Alle nieuws

How to get un-stuck by Architecting for Resilience

4 weeks ago

This blog is written by the participants of the PQC Benchmarking project, from TNO, Achmea, Belastingdienst, ABN Amro and ING. The test case described in this blog was carried out by Belastingdienst.

Introduction

As developments in quantum technology push forward, the resulting reality of weakened cryptographic strength puts us to the test. For decades we could rely on stable defaults like Diffie Hellman, RSA and elliptic curve cryptography (ECC). Now, we must revise how we achieve digital trust goals like confidentiality, integrity and availability, by adopting what is called post quantum cryptography (PQC).

The international community has been developing and standardizing the new post quantum algorithms. IT departments and businesses must prepare and start using them in their infrastructures and landscapes. Importantly, vendor readiness is not the only missing part of the puzzle. It is also a matter of knowledge and insights: these new algorithms are less familiar to IT practictioners and little is known about their performance in production systems.

PQC migration features a leading role for IT and security architects. They will need to consider available alternatives and their feasibility for each use case. This difficult task requires expertise that could greatly benefit from real world experiences, like benchmarks conducted in real-world implementation scenarios.

In our project, we started a benchmarking test to do just that. We share our most valuable lessons learned in this blog. You will learn about how we encountered various pitfalls that are typical for migration. The good news is: we found ways to pivot out of them!

The Quest for insights

To start off the benchmarking journey, we had to pick a use case that would produce practical knowledge and insights on the usage of PQC algorithms and that would be executable within the project's time constraints. We settled on migrating the connection setup for a message queuing (MQ) application in which clients use PQC certificates instead of classical certificates. What would the performance impact be?

One of the main considerations was that message queuing applications are a common component in IT architectures that require high performance. There was also a practical side: we found that message queuing is closely tied to the daily work of many teams, which meant that essential resources for a successful benchmarking project, such as domain knowledge, tooling, and a test environment were likely already available.

After settling on a use case, we had to decide which PQC algorithms and which performance characteristics we were going to test for.

It would be interesting to see the throughput effect of using PQC-enabled certificates for secure connections, since that is usually the bottleneck. We decided to create certificates with various PQC algorithms for key exchange and digital signatures, including hybrid combinations of classical and post-quantum algorithms, and to compare the results to the currently deployed quantum-vulnerable version of the message queuing application. Other characteristics we decided to include were CPU usage, network bandwidth and memory usage, since they can provide insights for other organisations and applications.

Test scenarios

Since there are a large number of candidate algorithms within the field of post-quantum cryptography, we had to select which post-quantum algorithms we would incorporate in the migration and benchmark.

We chose the four NIST standards (ML-KEM, ML-DSA, SLH-DSA and Falcon) and FrodoKEM and McEliece to test. Even though FrodoKEM and McEliece have not been standardized, they are often recommended by EU member states. This selection should provide a good overview of the most popular algorithms, with varying performance and conservativeness in the security assumptions.

Various organizations recommend the use of hybrid schemes, where classical and post-quantum algorithms are combined. If one of the algorithms fails, then security still holds, which can provide a fail-safe in case one of the newer algorithms is not as secure as we thought. We incorporated several hybrid schemes in our benchmarks as well.

To understand the impact of migrating KEMs and digital signatures schemes, we decided to benchmark the current classical solutions to have a baseline, to benchmark post-quantum KEMs in isolation (keeping the digital signature classical), post-quantum digital signatures in isolation (with classical KEMs), and finally to benchmark the combination of post-quantum KEMs and post-quantum signature schemes. An additional benefit of this separation is that companies who want to migrate KEMs first to mitigate the harvest-now-decrypt-later risk will get an impression of the initial performance degradation. Apart from insight into performance impact, we were also interested to learn about the difficulty of incorporating all these algorithms (and their hybrid combinations) in actual products.

