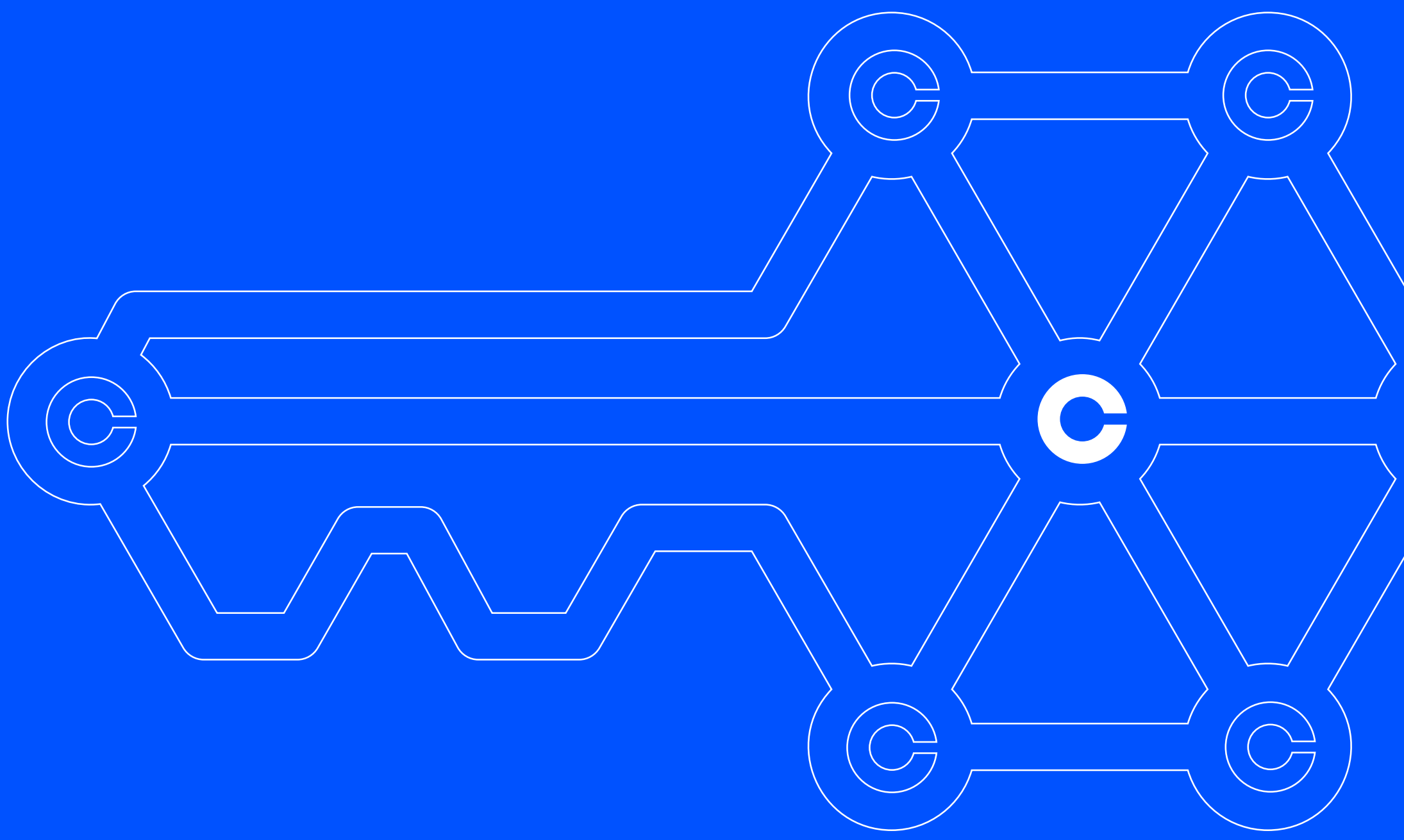
2026 Q1

coinbase **INDEPENDENT ADVISORY BOARD** ON QUANTUM COMPUTING AND BLOCKCHAIN

Quantum Computing & Blockchain

Authored by **the Coinbase Independent Advisory Board on Quantum Computing and Blockchain**



Position Paper **Date of publication: April 21, 2026**

Prof. Scott Aaronson
(UT Austin),

Prof. Dan Boneh
(Stanford),

Justin Drake
(Ethereum Foundation),

Prof. Sreeram Kannan
(Eigen Labs and University of Washington),

Prof. Yehuda Lindell
(Coinbase and Bar-Ilan University),

Prof. Dahlia Malkhi
(UCSB)

TL;DR (Key Takeaways)

Quantum computing should not be ignored:

We have high confidence that a large-scale, fault-tolerant quantum computer (FTQC) will ***eventually*** be built. As such, blockchains and the wider cryptographic ecosystem must prepare for this eventuality.

Cryptographic threat is not imminent but is now clearly on the horizon:

Breaking current cryptography requires a Fault-Tolerant Quantum Computer (FTQC), a machine significantly more complex than current devices. Building an FTQC capable of running Shor's algorithm for standard encryption (e.g., breaking 256-bit elliptic-curve keys) would require a large number of physical qubits and tens of millions of operations, which remains a significant engineering challenge. It is impossible to give an exact timeline for this, but the paper provides milestones that, if achieved, will indicate progress.

NIST recommends that PQ migrations should be carried out by 2035. This recommendation may reflect a judgment on the part of NIST and other US government agencies that a decade is an appropriate timeframe for migration.

We are not confident that cryptographically relevant QCs (CRQC) will not exist by 2035 or later, as recent research raises the possibility that the timeline may be shorter.

Quantum Simulation is the Primary Driver:

The realistic near-term commercial hope and primary driver of funding and development for quantum computers is quantum simulation (for physics, chemistry, and materials science). Progress in this field is the most reliable leading indicator for advances toward a cryptographically-relevant quantum computer.

PQC is Needed at All Blockchain Layers:

Post-quantum migration is essential for both the Consensus Layer (e.g., validator key signing in Proof-of-Stake) and the Execution Layer (e.g., user transaction signatures). Strategies should focus on achieving PQC protection at both layers without incurring large performance or cost penalties.

Post-quantum cryptography and readiness:

Cryptographic schemes for public-key cryptography (for asymmetric encryption, digital signatures and key exchange) that are secure against quantum computers have been researched for many years, and good schemes exist. Furthermore, there is a rigorous process by NIST that is still in process, but has already standardized a number of these schemes. In addition, many blockchains are already in the process of building a post-quantum strategy. As long as these processes continue at a good pace, the blockchain field should be ready for any quantum threat well before it arrives.

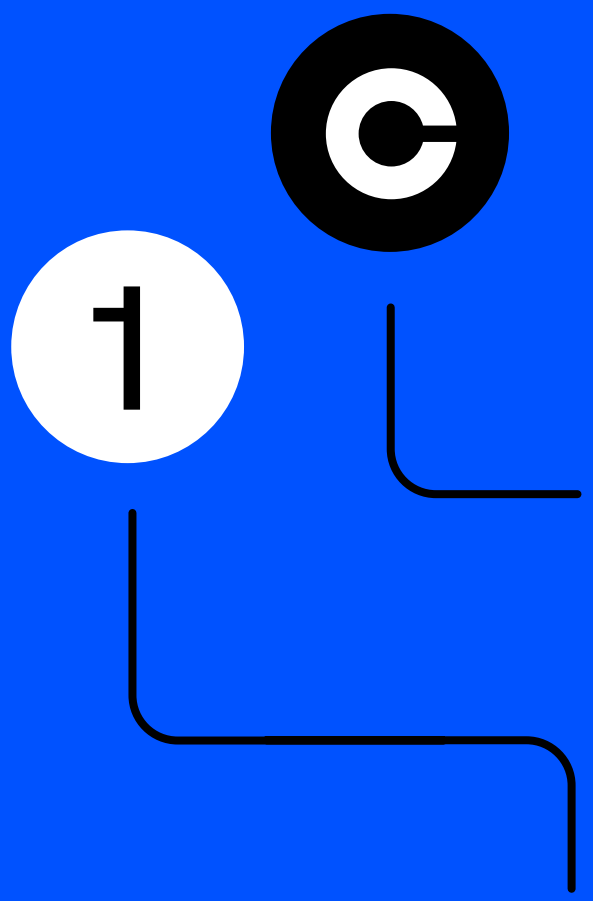
Migrations:

The blockchain community – including blockchains themselves, wallet providers, exchanges and custodians, and so on – should prepare for a quantum future earlier rather than later. Waiting for it to be urgent is not a good idea. Having said that, where possible, strategies should be adopted that provide PQ protection or the ability to quickly add PQ protection without incurring large performance penalties already today. The paper includes migration strategies of this type in proof-of-stake consensus mechanisms and for the execution layer, as well as descriptions of existing plans for some of the major blockchains. The discussion regarding quantum computing often revolves around the timeline. However, we believe that this debate on timelines is largely irrelevant (beyond that it is not imminent) since migrations should be planned for and prepared now. By properly preparing, we can be sure that by the time cryptographically- relevant quantum computers arrive, we will be ready.

Key challenges:

Although we have post-quantum solutions, there are still challenges that are important to resolve. Cryptography-wise, these include performance, with an emphasis on signature size (current PQ signature standards are much larger than classic ones), the design of post-quantum signature schemes with special properties like aggregation and efficient threshold signing, and the need to gain greater confidence in the security of the relatively new post-quantum cryptographic schemes (via research and initiatives like the Poseidon challenge). Beyond the cryptographic and technological, there are also community challenges. These include operational challenges like getting users to transition to new post-quantum schemes, and blockchain communities reaching decisions on how and when to update. One particularly difficult decision that many blockchains will need to deal with is how to handle dormant wallets that do not transition to post-quantum addresses by the given deadline. With the understanding that these are objectively difficult issues to decide, we strongly recommend that blockchains work to reach decisions sooner rather than later, and make these decisions public. This will mitigate current market uncertainty which is already deterring some from investing or increasing their investments in crypto.

Index



Quantum Computing Overview and the Current State of the Art

coinbase Independent Advisory Board on Quantum Computing and Blockchain

1.1: What is a Quantum Computer?

A quantum computer is a proposed new type of computer, which would leverage the counterintuitive laws of quantum mechanics to solve certain problems dramatically faster than we know how to solve them with a conventional (or "classical") computer.

In the same way that the building blocks of a classical computer are bits—i.e., 0's and 1's—the building blocks of a quantum computer are *qubits,* which can be in a so-called superposition of the 0 and 1 states. The power of quantum computers is directly related to the fact that, to describe a superposition with N qubits, one needs a list of 2^N parameters. When (say) N=1000, this is already more parameters than could be written down in the observable universe.

