Quantum-resistant digital signatures schemes for low-power IoT

1st Hannes Hattenbach
Computational Science
Freie Universität
Berlin, DE
hannes.hattenbach@fu-berlin.de

***Abstract*—**
***Index Terms*—Internet of Things, Quantum Resistance, Secure Signatures, Power Constraint Devices**

I. Introduction

The quantum revolution is coming. With quantum computers[1] on the way to get more and more functional, people are fearing a loss of their security and privacy. Or as [1] puts it, "principles of data integrity, message authentication, and nonrepudiation, are going to have profound aftermath on sensory data in terms of security and privacy." That is because there are algorithms based on Shors algorithm that can forge signatures and decrypt encrypted messages whos security is based on discrete logarithms, including elliptic curves or prime factorization, like our most common schemes ECDSA and RSA respectively are. The quantum computer only needs access to the public keys of these asymmetric schemes. The expenditure to forge a signature[2] with classic[3] computers rises exponentially with increased key length, therefor being essentially unbreakable by classic computers. A sufficient quantum computer on the other hand can derive a private key from a public key in polynomial time, therefor rendering these schemes broken.

That is why there are currently schemes under standardization [2] that are based on other hard problems (not number theory) like so called lattice problems that cannot be that easily forged by quantum computers to save our privacy and security.

One of the use cases not directly coming to mind for the end user, but being as important non the less is signing sensitive sensor data in the Internet of Things (IoT). Another problem coming up in the IoT compared to end-user-devices like Laptops and Smartphones though is the severe resource constraint-ness. The IoT consist of low power devices with very few storage and computing power.

In this paper i am going to evaluate existing signature schemes and their usage possibilities for the IoT regarding their performance metrics.

Therefor i am going to give a small introduction and background to quantum computing, being a little more detailed about their ability to break current encryption and signature standards. In the next section i will give an overview over current candidates for Quantum Resistant (QR) Algorithms and giving performance metrics for those. The following chapter will then focus on signature schemes in the IoT, starting with additional performance metrics relevant in the IoT. With a little more details about two failed signature schemes to highlight potential pitfalls. And finally focussing on the best signature contender for the IoT so far: FALCON.

[1] compare section III-A

[2] that is considered secure under normal circumstances

[3] we refer to classic if something is not directly leveraging entanglement or superposition

II. Background

A. Cryptography

Loosely speaking the main topic of cryptography can be divided into three groups. The first of these groups is about one way functions, that shall not, as the name implies, be efficiently reversible. If we create a smaller value of constant length from a bigger set of possibly variable length, we commonly refer to that as *hashing*. Cryptographic hashing is important for a variety of different applications like storing and matching passwords without the ability to infer any knowledge about that password. Hashing itself can be used for the next pillar of cryptography: signatures. Signature schemes are used to proof integrity or authenticity of any data. A signature scheme consists of two parts, signing and verifying. The last group is encryption, which ensures privacy/confidentiality of any data, s.t. only the right entities can decrypt this data. These schemes consist of the two parts encryption and decryption. Additionally to those parts for signatures as well as encryption there needs to be process of key-generation. We also differentiate between symmetric and asymmetric schemes. The first one has a different private and public key while the latter uses the same for de- and encryption. More details about which of those schemes will be more or less endangered by quantum computing are in section III-A and III-B.
In general we denote a signature scheme as the group of three algorithms {GEN, SIGN, VER} and a encryption scheme as {GEN, ENC, DEC}.

B. Internet of Things

The IoT consists of devices of all sorts, having in common, that they communicate with each other and the environment rather than directly with humans. Those devices range from automatic lights and smart home devices to tiny interconnected

TABLE I
IETF IoT CLASSES

Class	RAM	Flash
C0	<< 10 KiB	<< 100 KiB
C1	10 KiB	100 KiB
C2	50 KiB	250 KiB

sensors in automatic fabrication. A common characteristic though is, that most of these devices have limited processing power, flash storage and random access memory (RAM). A popular example for hobbyist IoT devices is the ESP32 from Espressif Microsystems. They offer multiple Modules with up to 240Mhz Clock on the 32 IC, up to 16MiB Flash Storage and 320KiB RAM. Which is more than other comparable devices but way less then a lower spec modern smartphone, with 10 times the frequency, 4GB of RAM and 64GB of storage.