The following table shows the test scenarios we defined for the benchmark:

PQC Algorithms:

Key exchange (KEM):
- A. ML-KEM,
- B. FrodoKEM,
- C. McEliece

Digital signature (DSA):
- D. ML-DSA
- E. Falcon
- F. SLH-DSA

Test cases:

	KEX	DSA	Algorithms to test
0.	Current (FFDH)	Current (RSA)	First baseline
1.	ECC	ECC	Second baseline
2.	PQC	ECC	A,B,C
3.	ECC	PQC	D,E,F
4.	PQC	PQC	All combinations of A-C with D-F
5.	(ECC+PQC)	ECC	A,B,C
6.	ECC	(ECC+PQC)	D,E,F
7.	(ECC+PQC)	(ECC+PQC)	All combinations of A-C with D-F

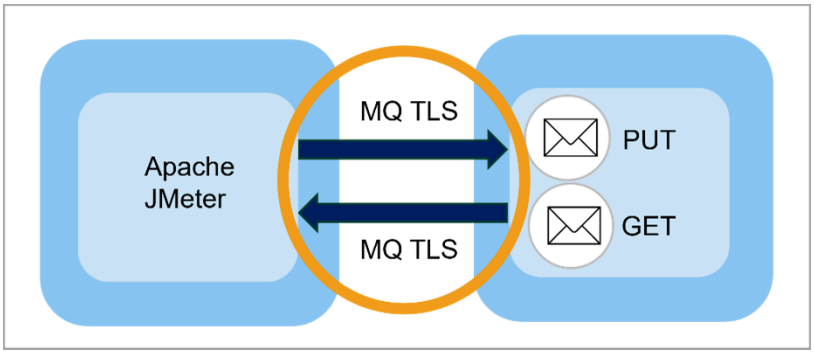
In this test only 22 of the 32 combinations were supported. For the technical specifics there is an in-depth addendum to this blog. You can read it here: How to get un-stuck by Architecting for Resilience 2 | News

---

Architecting for resilience

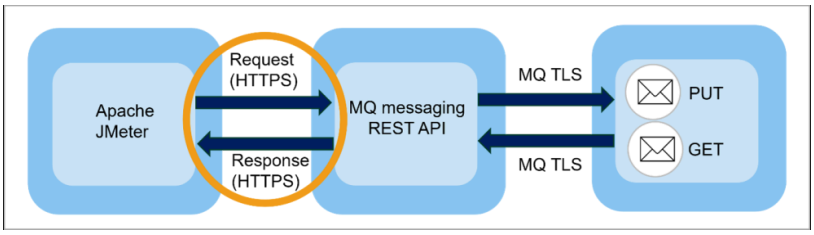
After the test scenarios had been determined, we could get to work to migrate the message queuing setup. As part of their daily work, our messaging tech team uses Apache JMeter for performance testing. They use certificates for the message queuing product and also for TLS connections between the clients and the servers. Not only does this mean that some tooling was already in place, it also meant that they had hands-on certificate expertise. That was a great way to start off.

The figure below shows our initial scenario. The orange part is where we intended to configure PQC:



However, we ran into availability impediments and as a result no vendor certificates could be used. Another major issue was the fact that the MQ product did not support PQC yet, so we could not launch our test scenarios using that product.

With some quick thinking, we adjusted our plan and decided to try using an OpenSSL generated certificate and a REST API for the message queuing connection:



But that didn't work either. Although PQC support was available in OpenSSL, the REST API wasn't ready. Another day, another setback. Even though this was bad news for us, it did confirm our suspicions that migrating our application was no trivial task and that experience at this stage gives insight into practical challenges.

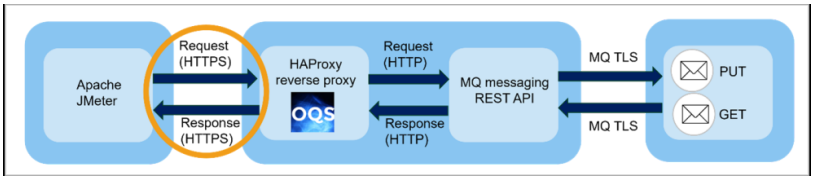
Time to take a step back and switch perspective.