Though the idea of a quantum computer was first proposed by Richard Feynman, David Deutsch, and other physicists in the early 1980s, modern interest in the subject dates to two seminal theoretical discoveries of the mid-1990s: Shor's quantum factoring algorithm and the theory of quantum error correction. The first discovery showed how quantum computers could threaten many of the forms of encryption that protect the Internet (then as now), while the second showed how they could in principle be scaled up even with noisy and imperfect hardware. All the progress in the decades since, both the theory of quantum algorithms and the engineering of quantum hardware, has built on this foundation.

Contrary to a popular misconception, a quantum computer would not be a generic "magic box" that speeds up the solutions to all problems that are hard for classical computers, from optimization to machine learning to cryptography. Nor would a quantum computer work by simply "trying all possible solutions in parallel"---a dramatic oversimplification of a mathematically subtle phenomenon. Instead, a quantum computer would exploit the way that quantum mechanics changes the rules of probability themselves, replacing probabilities (which are real numbers from 0 to 1) by complex numbers called "amplitudes." In a quantum algorithm, the goal is to choreograph a pattern of interference, where each incorrect answer has the various contributions to its amplitude "interfere destructively" and cancel out, whereas the correct answer has the contributions to its amplitude point in the same direction and reinforce each other. When (and only when) this can be arranged, the right answer can then be observed with a high probability.

What makes choreographing such a pattern tricky is that we ourselves don't know in advance which answer is the correct one: if we did, the task would be trivial! Also, of course, the whole exercise is useful only if it reveals the correct answer faster than any classical computer could do the same. But classical computing is itself a moving target: new classical algorithms are often discovered, and there are mysteries even about the limits of classical computation.

Crucially, what we're describing above are not engineering limitations, but would apply even to a perfect quantum computer, so long as it's bound by the laws of quantum mechanics.

1.2: What Is a Quantum Computer Good For?

Despite progress in many aspects of quantum algorithms, the known real-world applications of quantum computers still fall into three central classes that were recognized thirty years ago. **These are:**

(1) The simulation of quantum mechanics itself, for physics, chemistry, and materials science. This was the "original" application that Feynman, Deutsch, and other physicists had in mind when they proposed quantum computing in the 1980s. It's still probably the economically most important application known, as we discuss in more detail below.

(2) Breaking certain public-key cryptosystems: specifically, RSA, Diffie-Hellman, elliptic-curve cryptography, and other systems based on mathematical problems involving abelian groups. This was the famous application discovered by Peter Shor in 1994, and which brought quantum computing to much of the world's attention. It's crucial to understand that the advantage here depends on exploiting special structure in these particular cryptosystems, in just the way discussed before: arranging destructive interference on amplitudes of invalid keys, and constructive interference for the amplitude of the valid key. (This is achieved, in the case of Shor's algorithm, using a Fourier transform.) It is not a simple matter of "trying all possible keys in parallel," and it does not generalize even to all the known public-key cryptosystems, let alone to all of cryptography or all search problems. As will be discussed elsewhere in this document, efforts to generalize Shor's algorithm to break, for example, lattice-based cryptosystems have been unsuccessful so far.

(3) Search, machine learning, sampling, and estimation. Quantum algorithms have been discovered for many of these workhorse tasks of classical computer science, but the speedups they offer tend to be at least one of (a) speculative, (b) specialized, or (c) modest. As a central example, *Grover's algorithm,* discovered in 1996, works for an enormous class of search and optimization problems, including, e.g., mining of proof-of-work cryptocurrencies, and attacking nearly any cryptosystems, including the ones unaffected by Shor's algorithm. The downside is that the speedup of *Grover's algorithm* over classical algorithms is much more modest than for Shor's algorithm: specifically, it's *square-root* or *quadratic*, rather than exponential. As a result, cryptanalytic attacks based on **Grover's algorithm** can typically be neutralized by simply doubling the length of the key. Furthermore, because of the enormous constant overhead of running a fault-tolerant quantum computer, it's expected to be a long time before Grover's algorithm provides any advantage over classical computers at all. While many quantum search algorithms are now known besides Shor's and Grover's algorithms—including the so-called *adiabatic* and *QAOA* and *DQI* and *HHL* and *quantum walk* algorithms—suffice it to say that the speedups they offer are virtually always either "Grover-like" (i.e. modest), or show up only for specialized problems of unclear practical importance, or haven't been convincingly shown to exist at all.

Thus, within the near future, the realistic hopes for a *commercial* benefit from quantum computers (as opposed to a military or intelligence benefit) rest almost entirely on the first class of application: quantum simulation. This could conceivably be useful for, e.g., understanding high-temperature superconductivity, or designing new types of batteries, solar cells, proteins, or industrial chemical reactions including applications like more efficient production of fertilizers. If quantum simulation succeeds in these areas, the benefits could be huge, enabling the design of new materials with special properties. For example, a room temperature superconductor that does not require special conditions (like high pressure) could revolutionize the energy sector and much more.

Having said the above, even here, the prospects for real-world benefit are uncertain, depending for example on what new materials and chemical reactions exist to be found, and also, on whether approximation methods running entirely on classical computers (enhanced by the full arsenal of modern AI) can deliver results that are good enough in practice, rendering simulation on a quantum computer unnecessary.

We stress this point because progress in quantum computing hardware, just like in any kind of hardware, will ultimately depend on a "virtuous cycle," in which real-world applications drive funding, which allows the construction of better devices, which deliver more applications, which lead to more funding, and so forth. At present, the prospect of quantum simulation bears a large fraction of the weight of driving this cycle. Presumably, only if the cycle moves forward will quantum computing reach the point where, as a byproduct, it breaks current public-key cryptography. In particular, if quantum simulation does not materialize soon, the timeline for a cryptographic threat might extend significantly as (non-government) funding dries up. Note also that breaking cryptography alone may not necessarily financially justify the investment in constructing a quantum computer since our digital infrastructure can just move to cryptography that is secure against quantum attackers (aka, post-quantum cryptography) when needed.

1.3: Hardware and Timelines

So, how long do we have? When exactly will quantum computers be built, capable of running Shor's algorithm to break the versions of RSA, Diffie-Hellman, and elliptic curve cryptography used to protect cryptocurrencies?

Of course no one knows. But in thinking through this question, it's crucial to understand the role of *fault-tolerant quantum computation (FTQC)*. Running Shor's algorithm, to (say) factor one of the 2048-bit integers used in cryptography, would require millions of the elementary operations known as *two-qubit gates*. But two-qubit gates are inherently noisy, and the noise, if unaddressed, compounds over time, eventually rendering the quantum computation useless. Thus, it's widely believed that FTQC is *required* for any quantum computation that would threaten cryptography.

In principle, FTQC provides a way to build a reliable quantum computer out of unreliable parts—provided the parts aren't too unreliable. More precisely, a seminal result called the *Threshold Theorem*, proved in 1996, shows that there exists a constant c>0 such that, as long as every two-qubit gate fails with independent probability at most c, it's possible to "correct the errors faster than they happen," and thereby perform a quantum computation as long as desired. Doing this requires encoding every "logical" qubit, in the original quantum computation we care about, as an entangled state of hundreds or more "physical" qubits. Meanwhile, the number of two-qubit gates could get multiplied by thousands or even millions.

This is how, for using Shor's algorithm to factor a 2048-bit integer, people originally got estimates involving millions of physical qubits, and billions or trillions of two-qubit gates. With current technologies, devices capable of shuttling qubits around on such a scale could easily fill buildings. Note that recent work, referenced at the end of this document, has improved these estimates by perhaps two orders of magnitude, although at least another two orders of magnitude still remain between the scale that's been demonstrated experimentally and the scale known to suffice to breaking deployed cryptosystems.

To perform FTQC, one will also constantly need to *measure* the physical qubits, to find out where errors have occurred and what needs to be done to correct them. Decoding the "error syndromes" will need to be done using classical control hardware, in parallel for all qubits, and this will need to happen fast enough to keep up with the errors—placing stringent demands on classical computation. These difficulties help to explain why FTQC is only now beginning to move to the lab.

A final point is that, if the rate of error in the physical two-qubit gates rises above the critical "threshold" c, the errors will then overwhelm the ability of FTQC to correct them, with each round of error correction introducing more errors than it fixes. This is why perhaps the most important way to gauge progress toward building scalable quantum computers has been to ask: *what error rate for two-qubit gates has been achieved in the lab, and how does that error rate compare to the threshold for FTQC?* ***Progress here can (and does) happen on two fronts:***

(1) experimentalists can reduce the physical error rate for their two-qubit gates, and

(2) theorists can design fault-tolerance schemes able to cope with higher error rates.

Only when the two numbers cross do we get a "self-sustaining chain reaction," in which errors get corrected faster than new errors are introduced.

We stress this because quantum computing companies often trumpet the sheer number of qubits in their devices—but even millions of qubits *do not suffice* for scalable QC, if one can't reliably perform above-threshold two-qubit gates on those qubits. Worse, sometimes companies advertise the size of a number they're able to factor with a current device. Typically, these sorts of estimates don't use Shor's algorithm at all, but some other quantum algorithm (for example, adiabatic optimization) with poor scaling behavior on factoring. Such estimates are therefore useless for estimating progress toward scalability.

1.4: Where Are We?

Initial estimates for the fault-tolerant threshold suggested that 99.9999% accurate two-qubit gates would be needed. However, using modern fault-tolerance schemes, 99.9% accurate two-qubit gates might suffice, if one is willing to accept a large overhead (that is, many physical qubits per logical qubit, and many physical gates per logical gate).

Meanwhile, the two-qubit gate accuracies that are experimentally achievable in various hardware platforms have climbed from ~50% to ~90% to ~99% and even ~99.9%, according to public knowledge. Within the past year, Quantinuum and Google announced devices with ~99.9% accuracy two-qubit gates, which can be applied to pairs from among ~100 physical qubits. If this accuracy can be maintained, as one scales to tens or hundreds of thousands of physical qubits, it will theoretically suffice for FTQC. Having said that, scaling will be an enormous engineering endeavor. As one scales to larger systems, one faces new challenges: for example, to route qubits across longer distances (in the case of trapped-ion and neutral-atom qubits), or otherwise engineer longer-range communications between them (in the case of superconducting qubits). So, it becomes nontrivial even just to maintain the same two-qubit accuracies that were previously demonstrated, in the context of the larger system.

These uncertainties explain why it's so hard to answer the question "how many years until quantum computing threatens current cryptography?" On the one hand, the requisite building blocks have now been demonstrated in isolation; on the other hand, the task of integrating them into a large system still looks formidable. Among quantum computing researchers, one hears estimates these days of anywhere from a few years to a decade or more. In lieu of certainty, perhaps the best we can do is to stay alert to what *is* happening and react appropriately.

In our view, the following milestones would indicate clear progress toward a cryptographic threat:

- **Fault-tolerant two-qubit gates, performed more reliably than the corresponding physical gates, and/or more reliably as one scales to larger code sizes.**
- **Fault-tolerant implementation of Shor's algorithm used to factor a small number (even just 21), or some other demonstration task.**
- **A single logical qubit maintained in its state indefinitely through the use of quantum fault-tolerance.**
- **Advantages for quantum simulation tasks that matter in practice and can be verified using a classical computer.**

Conversely, delay in accomplishing these milestones would be a clear sign of delay toward cryptographically relevant quantum computing.