Since the IoT consists of very different types of constrained nodes the IETF introduced different classes on which to classify IoT nodes, those can be seen in table II-B

III. QUANTUM RESISTANT SECURITY

A. Quantum Computing

In contrast to classical computers, where information is processed in discrete states, a quantum computer leverages quantum mechanics to operate on so-called qubits - quantum objects that can be in superposition or entangled with each other. Opening a new kind of computing. One of the implications of that is, that it is now possible to factor large numbers in polynomial time using an algorithm developed by Shor [3]. This algorithm uses a so-called Quantum-Fourier-Transform (QFT) to (probabilistically) get the frequencies of which a given function output occurs. That can be used together with euclids algorithm of finding the greatest common devisor to derive the prime factors. Prior to to quantum computers this was considered a hard problem that could only be computed in exponential time and was therefor considered practically impossible and was used as the basis-problem for RSA encryption. Similar to that other common schemes like ECDSA can also be broken be slightly modified versions of Shors Algorithm.

B. QR Algorithms

The two main algorithms with practical use cases that have a great speed-up compared to classical solutions, are the already introduced algorithm by Shor and an algorithm by Grover that can essentially reverses one-way functions by creating a superposition over all possible inputs, flipping all inputs with the wanted output (without knowing the inputs) and then flipping this state about its mean and repeating this process a lot of times [4]. While Shors algorithm provides exponential speed-up, Grovers algorithm only provides quadratic speed-up. It was also shown, that something similar to grovers algorithm but with exponential speedup is impossible [5]. Which implies that Hashing as well as symmetric cryptography stays relatively secure. The quadratic speedup provided by quantum computers can easily be mitigated by doubling the key length.

TABLE II
QR SECURITY CLASSES AND THEIR TRADITIONAL COUNTERPARTS AS CLASSIFIED BY THE NIST

Class	security compareable to
1	AES-128
2	SHA256
3	AES-192
4	SHA384
5	AES-256

On the other hand though, classical asymmetric cryptography is endangered by shors algorithm and quantum computers.

But not all asymmetric cryptography schemes are equally affected. There are different proposals, both for QR encryption and for QR signature schemes. They all do have in common though, that their security is not absolutely mathematically proven, but based upon assumptions. We therefor need to consider a few measures that make schemes more or less secure.

1) Performance Metrics: Some performance metrics exist in QR schemes as well as in classic schemes.

Key length and key exchange message length [6] are the more obvious ones. The computing time also comes to mind as a performance metric. Here you need to differentiate between key generation, which is less important, since it should only occur rarely, and signing as well as signature verification [4].

Primarily in signatures another metric arises: how often can a private key be used before it needs to be switched out for another one, because the signature leaked information of the key. This is not particularly relevant in most cases, as methods can be used to create long term procedures from short term procedures (those where a key can rarely, if ever, be recycled). But it is relevant in the case of the IoT, since those methods require extra memory which is sparse in IoT-devices. Additionally they tend to make the signatures themselves longer, which also is not preferable in the IoT. [6]

Additionally to more traditional performance metrics we somehow need to measure the security of given schemes against an attack by a quantum computer. Sadly there is currently no standard benchmark to measure quantum resistance [7], nevertheless the NIST created a standard that describes how secure a scheme is against a quantum computer by classifying it within 5 classes that can be determined with grovers algorithm [8], [9]. Those classes can be seen in table III-B1

2) Encryption: QR encryption schemes can be based upon a multitude of different mathematical problems thought to be hard even for quantum computers.

Sadly, being thought of as secure mostly is not based upon actual rigorous proof but assumptions. Therefor one problem that was used as a asymmetric encryption basis, the knapsack problem, was broken soon after its introduction by so-called approximate lattice reduction attacks [6].

Later iterations which include “conjugacy search problem and related problems in braid groups, and the problem of

[4]as well as its counterparts de- and encryption

solving multivariate systems of polynomials in finite fields" [6] have been under active research with the latter being broken after standardization and implementation [6].

Nevertheless there is an implementation of a multivariate-based scheme, called Rainbow, that is also currently a contender for standardization. But as an encryption scheme its not very suitable since the process of decrypting in multivariate based schemes requires some guessing work [7] which is essentially bad in IoT enviroments. An additional problem that would make rainbow unsuitable for IoT use-cases is its big 22kB public key. While private keys can rather easily be shrunk in key-generation through help of a pseudo-random-generator, thats generally not the case for large public keys.