Cryptography is used as a *measure* to limit some *risk*. If the algorithm is vulnerable, that measure is now ineffective, and the risk resurfaces. You then have to look at how you limit that risk again, which can be done multiple ways. For instance by taking another cryptographic measure, by reducing sensitivity of the data, by limiting access to an application, with a physical security measure or by using a different security design.

We had to think out of the box. As Anita Wehmann from the Dutch quantum safe cryptography support program (QvC) puts it:

"You are not stuck if you cannot migrate!"

That made us pivot the architectural design to the following final setup:



We decoupled the architecture to add the flexibility we needed. We inserted a reverse proxy to bridge between the components that were PQC-ready and those that were not. Specifically, we deployed an HAProxy server with extensions from the OpenQuantumSafe framework and OpenSSL. For the time being we accept SSL offloading until the vendor supports PQC. This makes the number of quantum safe connections in the scenario smaller than we preferred (only the orange part).

However, there are advantages to this approach. We are still able to do a baseline benchmark, using most of the algorithmic combinations, we can put the test tooling in place and gain experience. At a later stage, when other components will support PQC, we can revert back to the previous architecture and rerun the tests.

Insights

In the end, we were able to execute all test scenarios we had defined. The tooling delivered the information on our main metric: message throughput per second. We can use this data to analyze the respective configurations. Additionally, we had a glance at CPU time, networking bandwidth and checked for signs of memory problems and data packet loss. In this section we will walk you through our high level conclusions. A more detailed analysis can be found in the in-depth addendum to this blog.

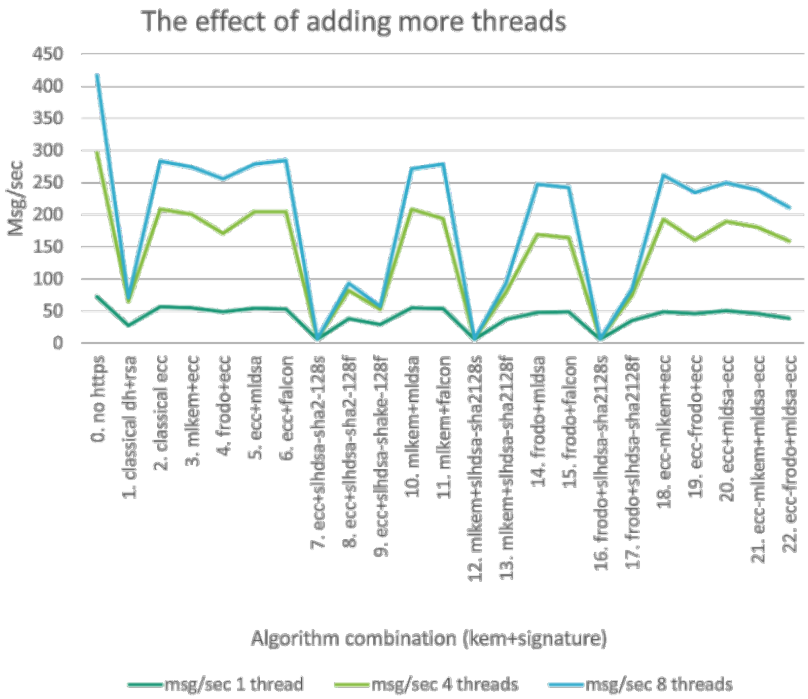
Baseline references

Regarding the baseline references, the data shows that classical ECC is significantly faster than classical RSA and Diffie Hellman. This means that ECC serves as a good reference point for what classical cryptography has to offer at the moment.

Multithreading

Our test scenario was executed multiple times, each with a different number of threads. Scaling threads seems to give linear performance increase for almost all KEM and DSA configurations. This means that CPU usage is generally not the bottleneck. During the tests, we observed no packet loss, which supports this conclusion. A likely explanation could be that network delays are large enough so that other threads can be run before the necessary input is received.