Of course, one needs to allow for the possibility that, once cryptographically relevant quantum computing is near, further advances toward that goal might stop being published openly, whether voluntarily or by government pressure or decree. We clarify that we see little indication that this has happened yet.

1.5: Types of Hardware

One might have thought that, by the time quantum computing had reached its current stage, there would be a single hardware platform that dominated all the others, analogous to the silicon transistor for classical computing. Surprisingly, however, there are still radically different hardware platforms for quantum computing being pursued in parallel, with complementary strengths and weaknesses. **Here are five of the main ones:**

Superconducting QC.

Pursued by Google, IBM, Rigetti, and D-Wave among others. Collective quantum states of electrons in superconducting circuits are used as the qubits. Two-qubit gate accuracies now surpass 99.8% in state-of-the-art systems. Advantages of superconducting qubits include extremely fast gate times, and the ability to build systems with hundreds of qubits using current chip fabrication technology. A disadvantage is that the qubits are fixed in place, so interactions between faraway qubits need to be engineered via a sequence of swaps or some other device. Scaling up seems to require integrating multiple superconducting chips, which hasn't been demonstrated at the time of writing.

Trapped ion QC.

Pursued by Quantinuum and IonQ among others. Spin states of atomic nuclei are used as the qubits. Two-qubit gate accuracies now surpass 99.9% in state-of-the-art systems. Advantages include superb accuracy, the fact that the qubits are all identical, and the fact that the qubits can be moved around using laser pulses for all-to-all connectivity. A disadvantage is slow gate times.

Neutral atom QC.

Pursued by QuEra and Atom Computing among others. Degrees of freedom of neutral atoms are used as the qubits. Two-qubit gate accuracies now surpass 99.5% in state-of-the-art systems. Advantages include the ability to scale to hundreds of qubits with parallel operations. However, targeting operations to individual qubits tends to be harder in this approach. Another disadvantage is slow gate times.

Photonic QC.

Pursued by PsiQuantum, Xanadu, and USTC (funded by the Chinese government) among others. States of photons are used as the qubits. An advantage is superb coherence times; a disadvantage is the need to constantly generate new photons to replace lost photons, and the need for some adaptive measurement scheme to get universal quantum computation (given the lack of significant interaction between photons). Less is known about the status of this approach compared to others, because PsiQuantum has not prioritized doing concrete demonstrations. However, Xanadu and USTC have demonstrated apparent quantum advantages for an artificial task called BosonSampling.

Topological QC.

Pursued mainly by Microsoft. Topological QC would require creating a new two-dimensional state of matter, which would support excitations called "nonabelian anyons" that could do universal quantum computation just by being braided around each other in an appropriate pattern. The promise of this approach is that, if and when it ever works, it might have less need for fault-tolerance than others, because causing an error would require changing the topology of how the anyons are braided. (In other words, some degree of fault-tolerance would be built into the physics.) At the time of writing, however, experts are still debating whether Microsoft has succeeded in creating even one topological qubit. Thus, topological QC is a longer-term possibility than the approaches above; the basic building blocks remain to be demonstrated.

1.6: Are There Insurmountable Challenges Ahead?

A few distinguished mathematicians, computer scientists, and physicists, including Gil Kalai, Leonid Levin, Michel Dyakonov, and Gerard 't Hooft, have taken the position that quantum computing is *impossible in principle*—that what seem to others like engineering difficulties will in fact never be surmounted for some inherent reason. This could be true, for example, if quantum mechanics itself were to fail as we tried to scale up QC, or if we were to discover some new principle on top of quantum mechanics, which guarantees the existence of noise that violates the assumptions of the Threshold Theorem and therefore can't be corrected by FTQC.

It's important to understand that this was never a mainstream view among experts, at least since the discovery of FTQC in 1996. The mainstream view could be summarized by saying that, if quantum mechanics were to fail, or some new principle were to be discovered that "screens off" or "censors" QC, that would constitute a once-per-century revolution in physics, and would be vastly more surprising and exciting than "mere success" in building a QC that worked as the theory has long predicted. From a physics perspective, QC working as predicted is the *conservative* option.

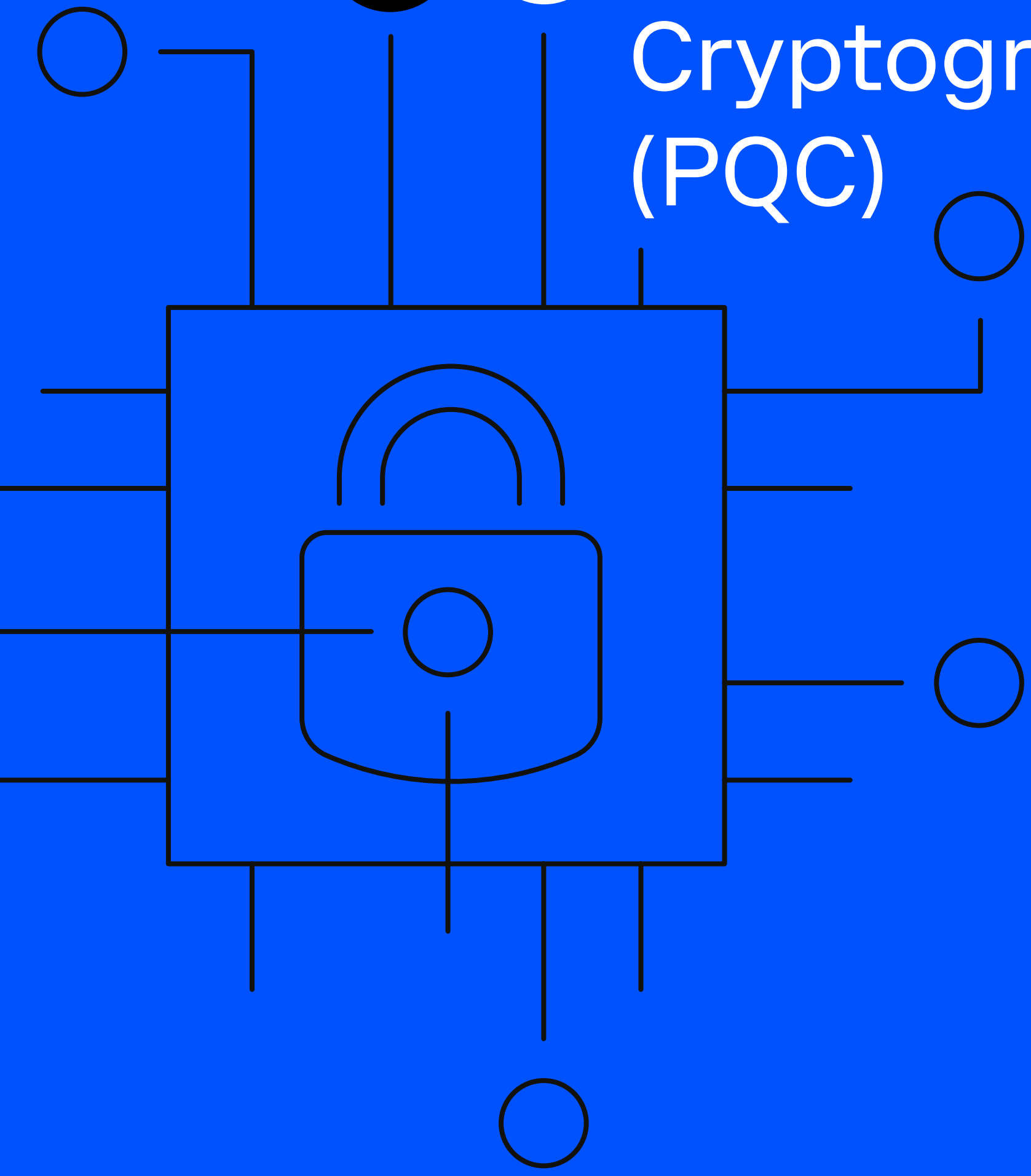
In any case, this debate is no longer in a purely abstract realm. Experiments can now weigh in. Gil Kalai, the most technically engaged of the skeptics, conjectured 15 years ago that, when or before quantum computing reached the scale of hundreds of qubits and thousands of gates, "correlated noise" would be discovered, violating the assumptions of the Threshold Theorem about the independence of errors, and the correlations would be of such of a kind as to render FTQC impossible. Today, however, companies like Google, Quantinuum, QuEra, and IBM routinely do experiments at the scale in question, and no sign of these correlations has been found. On the contrary, if there are T gates that each have an accuracy of p, then the accuracy of the overall circuit is observed to fall off like p^T, exactly as it would if the errors were independent, and of the kind that FTQC can ultimately correct. This has forced Kalai into arguing that there must be some mistake in the experiments or their analysis—a position that becomes less tenable with every new experiment that reports similar results.

In summary, we can say with extremely high confidence that people should not pin their hopes for the durability of existing blockchains, on the belief that QC is impossible for some fundamental reason.

We firmly believe that a large-scale fault-tolerant quantum computer will eventually be built, and that blockchains need to prepare for this eventuality.

This doesn't mean that the threat is imminent, and as we have discussed considerable progress is still needed. However, we believe that the time to begin preparing for it is now.

2 Post-Quantum Cryptography (PQC)



2.1: What is Post-Quantum Cryptography?

As discussed above, quantum computers – if large enough and with sufficiently low errors – can completely break the most-used public-key encryption and digital signature schemes based on RSA and elliptic-curve cryptography, using Shor's algorithm. In contrast, typical symmetric cryptography is assumed to be secure against quantum attacks, as long as the key is long enough to avoid the speedup achieved by Grover's algorithm. In more detail, Grover's algorithm enables a key-search in square-root of the size of the key space. Thus, a 256-bit key gives 128 bits of security (since √(2^256)=2^128). For collision-resistant hash-functions, 128-bit security requires a 256-bit output for classical computers. Although faster quantum algorithms do exist, they require an exponential number of qubits and are therefore impractical. We therefore conclude that AES-256 and SHA-256, for example, are assumed to be secure against quantum attacks.

Stated simply, post-quantum cryptography or PQC refers to cryptosystems that are secure against quantum attackers. Thus, RSA and elliptic curve cryptography are not post-quantum. In contrast, standard block ciphers and hash functions are post-quantum, as long as the key-lengths/outputs are large enough. Therefore, the main challenge is to construct post-quantum public-key encryption schemes and digital signatures. Fortunately, academic research in this area goes back over 20 years, and is aligned with the academic mission of studying problems that may become relevant decades into the future. Thus, we have quite a good understanding today of how to construct post-quantum cryptosystems, with the caveat t hat because we don't actually have quantum machines, the state of quantum cryptanalysis is still quite immature.