On the other hand we have a problem that is not yet very well researched and also not much in use, but has one implementation called SIKE. This problem is based upon supersingular elliptic Curves, which are itself a modification of elliptic curve problems that should make it quantum resistant. But since this topic isnt well-studied yet we are mostly left with Schmes based upon the following two thought-to-be quantum-hard problems.

The first one is so called code-based cryptography. Here the decoder has to correct errors of data that has been seemingly randomly shuffled, but only those with access to the private key can easily 'unshuffle' the data to then use special error correction codes. The most researched one is called McEliece and even has quite fast ($100\mu s$) and secure implementations. The main problem is, that the 'shuffling' is realized through $k * n$ matrices that are generally big (millions of bits) and therefor unfeasible for constrained IoT devices.

The second one will be discussed in greater detail in section III-B3, since it is also used as one of the main problems for signature schemes. Those schemes are called lattice based and also have some implementation with the most famous for encryption being NTRUEncrypt.

3) Signatures: The other pillar of cryptography, signature schemes, is what we will focus on in greater detail. As well as in encryption schemes we can differentiate between different underlying mathematical problems. Those are pretty much the same as in encryption schemes: Hash based, Lattice based, Multivariate polynomial based, Code based, Super-singular isogeny based. [1]

Rainbow is the only implementation of a QR signature that is a current contender for standardization that is neither lattice nor hash based. And as already mentioned in the previous section it is multivariate based.

Since this sparsity of alternatives we we also focus on hash and lattice based signatures in this paper.

Hash based signatures have their security based upon the hardness of reversing Hashes or one-way functions. The most easy one is the Lamport one time signature (OTS). [6] That signature has essentially two private keys for every bit in the message digest. Let $n \in \mathbb{N}$ be the bit-length of the digest, then the secret key would be:

$$k_{\text{priv}} = (S_{0,0}, S_{0,1})||(S_{1,0}, S_{1,1})|| \dots ||(S_{n,0}, S_{n,1})$$

The advantage of those schemes is, that the private keys do not have to have any special characteristic that could be taken advantage of by a quantum computer to break anything. They do have to be high entropy though, to not be easily forgeable with even a classic computer. These secrets are then hashed (with a one-way function h) and published as the public key

$$k_{\text{pub}} = (h(S_{0,0}), h(S_{0,1}))||(h(S_{1,0}), h(S_{1,1}))|| \dots ||(h(S_{n,0}), h(S_{n,1}))$$

When a message is signed the signer just publishes the secret corresponding to every bit of the digest (Sk, b with b being the bit-value in the k-th position ob the digest) s.t. everyone can hash that secret and see that this private keys are indeed the ones corresponding to the public key and the correct bit-value of the digest. Signing as well as verifying are therefor rather easy operations with one disadvantage: the keys and the signature are super big. But there are some rather easy improvements for this problem e.g. one could only sign the zeros, therefor reducing key sizes by a factor of 2 as well as average signature sizes. To mitigate an attack that can flip digest-zeros to ones a checksum is added (that can only be decreased by flipping a one to zero, which is impossible if you do not know the pre-image (private key) of that location). Another improvement often wrongly [5] cited as the successor to the Merkle OTS is the Winternitz Scheme (WOTS), which builds upon the same idea but uses a different (greater) basis b, which inturn makes the signing and verifying more computational expensive by needing to apply hashes b times. The great advantage though is that the keys and signatures also decrease by a factor of b. This can be a great advantage for IoT applications, since time is not as valuable as storage. Therefor WOTS is actually used in practice, for example as a signature on the IOTA distributed ledger. [11]

A directly visible disadvantage of those schemes (as the name implies) is that they can trivially only be used one time, since most of the private key gets public with the signature. A trivial countermeasure would be to append the next public keys to the message and sign them as well, but thats not a good idea in most use cases, since you might as well just use symmetric cryptography which is also considered as quantum resistant as hashes. Another idea would be to just publish a whole lot of private keys that can than be used one by one. But thats not a super brilliant idea since signer as well as verifier need to store all these keys which is specially infeasible in IoT scenarios (that have very constrained storage). Schemes that can be used multiple times are smartly called multiple time signatures (MTS).