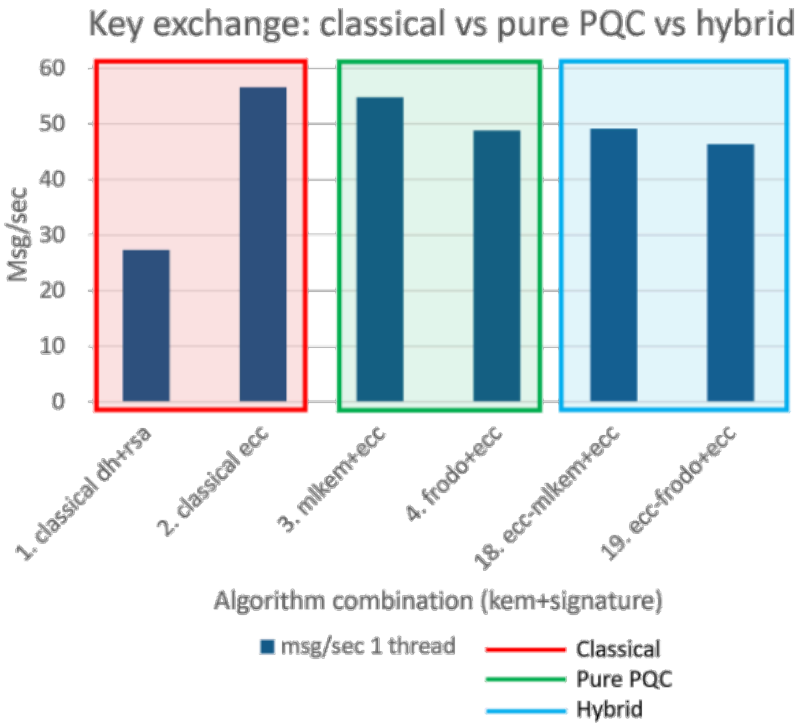
A significant finding is that the size-optimized digital signature algorithm SLH-DSA really puts resources to the test. Increasing thread count for the SLH-DSA test cases resulted in maximum CPU usage and absence of linear performance increase. These seem to be CPU intensive operations (test cases 7, 12 and 16 in the graph).



Key exchange

Comparing pure PQC ML-KEM to classical ECC, the performance results are quite similar. Surprisingly, the results for pure PQC FrodoKEM are not that far off. FrodoKEM has security benefits, because its security assumptions are more conservative than those of ML-KEM. Given the minor performance difference, it could be a relevant option for organizations that want to stay on the cautious side.

The performance loss for using hybrid ML-KEM instead of pure ML-KEM seems to be a lot less than expected, around 10 percent.