Clarification – PQC vs QC:

We stress that post-quantum cryptography is run on *classical computers* and is secure against *quantum attackers*. This is in contrast to things like QKD (quantum key distribution) which requires the (honest) users to use quantum systems. Although of academic and theoretical interest, "quantum cryptography" is impractical in general, and certainly not of relevance in the blockchain space. We therefore do not consider it in our discussions.

2.2: NIST Post-Quantum Standardization

The National Institute of Standards and Technology (NIST) began the long process of **standarding post-quantum cryptosystems** for public-key encryption, digital signatures, and key exchange algorithms way back in 2015.

This process, similar to the standardization of AES, has been run as an open competition with different teams submitting candidate schemes and publicizing them for open review and cryptanalysis. There have been four rounds in the competition, and a subset of the candidates were chosen to proceed in each round based on security and other criteria. The current status is that three algorithms have been standardized (in August 2024) after passing the fourth round: **FIPS 203** standardized ML-KEM as a public-key encryption scheme based on lattices (technically a key encapsulation mechanism, but achieving the same effect), **FIPS 204** standardized ML-DSA as a digital signature scheme based on lattices, and **FIPS 205** standardized SLH-DSA as a digital signature scheme based on hash functions. Furthermore, an additional lattice-based signature scheme called FN-DSA is in the process of being standardized as FIPS 206. (We describe below what it means to be lattice based, or hash-function based.)

NIST is still continuing in the standardization process, with the aim of adding additional algorithms. This is due to the understanding that although these schemes have been well studied and have undergone considerable public scrutiny, our understanding of the security of these types of schemes and our understanding of quantum cryptanalysis is still far from our understanding of things like block-cipher constructions and the hardness of the problems underlying RSA and elliptic-curve cryptography.

Observe that NIST has not yet standardized any key-exchange protocols, and this seems to be in part due to the fact that a leading contender SIKE, which was a round-4 candidate, was completely broken in 2022. SIKE is a scheme that is based on a different underlying hard problem called isogenies (all cryptographic constructions utilize hard problems at their core). The scheme managed to survive *five years* of the competition, and was eventually so badly broken that the key could be extracted in one hour on a single-core in a regular computer. This was very surprising since the scheme made it so far in the process and was considered a leading candidate. Furthermore, the scheme was completely broken using *classical computing* and not a quantum algorithm. We stress that this is not a failure of the process. On the contrary, this is a strong validation of the methodology employed by NIST, and it gives us higher confidence in the schemes that do survive. Having said that, it does emphasize that our confidence in RSA and elliptic-curve cryptography is much higher than any PQC scheme which is much newer.

2.3: Post-Quantum Digital Signatures

Digital signature schemes are central to securing blockchains, both at the consensus layer and at the execution layer.

There are a number of different post-quantum signature schemes, based on different assumptions and methods. Two of the leading methods are lattice-based and hash-based, and indeed the two post-quantum signature schemes that have already been standardized by NIST are ML-DSA (based on lattices) and SLH-DSA (based on hash functions). **We now discuss each of these:**

Lattice-based signatures:

A lattice is a mathematical structure that invites certain problems that are assumed to be hard, even for quantum computers. Cryptography is based on hard problems, and so lattices can be used in order to achieve schemes like digital signatures that are secure even against attackers with large-scale quantum computers. Lattice-based schemes are very attractive for the following reasons. Lattice-based hardness assumptions are not new, and have been studied for almost 30 years. Furthermore, lattice-based signing and verification (like that standardized in ML-DSA) is very fast and comparable to schemes like ECDSA and EdDSA. However, lattice-based signatures can be much larger, with ML-DSA public keys and signatures being approximately 40 times larger than ECDSA. This size difference can have a significant impact, as discussed below. Finally, although we have a good understanding of lattice-based hardness assumptions in general, as mentioned above, we do not have the same level of confidence as for schemes like ECDSA that have been in use for decades.

Hash-based signatures:

This method relies on the security of hash functions. When instantiating these with hash functions like the SHA family, the level of confidence that we have regarding their security is very high, and indeed comparable to schemes like ECDSA (if not even better). However, standard plug-in hash-based schemes, like those standardized in SLH-DSA, are very challenging from a performance point of view. For example, SLH-DSA comes in two variants – a variant with smaller signatures and slower signing, and a variant with larger signatures and faster signing. The smaller signatures are still about 100 times larger than ECDSA signatures, and signing time for this variant is about 10,000 times slower. The faster signing variant has signatures about 250 times larger than ECDSA with signing time about 1,000 times slower. Deploying these schemes on blockchains will clearly be very challenging.

* When we say comparable here, we do not mean fully equivalent but rather that the signing and verification times are similar enough to not really matter in blockchain applications (in particular, within one order of magnitude of each other, and typically much less than that).

In order to address this, there are two approaches:

1. ***Use stateful signatures:*** A standard signature scheme is stateless, meaning that each signature is generated as a function purely of the (initial) private key and the message being signed. In contrast, stateful hash-based signatures like **XMSS** require the signer to hold a record of all previously-signed messages or to update the key after each signing operation. The use of state is not an issue for some use cases like Ethereum proof-of-stake attesting, but for user transactions on blockchains, it introduces many challenges. For example, it is not sufficient to backup the initial key anymore since restoring from backup will result in rewinding the state to the beginning which renders the scheme completely broken, and it isn't possible to sign independently using different devices since different devices may reuse the same state path which breaks the scheme. These barriers are not insurmountable, but they certainly mean that stateful signatures cannot serve as a straightforward replacement. In addition, although these are much more efficient than stateless hash-based signatures, the public key and signature sizes are still at least an order of magnitude larger than ECDSA.

2. ***Use signature aggregation with ZKSNARKs:*** ZK SNARKs enable aggregating many signatures into a short message, while also enabling fast verification of all signatures. This can be used to resolve the problem of hash-based signature sizes on blockchain in settings where aggregation is possible. However, the generation of the ZK SNARKs themselves can be very expensive, especially if standard hash functions like the SHA family are used. It is possible to replace these with "ZK friendly" hash functions, like Poseidon. However, in this case, we return to the problem of confidence in the scheme; although Poseidon is widely used today and we have no reason to doubt its security, our confidence in the security of a signature scheme based on Poseidon is not as high as for SHA or as for schemes like ECDSA. In addition, the introduction of signature aggregation requires significant changes to blockchains and significant additional cost (especially for stateless schemes), and so isn't something that can serve as an immediate replacement.

Aggregate signatures:

As will be discussed below, many consensus mechanisms use digital signatures. In practice, BLS is widely used since it is an *aggregate signature* scheme, meaning that individual parties can each sign and the signatures can be publicly aggregated into one signature afterwards. As of now, PQ aggregate signatures are still an active area of research and although there are solutions, they do not have the performance of BLS and are a challenge to use in consensus settings.

Threshold signatures:

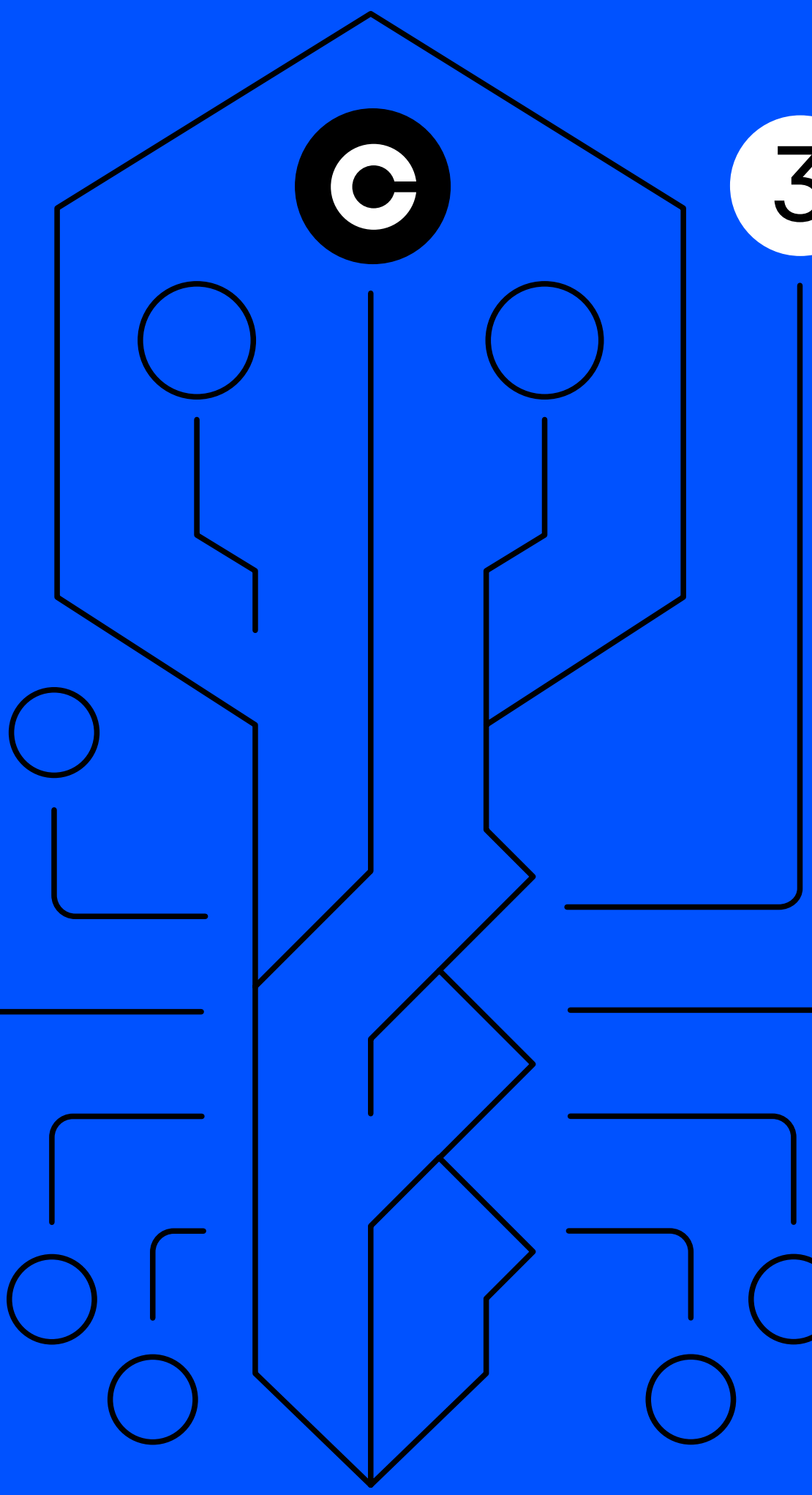
Briefly, a *K*-out-of-*N* threshold signature scheme, is a variant of a digital signature scheme in which the ability to sign is split among *N* users in such a way that any *N* of them can construct signatures, while any group of *K* – 1 or fewer users cannot. These are used both in consensus, as well as in many key-management systems to provide greater control over signing and to prevent single points of failure where breaking into a single machine that holds a full key suffices to steal all the funds protected by that key. As with aggregate signatures, although there are PQ threshold signature schemes, better schemes are needed and this is an active area of ongoing research.