A smarter approach to simply publishing n public keys and storing n private keys was proposed by merkle [10]. His approach uses so called merkle hash trees to make it possible to have a very small public key that can still verify n signatures

[5]the merkle OTS has two parts, one that is similar to the Lamport scheme (which was then improved by Winternitz) and one that uses Merkle Hash Trees, which most of the literature refers to as the Merkle Signature, but is not a incessor of the winternitz scheme which does not use Merkle Trees [10]

on the tradeoff that every signature now increases by a factor of $\log(n)$. The idea is as follows:

Algorithm 1 GEN

1) generate $n = 2^m$ random values, those are the private keys.
2) for every private key k^i_{priv} generate a one-time public key $k^i_{\text{pub}} = h(k^i_{\text{priv}})$ (until here it is similar to a trivial MTS)
3) hash every two 'neighboring' keys k^i_{pub}, k^j_{pub} together in pairs to generate $n/2$ new hashes $h_{ij} = h(k^i_{\text{pub}}, k^j_{\text{pub}})$
4) hash those in pairs for the next iteration and repeat until the hash-tree is complete and we only have one root hash denoted as k_{pub}
5) publish k_{pub} that can now be used to verify n signatures

Algorithm 2 SIGN

1) input message digest M_i
2) sign as described for Lamport or Winternitz schemes (or other OTS schemes that generate the public key by hashing the private key): $S_i = \text{Sign}(M_i)$
3) publish S_i together with all hashes h needed to iteratively generate the root hash k_{pub}. These are m hashes.

Algorithm 3 VER

1) input signature $(S_i, [h_j, h_{i+2,j+2}, \ldots, h_{i-k}])$ and digest M_i and already known multiple use public key $k_{\text{pub}} = h_{0-(n-1)}$
2) hash S_i to generate k^i_{pub}
3) hash $k^i_{\text{pub}} = h_i$ together with h_j to generate h_{ij}
4) hash the value from previous step together with the next hash given by the signature
5) repeat step 4) until the root hash k_{pub} should be found (thats m steps in total) return True if they are equal and False otherwise

This is already very useful for IoT actors that only need to verify, less so for sensors that still need to store all n private keys. The computational cost is higher, caused by calculating all those hashes but thats commonly worth the tradeoff.

On the other side (the signer) we need to store all n private keys and calculate m hashes every time we want to sign anything. The second step can be skipped by also storing the hashes instead of calculating them, which increases the storage needed by a factor of 2. But thats infeasible for most storage constrained devices. Therefor an additional tweak was applied to this algorithm: Instead of randomly generating each private key and storing it, we use a Pseudo-Random-Generator (PRG) together with a seed and a counter to be able to generate every private key on the fly. We can then iterativly generate our merkle tree and drop every nodes we already used to calculate the next parent hash without exceeding our RAM to generate the root hash to publish as the public multi time key. For signing we can then create our private key again with the help of the PRG and again calculate all needed hashes iterativly the same way. But thats rather computationally expensive to recalculate the whole tree on every signature. Thats why we should cache as many in-between hashes as possible since every already stored hash reduces the computational expenses by a factor of 2. The verification stays the same.

This scheme is know as the eXtendes Merkle Signature Scheme (XMSS) which also has some further variants and developments. [12]

Another disadvantage of schemes as described is the so-called statefulness, which means that the signer cannot just sign any message with a key after being reset, since some kind of state is needed that would be lost in a reset. [1]

In a stateless scheme on the other hand, all you need to sign a message is a static private key. That brings us to the other kind of signature schemes, ones that are more similar to traditional asymmetric crypto in the sense that they rely on not so trivial mathematical problem that is not easily algorithmically solveable. But instead of prime factorization or Elliptic curve calculation, this one seems to be hard to solve, even by a quantum computer. The problem for most of these schemes are Lattice Based.

