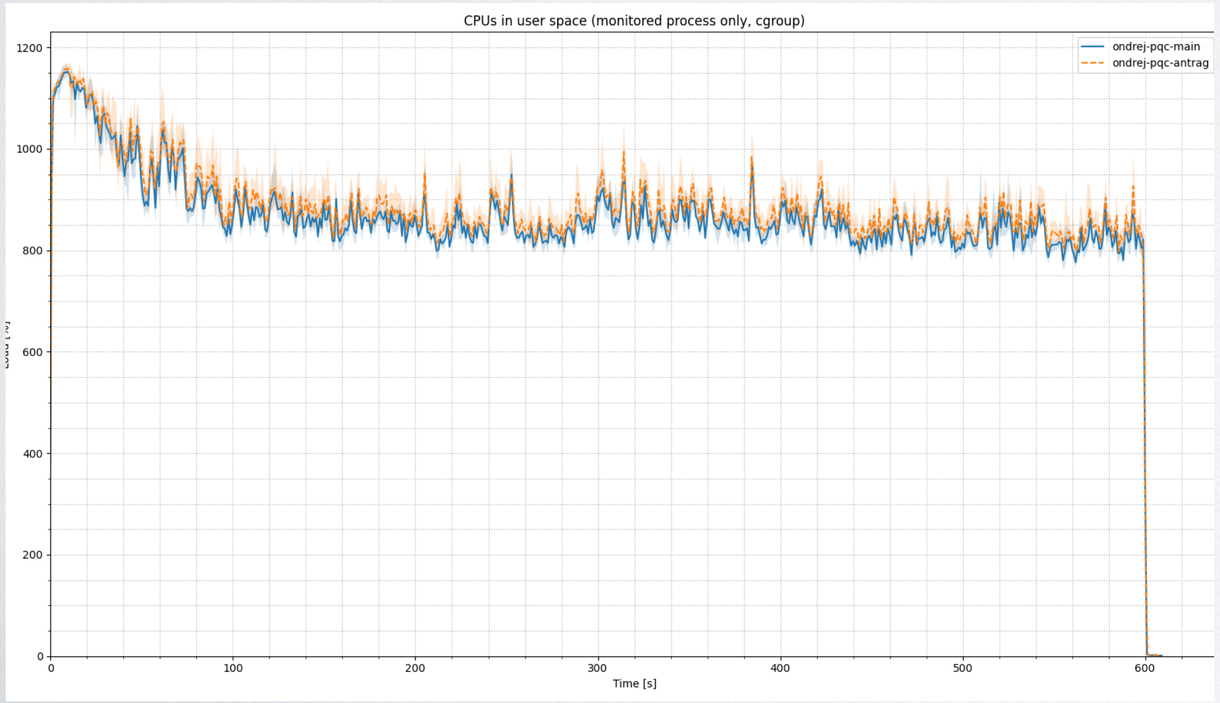

CPUs usage (monitored process only, cgroup)
ondrej-pqc-main
ondrej-pqc-mayo
Time [s]

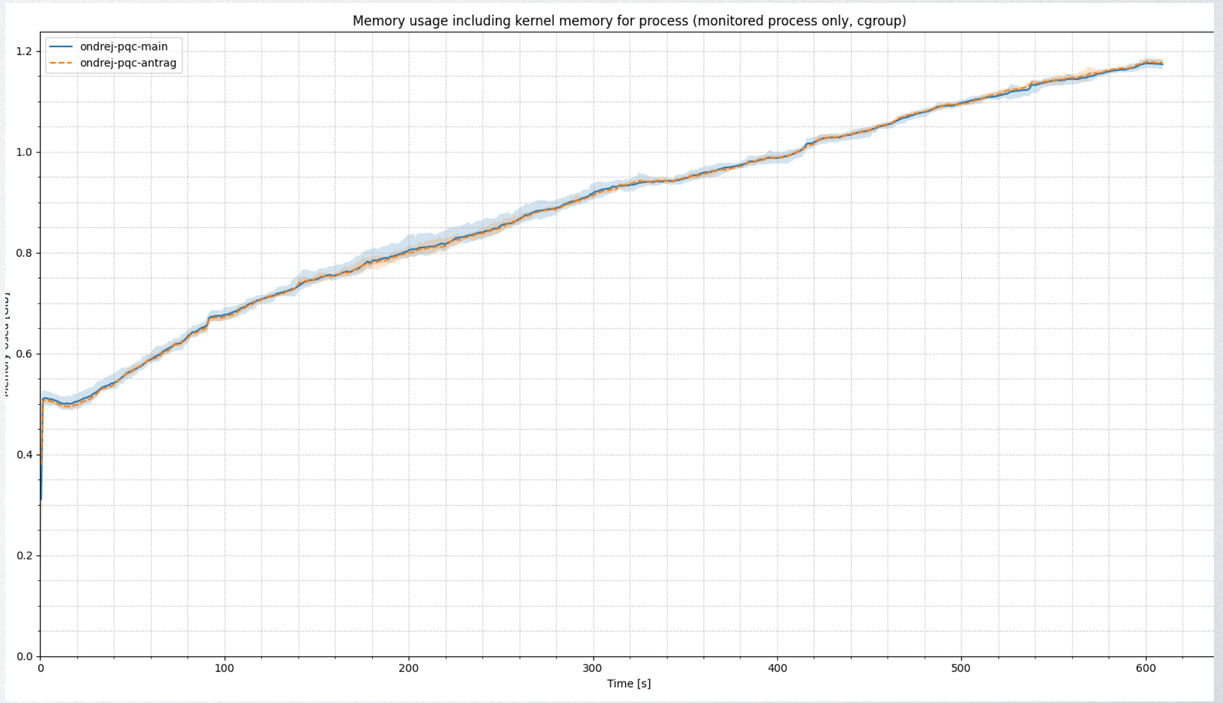

Memory usage including kernel memory for process (monitored process only, cgroup)
ondrej-pqc-main
ondrej-pqc-mayo
Time [s]

ANTRAG-512


Latency, all-groups, since test time 0 until 300
ondrej-pqc-main (186 k QPS)
ondrej-pqc-antrag (186 k QPS)
Slowest percentile


Latency, all-groups, since test time 300 until 600
ondrej-pqc-main (185 k QPS)
ondrej-pqc-antrag (185 k QPS)
Slowest percentile


CPUs in user space (monitored process only, cgroup)
ondrej-pqc-main
ondrej-pqc-antrag
Time [s]


Memory usage including kernel memory for process (monitored process only, cgroup)
ondrej-pqc-main
ondrej-pqc-antrag
Time [s]

ANTRAG-512

- Authors talk to me! Yay! Hooray!
- The multi-threading is broken – signing is much slower when multithreaded, feels like global-lock on the secret key
- The signing API is slightly confusing, when encoding the signature fails, you need to retry the signing with a different salt (internally), so it is a basically loop over `encode_sig()` result
- It uses own `rng_bytes()` for entropy – but it needs manual initialization

FUTURE WORK

- Test different levels of DNS hierarchy
- More (different) algorithms
- Look at the pseudo-random sub-domain patterns
- Forced re-keying (small TTLs, etc)
- Implement NSEC3 aggressive caching (does it help?)
- Use SystemTap/DTrace probes to measure the crypto operations
- Use the optimized (AVX2) implementations

THANK YOU