2.4: Comparing PQ-Signatures to Existing Schemes

The following table is taken from a **Cloudflare blog** on PQ signature standardization (November 2024). There are two SLH-DSA variants presented; 128s is optimized for smaller signatures, and 128f is optimized for faster signing.

		Sizes (bytes)		CPU time	
PQC		Public key	Signature	Signing	Verification
Ed25519	✗	32	64	0.15	1.3
RSA 2048	✗	256	256	80	0.4
ML-DSA 44	✓	1,312	2,420	1 (baseline)	1 (baseline)
FN-DSA 512	✓	897	666	3	0.7
SLH-DSA 128s	✓	32	7,856	14,000	40
SLH-DSA 128f	✓	32	17,088	720	110

Observe that there is a very clear tradeoff between ML-DSA (FIPS 204) and SLH-DSA (FIPS 205), with ML-DSA achieving much faster signing times and a much smaller signature. In contrast, the public-key for SLH-DSA is very small, but the signatures are much bigger (especially for the 128f version). FN-DSA has signatures and public keys that are smaller than ML-DSA (but still much larger than ECDSA or EdDSA). Once standardized in FIPS 206, this may be a better default for blockchain than ML-DSA. In any case, it is clear that PQ signatures are significantly more expensive than elliptic-curve ones in use today, and how this impacts blockchains will be discussed below.



3 Post-Quantum Cryptography and the Consensus Layer

3.1: What is Consensus and State-Machine *(Virtual Machine)* Replication

At the core of blockchains is an engine that forms consensus on a ledger of transactions that maintain the state of financial assets. The distributed ledger serves as a ground-truth source of ownership of digital assets, replacing traditional databases maintained by centralized banks and clearinghouses.

Distributed consensus systems follow the state-machine replication paradigm and are conceptually built from two layers: the **consensus layer,** which establishes agreement on a sequence of transactions, and the **execution layer,** which is implemented as a state machine (or a "virtual machine") that starts from a known genesis state and applies transactions atomically to the state in the agreed, deterministic order.

Making both these layers PQ-safe is about replacing cryptographic primitives without breaking performance or liveness guarantees. Generally, key concerns in migrating to PQ safety is the size of data and cost of computation. An additional challenge is orchestrating active switchover of cryptographic keys by users.

In this section, we focus on the consensus layer. We begin by first giving background on what types of classic cryptography is employed in different consensus cores.

3.2: What Types Of Classic Cryptography Can Break In Different Consensus Cores?

We first discuss the use of PQ-vulnerable cryptographic primitives at today's distributed consensus technologies. Generally, the main vulnerabilities stem from Shor's algorithm which a powerful PQ computer may use to break classical public-key cryptography. Additional attention is paid to Grover's attack on hash functions.

Below we discuss Bitcoin's Proof-of-Work based Nakamoto Consensus (NC) potential vulnerability to Grover's attack. We then explore the use of classical cryptography in various types of Proof-of-Stake based Byzantine fault tolerant (BFT) Consensus.

3.3: Nakamoto Consensus

Nakamoto Consensus relies on the difficulty of inverting hash functions to secure a cryptographic puzzle mechanism used to secure the formation of the Bitcoin chain. Additionally, it uses the hardness of finding hash collisions to maintain a structurally secure chain via hash-chaining. In principle, Grover's attack poses a threat by reducing the search time; for the case of solving the cryptographic puzzle the reduction is quadratic. In practice, however, Grover's quadratic speedup does not translate to a real speedup for the puzzle sizes due to the much slower time per qubit operation in a quantum computer versus a highly optimized ASIC used for mining today. Thus, Nakamoto consensus mechanisms are essentially post-quantum secure.

3.4: Byzantine Fault Tolerance (BFT)

Many blockchains consider PoW too energy-costly or slow, and instead rely on solutions to the BFT Consensus problem in order to collectively maintain the security and integrity of a global ledger by a designated committee of nodes. In most practical deployments, the configuration of the consensus committee is dynamically changing, and itself maintained by and on the blockchain. A popular form for governing the committee configuration is Proof-of-Stake.

The BFT engines of different blockchains make use of a variety of cryptographic primitives. Essentially all BFT protocols use public key cryptography–for example, Sui, Aptos and Ethereum use BLS signatures–for maintaining consensus among validators. Algorand relies on a Verifiable Random Function (VRF) for randomness generation and validator selection. Ethereum uses KZG commitments for data availability sampling. All of these primitives are threatened by Shor's algorithm if run on a sufficiently powerful quantum computer, thereby compromising their security.

3.5: BFT and Aggregate/Threshold-Signatures

Many blockchains use aggregate signature schemes as a core part of their BFT consensus. The idea is that since votes are cast by validators on the same block, votes may be aggregated or thresholded in order to reduce the costs associated with sending validator signatures, verifying and storing them.

For example, in every Ethereum slot, thousands of validators attest (vote) on the next block. Instead of including thousands of signatures on-chain, Ethereum aggregates them into a few BLS signatures and includes a bitfield saying which validators participated. The Ethereum Finality Gadget also runs a BFT algorithm that employs BLS threshold-signatures, and the Ethereum Light Client service uses sync committee BLS-aggregated signatures from (512) validators to read the Ethereum ledger state. Likewise, Sui, Aptos and other HotStuff descendants use BLS threshold signatures and signature aggregation to form certificates on blocks and on steps of the protocol. Other blockchains, like Polkadot, use Schnorr threshold signatures in the consensus protocol, and many other blockchains that use aggregate signatures to reduce communication and storage. Importantly, existing methods for threshold and aggregate signatures are *not* post-quantum secure. Notably, there does not exist any post-quantum scheme with the simple aggregation properties of BLS. Thus, blockchains that rely on aggregate and threshold signatures for consensus are not post-quantum secure, and do not have any easy plug-and-play replacement in order to make them post-quantum secure.

3.6: Consensus over Authenticated Channels and its use in MPC

One class of consensus solutions that is naturally resilient to quantum-computing threats replaces cryptographic signatures with the assumption of authenticated communication channels.

A natural objection is that establishing authenticated channels itself appears to rely, circularly, on public-key cryptography. However, channel establishment is a one-time operation, for which computational cost and signature size are typically not critical concerns. Once a PQ authenticated channel has been established, subsequent communication can rely solely on symmetric cryptography for ongoing operation.

Consensus solutions in this model do not employ signatures, let alone signature aggregation/thresholding. Rather, they use symmetric encryption and hashing. They thus provide resilience to quantum computing attacks on public-key encryption schemes. Early signature-free protocols designed to deliver practical latency against partial synchrony (e.g., Bracha's broadcast and PBFT) suffer from high communication costs and are quite complex, but recent protocols in this model (e.g., Information-Theoretic HotStuff and TetraBFT) are communication-efficient; asynchronous solutions may have higher expected latency but guarantee liveness against full asynchrony (e.g., https://eprint.iacr.org/2024/677 and https://arxiv.org/pdf/2505.02761).

There is, nonetheless, a fundamental limitation of this model: consensus decisions reached among the participants cannot be externally verified by third parties. However, there are good reasons to expect that some blockchains may adopt this approach for their consensus cores. A reasonable design is for validators to use the authenticated channels to commit consensus decisions among themselves. When an external observer needs an attestation on a block, it can query F+1 validators over authenticated channels; alternatively, in practice observers often trust a single RPC server to query the state of blockchains anyway. Additionally, as recommended below for classic consensus chains, these chains can generate periodic (potentially PQ) signed checkpoints.

Additionally, consensus without external verifiability can be harnessed to enhance MPC performance (e.g., in HoneyBadgerMPC and Velox), as well as achieve certain other properties like fairness. Blockchains and MPC share the same assumptions and vision, minimizing trust in individual participants and putting trust in a committee as a whole. There are a myriad of use-cases for MPC in blockchains, that either complement or replace zero-knowledge proofs: an MPC network can implement private smart contracts, e.g., Uniswap Labs auctions, or DAO-style coordination, e.g. Aragon. The Account Abstraction provided, for example, by Near Protocol Chain Signatures maintains collective escrow of a user's cross-chain accounts via MPC. MPC is also implemented over decentralized TEE networks, e.g, in MEV-boost to secure MEV extraction and coordination, and in Chainlink and LayerZero Oracles for trust-minimization and confidential computing.

3.7: The Cost of PQC in the Consensus Layer

When considering PQ secure replacements, the main considerations in the Consensus Layer are the computation time to generate/verify signatures and the cost of communicating signatures employed by validators to vote and form agreement on blocks of transactions.

Each block could have one signature per validator and consensus finality requires signatures from some threshold of validators in the network. Generating these signatures, verifying them, and communicating them present scaling challenges. In Ethereum for example, there are currently an estimated one million validators, two-thirds of which need to provide signed block attestations in the course of each 384 second epoch; that's ~1,000,000 attestations per 6.4 minutes. Other networks typically have a much smaller number of voting validators, e.g., Cosmos and Sui currently have around 200 validators, and Algorand (which has thousands of validators) uses sampled sub-committees via sortition per block. Hence, their signature scaling problem is less acute, but remains a potential performance bottleneck.

3.8: What About Aggregation/ Thresholding?

Even with classical signature schemes, most consensus systems employ aggregation or thresholding to deal with scaling issues. But while threshold/aggregate PQ signature variants exist, they are still in the research / emerging stage and suffer from certain inefficiencies, such as large signatures (a comparable increase as with individual PQ signatures); and slow signing times. There are no fully standardised PQ aggregate or PQ threshold signature schemes published by NIST yet. There are also no plans for NIST to standardise PQ aggregatable signatures.

An important drawback of many existing PQ threshold/aggregate schemes which produce actual signatures as output (not ZKP of their existence) is that they involve multiple rounds of communication among participants. Thus, combining results is not a simple offline algebraic combine like the non-PQ-secure BLS; rather it requires interaction during signing. This is problematic in consensus protocols, especially in asynchronous settings: after each participant computes its partial signature share, the signer nodes must interactively communicate with one another (or with a combiner that interacts with them) to produce the final signature. The crypto literature sometimes calls this the "interactive signing problem" or the "interactive combination bottleneck". In contrast, known threshold schemes for classical signatures such as threshold-BLS are non-interactive.

In addition to the performance gap between current PQ threshold/aggregate research and best classical practice, as in every nascent cryptographic scheme, further scrutiny of emerging standards is needed by community experts to increase confidence in its security.

Another option is to use a generic ZKP-based construction for aggregate signatures -- the combiner proves knowledge of t-of-n signatures. The idea is to compress all signatures using a Merkle tree and have the validation of t Merkle proofs be in a succinct ZK-proof with fast (sub-linear) verification. The nice property of this option is that it has a silent setup and dynamic parameters (it is easy to change n and t). However, presently, proofs that are compressed (e.g., using Groth16 or pairing-based Plonk) become unsound in the presence of a quantum computer, and other proof schemes result in fairly large signatures. Additionally, the generation of ZKP's requires orders of magnitude more computation than mere signing. However, often signature aggregation may be taken off the critical path of consensus, compressing the blockchain lazily post consensus-commitment.