A lattice in this case is a high-dimensional grid with only integer values. Or to be more precise: "An n-dimensional lattice is the set of vectors that can be expressed as the sum of integer multiples of a specific set of n vectors, collectively called the basis of the lattice—note that there are an infinite number of different bases that will all generate the same lattice" [6] To put it mathematically we can denote a Lattice L as $L = \{\sum a_i * b_i : a_i \in \mathbb{Z}\}$ with $b_0, \ldots, b_n$ being arbitrary basis vectors. The mathematical problems that these schemes are based upon are the shortest vector problem (SVP) where a very short vector between to points need to be found or the Closest Vector Problem (CVP), where a lattice vector needs to be found that is closest to a given arbitrary point. The directly arising problem though is, that to get reasonable security the basis (which serves as a private key) of the lattice needs to be in the range of megabits, which again is not ideal for our use-cases. That is why researchers developed the NTRU cryptosystem, that introduces certain symmetries to the lattice structure s.t. the key sizes can be much smaller while lowering the security only slightly. [6], [7] These new schemes are not only resistant to quantum attacks but also improve efficiency compared to traditional cryptography by having speed improvement by a factor of 10-100. Sadly these lattice structures where vulnerable through lattice reduction techniques to Chosen Ciphertext Attacks (CCA) in the case of encryption schemes. But that was fixed with the introduction of a special padding scheme that made these attacks impossible but also increased the key-lengths [6] . In the case of signature schemes (like NTRUSign) the problem is even more severe. The signature works by first mapping the message to a vector and then signing by solving the CVP for this vector. The problem is, that this procedure leaks information about the private key s.t. it was shown to be practically broken after only

around 400 signatures. To mitigate that issue the signer does not give the actual closest lattice vector, but a lattice vector that is close enough by a certain measure, but not necessary the closest. Therefor the leaked information is nearly neglectable and the signature and private key secure for around a billion signatures, although it is still advised to change the private key after around 10 million signatures. That is totally feasable compared to some MTS mentioned before since in many cases 10 million signatures is a whole lot.

Actual Lattice based implementation that were proposed in 2017 are GPV, GLP and BLISS. But now there are newer and better implementations like FALCON that will be discussed in section IV-B.

IV. QR Signatures in IoT

1) Performance Metrics in IoT:

A. qTESLA

B. FALCON

V. Conclusion

References

[1] S. Suhail, R. Hussain, A. Khan, and C. S. Hong, "On the role of hash-based signatures in quantum-safe internet of things: Current solutions and future directions," *IEEE Internet of Things Journal*, vol. 8, no. 1, pp. 1–17, 2021.

[2] https://github.com/PQClean/PQClean.

[3] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Review*, vol. 41, no. 2, pp. 303–332, 1999. [Online]. Available: https://doi.org/10.1137/S0036144598347011

[4] L. K. Grover, "Quantum mechanics helps in searching for a needle in a haystack," *Phys. Rev. Lett.*, vol. 79, pp. 325–328, Jul 1997. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.79.325

[5] C. H. Bennett, E. Bernstein, G. Brassard, and U. Vazirani, "Strengths and weaknesses of quantum computing," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1510–1523, 1997. [Online]. Available: https://doi.org/10.1137/S0097539796300933

[6] R. A. Perlner and D. A. Cooper, "Quantum resistant public key cryptography: A survey," in *Proceedings of the 8th Symposium on Identity and Trust on the Internet*, ser. IDtrust '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 85–93. [Online]. Available: https://doi.org/10.1145/1527017.1527028

[7] T. M. Fernández-Caramés, "From pre-quantum to post-quantum iot security: A survey on quantum-resistant cryptosystems for the internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6457–6480, 2020.

[8] A. Khalid, S. McCarthy, M. O'Neill, and W. Liu, "Lattice-based cryptography for iot in a quantum world: Are we ready?" in *2019 IEEE 8th International Workshop on Advances in Sensors and Interfaces (IWASI)*, 2019, pp. 194–199.

[9] M. J. O. Saarinen, "Mobile energy requirements of the upcoming nist post-quantum cryptography standards," in *2020 8th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)*, 2020, pp. 23–30.

[10] R. Merkle, "A certified digital signature," vol. 435, 08 1989, pp. 218–238.

[11] I. Foundation, "Assuring authenticity in the tangle with signatures," https://blog.iota.org/assuring-authenticity-in-the-tangle-with-signatures-791897d7b998/, 2019.

[12] C. Cheng, R. Lu, A. Petzoldt, and T. Takagi, "Securing the internet of things in a quantum world," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 116–120, 2017.