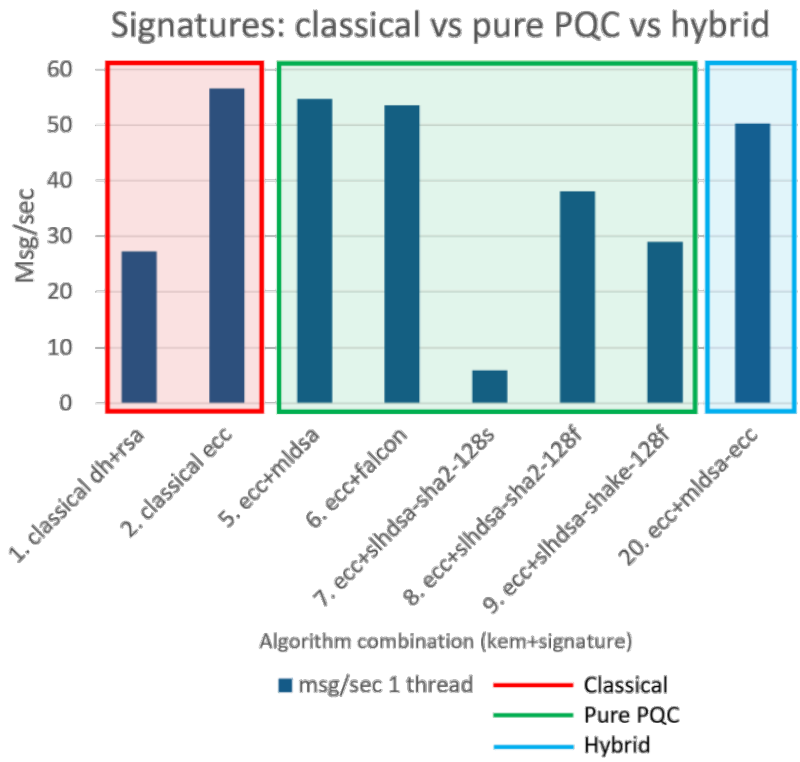


Digital signatures

Looking at the combination of classical KEMs with various pure PQC digital signatures, we can see quite a few performance differences.

When we compare ML-DSA and Falcon signatures in a pure PQC configuration to classical ECC signatures, their performance is quite similar. However, for SLH-DSA signatures, it's a whole different story. They perform significantly worse, especially size-optimized SLH-DSA (slhdsa-sha2-128s).

For hybrid configurations we expected some loss of performance, due to the verification overhead, but our data didn't support that. The results imply that hybrid signatures come with a limited performance cost. That's good news, as hybrid configurations are often recommended in the current phase of the PQC migration process.



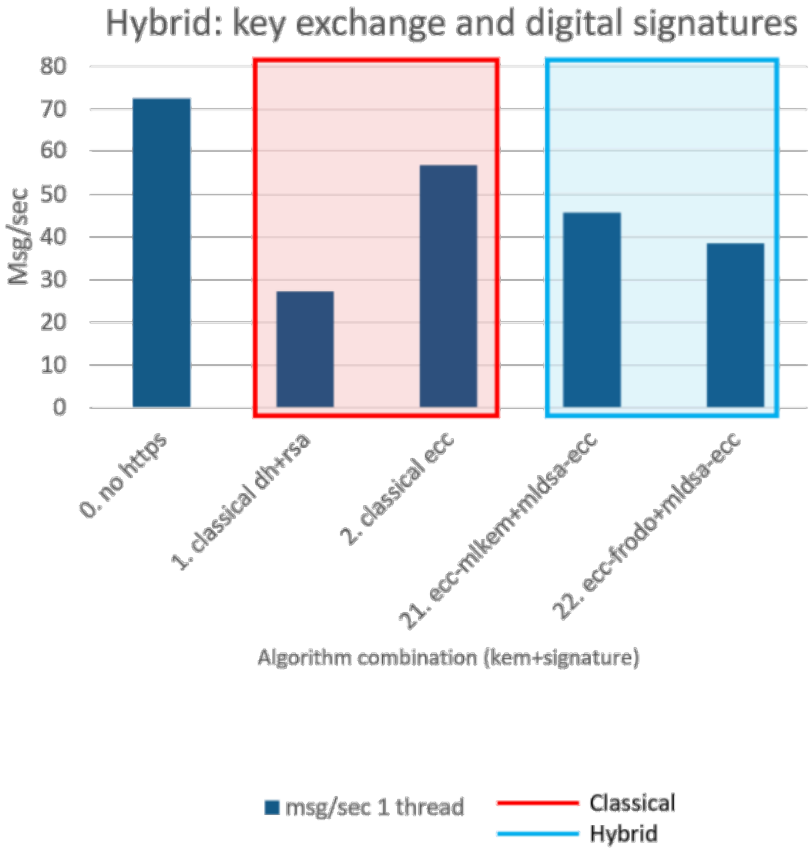
Pure PQC key exchange and digital signatures

From the previous results, we would expect that the combination of pure PQC for both KEMs and DSAs would not give a big performance loss. That is, when we leave out SLH-DSA. We can see that confirmed in our results for the ML-KEM and ML-DSA combination. A little more performance loss is shown for the combinations using FrodoKEM. Note that FrodoKEM can be more problematic if message size matters, which was apparently not the case in our test setup. See the addendum for the details on size.