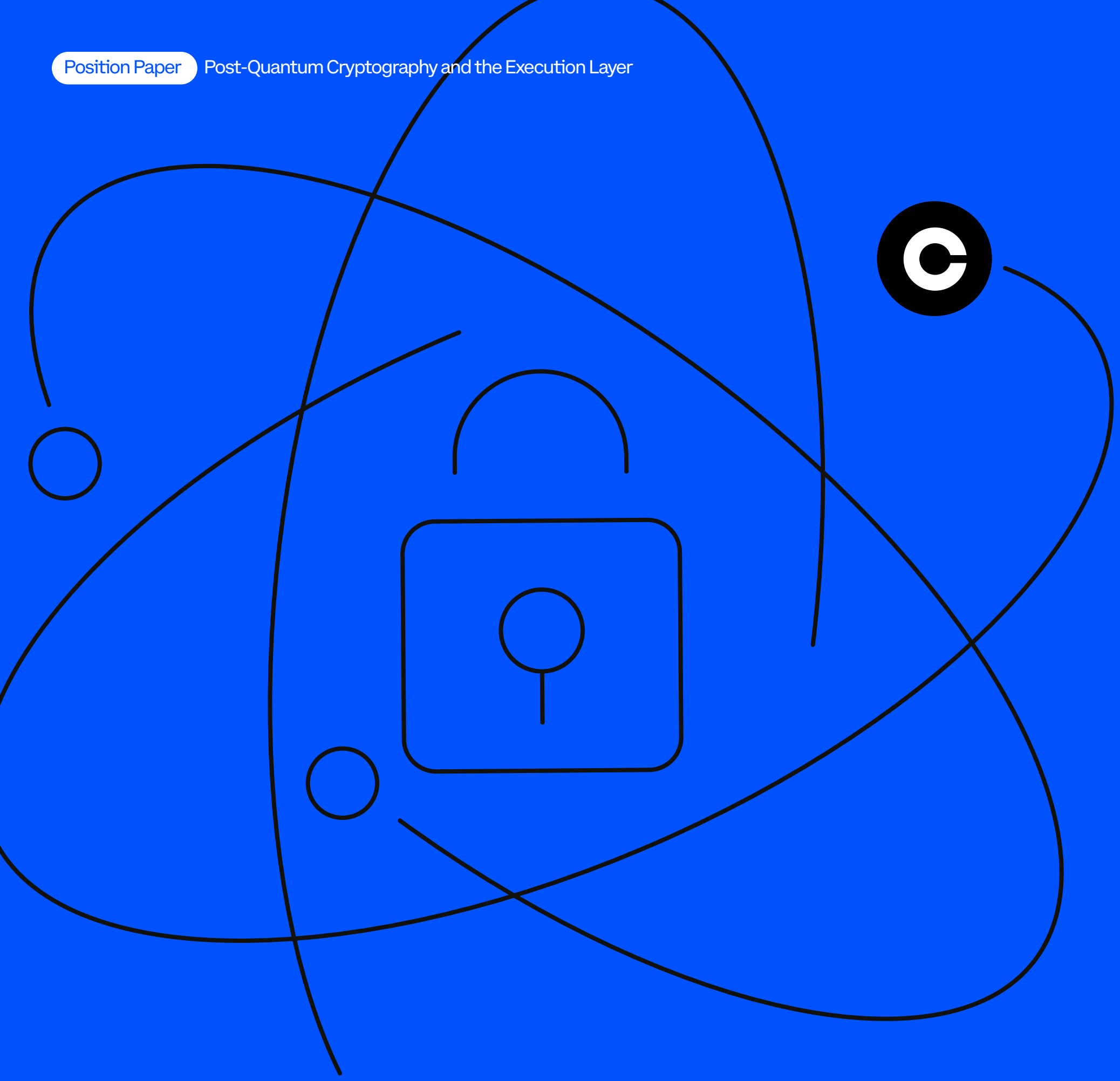
3.9: Recommendations – Consensus Layer

In most blockchain systems, the integrity of the ledger history is naturally reinforced through hash-chaining. Each block contains the cryptographic hash of its predecessor, so validating a block implicitly certifies all prior blocks. As a result, once a block is accepted and later protected by a strong cryptographic primitive, the entire chain of preceding blocks inherits that protection. This property plays an important role in post-quantum migration: the addition of even a single post-quantum signature to the chain can effectively anchor and protect all earlier blocks against retrospective forgery. In other words, backward certification arises natively from the hash-linked structure of the ledger.

Therefore, and given that replacing classical signatures in the consensus core may be complex and may introduce performance overhead, we recommend adopting a staged migration strategy based on checkpoints. In such an approach, PQ signatures are first introduced selectively, for example, by signing every (k)-th block (e.g., every 1000 blocks). These PQ-protected checkpoints act as cryptographic anchors for the history preceding them. Alternatively, checkpointing could be performed retroactively, outside the critical validation path, by periodically producing PQ-signed attestations to the current chain tip. In the event that classical signatures used between checkpoints are eventually broken, the scope of potential damage is confined to the limited range of blocks after the most recent PQ checkpoint. Any disputes within that range could then be resolved through community or social consensus mechanisms.

Fully migrating the consensus cores of existing blockchains ultimately requires replacing the digital signatures produced by validators. Since PQ signature standards are still evolving, we recommend maintaining cryptographic agility at the validator level. This necessitates changes at the consensus protocol level. It also requires publishing the latest configuration of validator public keys somewhere other than inside a consensus decision in order to prevent it from being overridden by a long-range attack.

A particular challenge arises for blockchains whose consensus mechanisms rely heavily on aggregate or threshold signatures. As discussed above, currently known post-quantum signature schemes generally do not offer the same aggregation properties or efficiency. Consequently, migrating such systems to PQ security may require not only replacing the signature primitive but also modifying the consensus protocol itself. While research into post-quantum aggregate and threshold signatures is active and promising, viable plug-and-play replacements are not yet available. For this reason, we recommend that blockchains relying on these mechanisms begin developing a post-quantum migration roadmap as early as possible, including contingency plans that do not depend on the availability of PQ aggregation in the near term.



4 Post-Quantum Cryptography and the Execution Layer

4.1: Transaction Signatures

The execution layer in state-machine replication is responsible for deterministically applying an agreed sequence of transactions to a shared state. In blockchains, this layer executes transactions that mutate the ledger state, such as balances, contract storage, or application variables, starting from a known genesis state.

Cryptographic signatures attached to transactions authenticate the sender and authorize state changes, ensuring that only valid, owner-approved operations are applied during execution. Because every transaction must carry cryptographic authorization data that is broadcast, verified, and stored, transactions are among the dominant contributors to bandwidth and storage costs in most blockchains. For this reason, blockchains invest heavily in compact signature schemes such as ECDSA and Schnorr, as well as multisignature and threshold aggregation techniques, to reduce the transaction footprint. All of these mechanisms would need to be replaced with PQ alternatives.

4.2: Post-Quantum Signing and a Naive PQC Strategy

Theoretically, one can just move over to a post-quantum secure signing scheme like one of those described above, and then quantum computers are no longer a potential threat to the execution layer. Practically, it is far from as simple as that.

The reason for this is that no post-quantum signature scheme can serve as an equivalent replacement. Specifically, consider the strategy of replacing ECDSA/EdDSA with either ML-DSA or SLH-DSA (or FN-DSA when that becomes standardized). Although this seems like a straightforward solution, it is actually quite problematic. Regarding lattice-based schemes like ML-DSA or FN-DSA, we may be actually downgrading security, since we will be moving to a signature scheme which has much less mileage and has not been studied to nearly the depth of schemes like ECDSA and EdDSA. This is especially concerning since we don't actually have a quantum threat right now.

We strongly believe that our strategy for preparing for a potential future attack should not weaken our existing security posture.

However, switching from ECDSA to ML-DSA, for example, would in some sense do exactly that.

Regarding the switch to hash-based signatures, such as SLH-DSA, this either encounters the challenge of much larger signatures with ramifications as discussed below **(The Cost of PQC at the Execution Layer)**, or requires re-designing the blockchain to work with SNARK-based signature aggregation by default. This prevents a plug-and-play replacement and requires a larger re-design. In addition, in many cases in order to make the performance reasonable, there is a desire to use the Poseidon hash function whose security is not yet as well established as other hash functions (like the SHA family). Thus, simply moving directly to post-quantum signatures today does not seem to be an optimal strategy. Fortunately, as we will discuss below, there are other alternatives.

4.3: Strategies and Security for Post-Quantum Signing

Before describing additional strategies, we will enumerate a number of properties that are desired; not all strategies will fulfill them all, **but this enumeration will help us to analyze and compare the different strategies:**

P1:
The transition does not compromise our current security posture, in the sense that any attacker who can break the new scheme can already break the existing scheme.
(This means that the "increase" in security is not subjective but mathematically guaranteed.)

P2:
The new scheme provides post-quantum security. This has two variants:

P2a: the new scheme is post-quantum secure as-is.

P2b: the new scheme enables a fast switch to post-quantum security; all that is required is a decision to operate in a different way that is already deployed.

P3:
The new scheme does not add significant cost to the current way of working, as long as no quantum threat is imminent.

P4:
The new scheme requires minimal (if any) changes to the blockchain and current way of working, as long as no quantum threat is imminent.

Note that not all properties are equal. P1 and P2 are critical; within P2 property P2a is stronger, but P2b seems sufficient at this time. P3 and P4 are more subjective, since the notions of "significant cost" and "minimal changes" are not mathematically well defined.

Before proceeding to the new strategies, we analyze the strategy discussed above of switching to ML-DSA of FN-DSA, **using the above properties:**

*The switch to SLH-DSA is not practical as a direct replacement without additional changes, so is not considered here.

P1:
This property is not fulfilled

P2:
This property is fulfilled according to the stronger variant P2a

P3:
This property is not fulfilled

P4:
This property is not fulfilled

We now proceed to consider three additional strategies, and then provide our recommendation.

Strategy 1 –

Generate private keys as hash outputs:

Cryptographic hash functions are basically post-quantum secure, in the sense that increasing the output size is sufficient. This is not *necessarily* always the case, but is assumed to be true for standard cryptographic hash functions. This means that the task of finding x given y=H(x) for an appropriate hash function H is a hard problem even for a quantum computer (assuming that the output of H is long enough; for this purpose, 256 bits suffices). Now, instead of choosing ECDSA keys by just choosing a random private key x and then computing the public key to be Q=x . G, consider the strategy of choosing a random x, and then setting the elliptic-curve private key to be y=H(x) and the public key to be Q=y . G. Clearly, this makes no difference, and indeed in many cases keys – and in particular, all keys generated via mnemonics like those described in BIP-39 – are already generated in this way. Thus, no change is needed to the blockchain at all, and everything can continue operating as today, with the only change being how the private key is chosen. Note that EdDSA keys are already generated in this way, as prescribed in the standard.

How does this help achieve post-quantum security? The idea is that if a quantum threat becomes realistic, then it is possible to continue using the same addresses and keys, but instead of signing using ECDSA or EdDSA, a signature can be constructed based on the owner's knowledge of the preimage of the private key, according to H. Technically, given an address which is *Q or H'(Q)* for some hash function H' depending on the blockchain, the signature can be based on a ZK proof that the owner knows x such that *Q=H(x) . G*. Importantly, we know how to construct secure signatures from such ZK proofs, and we know how to construct these to be post-quantum secure. Thus, all that is needed is to stop accepting the original ECDSA or EdDSA signatures, and to only accept the post-quantum ZK-based signatures from that point on. This has the extremely attractive benefit that almost nothing is needed today, beyond changing how keys are generated and building the software stack to accept these new post-quantum signatures (but without using that stack now). Furthermore, many existing wallets would need to change nothing, since they already use mnemonics and so have the needed preimage. We remark that a variant of the above can also be used for keys who were generated in a regular way, without knowing the preimage of Q, as long as only a hash H'(Q) of the public key has been revealed on the blockchain. In this case, it is possible to provide a ZK proof that the owner knows *x* such that address *=H'(Q) where Q=x . G*.

Strategy 1

Regarding the properties described above, we have the following:

P1:

This property is fulfilled (since the blockchain currently continues to use ECDSA or EdDSA as before). *Note that once the blockchain enables ZK-proof based signatures, then this depends on the assumptions used for constructing that ZK proof.*

P2:

This property is fulfilled according to the weaker variant P2b (since post-quantum security is only achieved when the blockchain declares that only the new signature type will be accepted)

P3:

This property is fulfilled

P4:

This property is fulfilled (the only change required is how keys are generated, and adding the ability to verify post-quantum ZK-based signatures)

This therefore looks like an ideal solution, and is even being deployed today **(see http://soundness.xyz)** However, it does suffer from a major disadvantage in that such signatures will likely be very expensive, and so this would not be a good long-term solution in a post-quantum world. There is also the danger that people won't move their assets even if they don't have a preimage, because everything continues as before. However, this opens up a much bigger discussion regarding individual responsibility, which is beyond the scope of this document.

We remark also that as long as the public keys remain hashed (i.e., when operating with one-use keys on Bitcoin), property P2a *is* achieved. This is because a quantum computer *is* unable to break elliptic-curve signatures without seeing the public key.

Strategy 2 –

Move to 2-out-of-2 hybrid/double signing: Multisig is a method that enables parties to define multiple addresses together with a threshold to be associated with an asset, with the result being that a transaction is only valid if a threshold of valid signatures with those addresses are provided. Multisig has been a part of Bitcoin since 2012, and so is a well-known paradigm. This strategy works by adding a post-quantum signature scheme, and requiring that every transaction includes both an ECDSA/EdDSA signature as well as a post-quantum signature (e.g., ML-DSA). This improves over the strategy of just moving over now to a post-quantum scheme, since security is strictly improved. In particular, the new scheme can only be broken if both the elliptic-curve scheme is broken *and* the post-quantum signature scheme is broken.

Regarding the properties described above, we have the following:

P1:

This property is fulfilled (since breaking the transaction signing requires also breaking the existing elliptic-curve scheme in use)

P2:

This property is fulfilled according to the stronger variant P2a (since all transactions requires post-quantum signing from day one)

P3:

This property is not fulfilled (the cost of post-quantum signatures is incurred from day one)

P4:

This depends on the blockchain. In blockchains that natively support smart contracts (like Ethereum), this is fulfilled. In blockchains like Bitcoin that do not, this requires a larger change since support for a new signature scheme needs to be added. However, even for Bitcoin, it is not a radical change, but just the addition of a new signature scheme and a change regarding the policy of all signatures being 2-of-2 of this type.

This strategy is therefore ideal for properties P1 and P2, but problematic for P3 and P4 (regarding P4, it depends on the blockchain).

Strategy 3 –

Move to 1-out-of-2 (or more) signing:
This strategy is very similar to the previous one, except that instead of requiring *both* signatures, it suffices to provide a signature either using the elliptic-curve scheme or the post-quantum scheme. If the quantum threat becomes realistic, then the blockchain simply needs to stop accepting elliptic-curve signatures, and only accept the post-quantum signature. This has a major advantage that no additional cost is incurred today. Note that if implemented naively, property P1 may not be fulfilled, since a break in either of the signature schemes suffices to generate a fraudulent transaction. This can be remedied by including only the hash of each public key, and only revealing the public key of the scheme that is used. This enables the use of elliptic-curve signing now, and switching to post-quantum signing when needed, and only revealing those public keys as needed after the switch takes place.

Regarding the properties described above, we have the following:

P1:
This property is fulfilled (by not revealing the actual post-quantum public key, the only way to break the scheme is to break the elliptic-curve signing)

P2:
This property is fulfilled according to the weaker variant P2b (since breaking elliptic-curve cryptography suffices until the blockchain switches)

P3:
This property is fulfilled (elliptic-curve signing continues as today)

P4:
As in strategy 2, this depends on the blockchain

We remark that this is essentially what is proposed in BIP-360 for Bitcoin. BIP-360 is more flexible than what we described here, but it achieves the same effect. It is also designed to utilize existing Bitcoin methods so that the change is not major.

In the same way as for strategy 1, as long as the public keys remain hashed (i.e., when operating with one-use keys), property P2a is achieved.

Summary Table:

We summarize the 4 strategies (including the naive one) and the properties achieved in the following table

	Plug-and-Play Replacement	Private Keys as Hash Outputs	Move to 2-of-2	Move to 1-of-2
P1 (current security)	NO	YES	YES	YES
P2 (PQ security)	YES (P2a)	YES (P2b)	YES (P2a)	YES (P2b)
P3 (cost now)	NO	YES	NO	YES
P4 (change)	NO	YES	DEPENDS	DEPENDS
NOTES:		PQ signing is not efficient so not a long-term solution. P2a is achieved as long as only hashed addresses are revealed.		Essentially BIP-360 for Bitcoin. P2a is achieved as long as only hashed addresses are revealed.

4.4: Recommendations – Execution Layer

There is no definitive correct answer to what is best. This is because there are incomparable tradeoffs, like whether it is better to incur cost and get higher security from day one or not.

More significantly, it depends very much on each blockchain and its existing design. For example, since Ethereum is smart-contract based, it can easily support a major change to how signing works at the execution layer, without making any governance changes at all. In fact, each smart contract owner can decide on its own strategy. It is therefore less important for Ethereum to make changes right now to the execution layer (unlike the consensus layer which does require change), but it may be desired in order to present a fully post-quantum solution. In contrast, any change to Bitcoin takes significant time, and requires a change to how miners process transactions and work. As such, solutions for Bitcoin need to have minimal changes, and need to be considered sooner. Indeed, BIP-360 is an important step in this direction.

Having said all of the above, we recommend the "move to 1-of-2" strategy. This strikes an excellent balance of preparing for the post-quantum threat while not paying any additional cost until needed. We stress that the strategy as described is a high-level template, and it can be implemented in many different ways; our recommendation is to adopt an approach that is based on this (or an equivalent) idea.

Irrespective of the strategy chosen, following any PQ upgrade (with the exception of the first strategy for keys that were already generated in this way), account holders need to transfer their balances to new accounts protected by PQ signature schemes. Such transfers become finalized by the underlying consensus engine. Therefore, these transfers must also be secured by a checkpoint or an upgrade to the PQ validator signatures; otherwise, they would be at risk from a long-range attack (as discussed above regarding the consensus layer). Orchestrating owners' key switching presents several headaches in itself. First, there are millions of owned accounts/UTXOs, and at the current transaction rates of blockchains like Bitcoin and Ethereum, it may take months just to commit the sheer volume of switchover transactions. The period over which users will have to transition before they are deactivated will most likely be considerably long (even years). Second, there may inevitably remain accounts whose owners will not participate in the switchover. Two possibilities for handling vulnerable accounts are leaving them as a sitting honeypot to be hacked or burning their coin balances. We discuss this specific dilemma in more detail below.

4.5: Crypto-Agility

Crypto-agility refers to the level of ease or difficulty it is possible to switch cryptographic algorithms in an application.

In general, the more the application depends on the specifics of the algorithm – e.g., if there is strong dependence on the size of the key and ciphertext – the harder it is to switch to a different cryptographic scheme. Crypto-agility is a highly recommended practice in general, due to the fact that cryptographic schemes are sometimes broken (e.g., MD5 and SHA-1) and sometimes key sizes need to be updated (e.g., RSA-1024 to RSA-2048 and higher). However, in the context of PQC this is even more important since cryptosystems that are post-quantum secure are significantly newer. Although they have undergone considerable review, they are not as well understood as primitives like RSA and ECC, and the required key lengths may change over time. As such, our strong recommendation is that blockchains should include agility planning so that if a cryptographic algorithm needs to be replaced, then it can be replaced quickly and without significant disruption.

4.6: The Cost of PQC at the Execution Layer

Making the execution layer post-quantum secure requires replacing existing signature schemes and providing users with mechanisms to migrate their keys. **There are two principal challenges.**

The first is orchestrating the transition without users losing access to their assets. Crucially, a switch to post-quantum signatures requires owners to actively migrate their coins, and the system must also deal with inevitably abandoned accounts that are not upgraded. The second challenge is performance. Here, the key concerns are the size of blocks carrying transaction data and the cost of verification. Each block carries multiple (signed) transactions – from e.g., ~200 on Ethereum up to thousands on Algorand. Existing blockchains bound blocks by size or storage gas-cost, not the number of transactions, because transactions can vary vastly in size and computation time. Even tens of additional bytes per input compound into large bandwidth, fee, and storage costs at scale.

What is the impact of transaction signature size on communication and storage?

A typical ECDSA signature is 64 bytes in raw form (or about 71 bytes on average when DER-encoded). Suppose a 1 MB block carries 60% signatures (a hypothetical number), corresponding to roughly 14,000 ECDSA signatures. At 144 blocks per day, this implies storing about 86 MB of signature data per day, or roughly 31 GB per year, in signatures alone. Replacing these with PQ signatures—for example, ML-DSA with a 2,420-byte signature—would increase the block size by roughly 38x for the same number of transactions, yielding blocks of approximately 38 MB. At the same rate, this would require storing over 1 TB of signature data per year.