KEM	DSA	Msg/sec (1 thread)
First Baseline (DH)	First Baseline (RSA)	27,3
Second Baseline (ECC)	Second Baseline (ECC)	56,6
ML-KEM	ML-DSA	55,3
ML-KEM	Falcon	53,8
ML-KEM	SLH-DSA-SHA2-128s	6
ML-KEM	SLH-DSA-SHA2-128f	37,3
FrodoKEM	ML-DSA	47,8
FrodoKEM	Falcon	48,7
FrodoKEM	SLH-DSA-SHA2-128s	5,9
FrodoKEM	SLH-DSA-SHA2-128f	35,5

Hybrid key exchange and digital signatures

In the previous paragraph we looked at pure PQC test results. Which conclusions can we draw from combining the fastest classical algorithms with the fastest post-quantum algorithms? Using them in hybrid configuration for both key exchange and digital signatures, we measured about a 20% performance decrease in message throughput. Adding more threads helped bring that down to about 15%.



Conclusion

The quest for insights on what it means to use PQC in real life situations was an interesting journey. Migrating away from quantum-vulnerable cryptography and transitioning to quantum-safe alternatives can bring many challenges. One of them is interoperability between systems that are migration-ready and components that aren't.

Do not get stuck waiting

It's easy to get stuck in a situation where you have to wait until every part is ready to migrate, contributing to what is sometimes called 'crypto procrastination'. However, there is urgency to move forward with the PQC migration, as quantum computers that can break current cryptography are rapidly getting closer. We found that alternative architectural options can be decisive in regaining momentum within the transition.

Decoupling is an alternative to protect systems that do not yet support PQC

In this PQC Benchmarking project we inserted a decoupling component in the architecture. Even though our application was not ready, by using a PQC reverse proxy on the same server we could protect the sensitive data on the network, and limit the risk. As a result, we were also able to develop, test and run a benchmarking script for almost all cryptographic test cases we defined. Next to solving the problem of waiting for a vendor, it also makes it possible to quickly update the cryptographic algorithms used. This type of flexibility, often referred to as crypto-agility, is very desirable for cryptographic systems.

The performance penalty varied, but generally the impact was less than expected

We focused on message throughput as a performance metric in a message queuing scenario. How do the PQC options we have for key exchange and digital signatures compare to the current DH/RSA and elliptic curve algorithms? From our scoped benchmark, we can conclude that the ML-KEM and FrodoKEM algorithms seem to perform quite similarly to classical ECC. Even the hybrid combination of these KEMs and ECC does not seem to come at a very high performance loss.

Experiments help to get experience and will give clarity on your options

SLH-DSA has the advantage that it has very conservative security assumptions. Cryptographers understand its security so well that they trust it as much as the security of ECC and DH/RSA. As a result, it would not need to be combined with classical solutions in a hybrid combination, which could be beneficial for performance. However, our results show that SLH-DSA performs quite poorly, even compared to the hybrid configurations of classical and post-quantum DSAs, which makes it less practical.

All in all, the journey felt like a rollercoaster at times, but we are glad to have learned more about performance trade-offs between the post-quantum algorithms and architectural solutions to vendor dependencies. We hope that we have inspired you to run your own benchmarks within real world IT infrastructures, so we can smoothly transition to cryptographic resilience together.

Do you want to know the technical details? See the addendum to this blog here.

Keep a look out for the other blogs in this series! In this series of four we share both organizational and technical results, from four different perspectives: Management, Vendor Management, Architects and Developers. Read the first blogs about manager here and the second one for developers here.

Disclaimer: first image was created with CoPilot.

Deel deze pagina

Alleen door samenwerking kunnen we de beste resultaten behalen in de strijd tegen cybercriminaliteit