However, the precise impact of increased signature and public-key sizes on transaction rates, throughput, and overall blockchain storage and cost is not straightforward to derive. For example, since SegWit (BIP141, 2017), Bitcoin no longer measures block size purely in bytes. Instead, it uses a weight system in which non-witness data counts four times more than witness data (the public key and signature are considered witness data), with a maximum of 4,000,000 weight units per block. A virtual byte (vB) is defined as four weight units, yielding an effective limit of 1,000,000 vbytes per block. As discussed above, replacing an ECDSA signature (71 bytes DER) with an ML-DSA-44 signature (2,420 bytes) increases the signature size by roughly 38x (similar increase factor of public key size). However, because witness data is discounted by a factor of four, the impact on vbytes is smaller. In recent years, on average Bitcoin signatures and pubkeys make up 50%-66% of raw transaction sizes in bytes. Measured in vbytes, the PQ blowup factor is therefore only about .5+.5×(38/4) ≈ 5 to .33+ .66×(38/4) ≈ 7. By this back-of-the-envelope calculation, the number of PQ transactions that fit in a block, and thus throughput, would decrease by roughly 5-7x, not 38x.

As another point of comparison, Ethereum uses only one signature per transaction, but blocks are constrained by gas rather than by size. However, the computational cost of PQ primitives is not yet well understood, making it difficult to translate increased signature size directly into throughput and cost projections.

From all of the above, it is clear that if used naively, replacing transaction signatures with PQ-signatures would reduce blockchain throughput by one to two orders of magnitude, massively increase fees, or cause explosive chain growth. Therefore, in the post-quantum world signature size becomes a first-order architectural constraint.

Signatures are not a new tax on blockchains, and indeed, several existing scaling approaches mitigate the storage and communication costs. These include pruning the history of blockchains and deferring archival responsibility to a dedicated infrastructure, data-availability services that offset validator communication and (short-term) storage costs, and rollups bundled with off-chain data-availability that remove the need for validators to process transactions. These techniques and others may be considered as part of a migration strategy to PQ blockchains, in addition to PQC cryptography per se.

4.7: Recommendations – PQC Key Management

The discussion above regarding PQC at the execution layer has focused on the blockchain challenges. However, when considering the execution layer, it is crucial to also consider the impact on the users who interact with the blockchain via the execution layer. In particular, the way that signing keys are managed and protected, whether it be a user wallet or an exchange/custodian's key-management system (KMS), must be updated to support the post-quantum signature scheme that the relevant blockchains transition to.

This transition can raise considerable challenges. For example, if the wallet/KMS in question is hardware-based, then the hardware needs to be updated to support the new signature scheme. In particular, adding new algorithms to special-purpose or general-purpose HSMs takes time. Likewise, threshold-signature (MPC) based key-management needs to be updated to support the new scheme. However, not all signing schemes have efficient MPC protocols that are known. Furthermore, not all signature schemes seem amenable to efficient MPC. For example, ML-DSA appears to be amenable to efficient MPC, and some protocols have already been published. However, it is unlikely that an efficient MPC protocol will ever be possible for SLH-DSA. To compound the above challenges, different blockchains may adopt different PQ signature schemes, and most of the major blockchains have not yet committed to exactly what algorithm will be used. Thus, it isn't possible to begin fully preparing yet for the transition.

Based on the above, our recommendation is for organizations providing KMS services (whether they be in the form of a hardware device, user software, or via an exchange or custodian) to begin the process of understanding what is required of them to transition to different algorithms, and to have clear plans of how the transition will be carried out. This will enable them to respond in a timely manner to blockchain transitions to PQC, when they occur.

5

Post-Quantum Plans for Major Blockchains

Every blockchain will need to make some adjustments in order to become post-quantum secure. In this section we look at some proposals for the Bitcoin, Ethereum, Solana, Algorand, Sui, and Aptos chains. Making these chains post-quantum secure will require updating both consensus node software and user wallet software. It will also require user action.

5.1: Bitcoin

Bitcoin assets are held in objects called UTXOs (unspent transaction outputs). Transferring the assets in a UTXO often requires a signed transaction, although some UTXOs can be spent by other means. Bitcoin natively supports the ECDSA and Schnorr signature schemes for signing transactions. In some UTXOs only the hash of the public key that is needed to spend the UTXO is available onchain, while the cleartext public key remains hidden. Such UTXOs are safe from a direct quantum attack on the public key. However, it is important to keep in mind that even such UTXOs may be at risk.

- **First,** the cleartext public key may be exposed by offchain data, such as a payment channel, and this enables a quantum attack on these UTXOs.

- **Second,** when a UTXO is spent, the spending transaction reveals the public key in the clear. Since the spending transaction lives in the mempool for several minutes before it is mined into a block, a quantum adversary theoretically has several minutes to break the key and issue a competing transaction that steals the UTXO assets. A quantum computer that can break a key in this short amount of time seems further away than one that can do so in days or weeks. Nevertheless, the Bitcoin community is exploring a commit/reveal approach to enable safe spending of a pre-quantum UTXO whose public key is not available on chain in the clear. While this requires a small change to how miners operate, it provides a relatively safe way to spend a pre-quantum UTXO.

For some UTXOs, the relevant public key is available in the clear onchain, and such UTXOs are potentially vulnerable to a quantum attack by simply harvesting onchain public keys, and using a quantum computer to recover the corresponding signing keys. At the time of this writing, Project 11 estimates that about 6.9M BTC are currently held in UTXOs for which the cleartext public key is known.

- **About 1.7M of these BTC are held in old Pay-to-Public-Key (P2PK) UTXOs, where the public key is available on-chain unhashed.** Some of these are called "Satoshi coins" and represent mining rewards from the early days of Bitcoin. Each Satoshi UTXO holds 50 BTC, and in aggregate they are controlled by about 20,000 different public keys. Since many of these are old UTXOs it is possible that a sizable fraction of these coins are abandoned, meaning that the spending private key is no longer known. Such abandoned coins cannot be moved to a post-quantum address, until a large quantum computer becomes available (or a classical attack on ECDSA is discovered). We discuss what to do about abandoned coins – a question that affects all chains, not just Bitcoin – at the end of this section.

- **About 1M of the 6.9M exposed BTC are held in only eleven different addresses.** If nothing changes, then running a future quantum computer on these eleven public keys could result in the theft of about 1M BTC. Since each of these whale addresses holds way over a 1000 BTC at present, addresses that hold fewer than 1000 BTC will likely not be the first target for an attacker. Hence, these eleven addresses serve as a good canary for when a large quantum computer comes online (unless of course they migrate beforehand).

How will Bitcoin become post-quantum secure? Bitcoin's current approach is to ensure that all UTXO public keys can be hidden behind a hash function. While this is supported by the Pay-to-Public-Key-Hash (P2PKH) output type, the popular Pay-to-Taproot (P2TR) output type contains an onchain cleartext ECDSA key that can be used to spend the taproot output. **Consequently, all P2TR UTXOs on the Bitcoin network are currently quantum vulnerable.** To mitigate this issue, the BIP-360 proposal introduces a new taproot output type called Pay-to-Merkle-Root (P2MR) that removes this public key altogether. Once this proposal is enabled on Bitcoin mainnet, transitioning a P2TR output to a P2MR output will remove this vulnerability.

Beyond these modest proposals for securing assets behind a hashed public key, the Bitcoin core devs are taking a wait-and-see approach. One reason to wait is the possibility that a better post-quantum signature scheme will be developed by the research community. In addition, some core devs are exploring hash-based signatures for Bitcoin. These signature schemes are attractive because they can be implemented without introducing new cryptographic assumptions in the Bitcoin network. The downside is that these signatures are much larger than ECDSA signatures, and without additional techniques, this may reduce the Bitcoin transaction rate. We remark that the wait-and-see approach has a price in that it causes market uncertainty. Thus, waiting for the exact migration plan can make sense, but it should come with a clear statement of strategy and preparation to enable speedy migration if needed.

The transition period.

Once Bitcoin adopts a post-quantum solution, all vulnerable UTXOs will need to be spent and transitioned to a post-quantum address. Project 11 estimates that there are currently about 13.6M exposed addresses (i.e., addresses for which the public key is available in the clear on chain). Given the low Bitcoin transaction rate, the transition will take, at a minimum, several months to complete.

The impact on proof-of-work.

So far we discussed the impact of a quantum computer on Bitcoin UTXOs. A quantum computer may also impact proof-of-work consensus via Grover's algorithm, a quantum algorithm that, in theory, provides a quadratic speed up for all search problems. A quantum computer running Grover's algorithm could, in theory, solve the proof-of-work challenge faster than a classical computer. Thus, a Bitcoin miner employing a quantum computer could collect all the Bitcoin mining rewards and centralize all block production. However, this is unlikely. For the size of proof-of-work challenges, the overhead of running Grover's algorithm on a quantum computer is so high that it dominates the quadratic speed up. As such, we expect that for many years to come, a classical Bitcoin miner will always outperform a quantum Bitcoin miner.

5.2: Ethereum

The Ethereum network currently has four components that can be impacted by a quantum computer:

1. **At the execution layer (EL) externally owned accounts, or EOAs, are vulnerable:** transactions that spend EOA assets are signed using ECDSA, where the public key can be derived from data stored on chain, if the EOA has made at least one outgoing transaction. A quantum attacker can break this public key and issue transactions on behalf of the EOA owner.
2. **At the consensus layer (CL)** validators attest to new blocks using the BLS signature scheme, where the public key is available on chain. A quantum attacker can break this public key and attest to invalid blocks on behalf of the validator.
3. **The EVM provides precompiles for verifying pairing-based proofs,** such as Groth16 proofs. A quantum attacker can break the structured reference string (SRS) of the proof system and construct a valid proof for a false statement.
4. **Finally, the data availability scheme used in the data layer (DL) is based on a pairing-based polynomial commitment scheme called KZG.** A quantum attacker can break the KZG structured reference string (SRS) and cause unavailable data to appear as if it were available.

The Ethereum community published a detailed plan for mitigating each of these four issues. This is part of a major upgrade to Ethereum that will also improve the network's transaction rate. Let us first review the plans for a post-quantum signature scheme in Ethereum.

Post-quantum signatures for Ethereum.

Post-quantum signatures are needed at both the consensus layer, for attesting to blocks, and at the execution layer, for signing transactions. The current plan is to transition to hash-based signatures for both the consensus and execution layers. The primary benefit of hash-based signatures is that their security is based solely on the security of the underlying hash function. If a standard cryptographic hash function is used, then this does not introduce new security assumptions to Ethereum. The downside of hash-based signatures is that the signatures tend to be relatively long, but this can be resolved using SNARK-based signature aggregation, as discussed further below.

There are two flavors of hash-based signatures: stateless and stateful. Briefly, in a stateful signature scheme the signer maintains some state that is updated every time a new signature is issued, and it is paramount that the same state is never reused to sign two different messages. The benefit of a stateful signature scheme is that signatures tend to be shorter than for a stateless scheme.

A. Stateful hash-based signatures are a good fit for the consensus layer (CL), because the block height can be used as the non repeating state. Indeed, validators should not sign two different blocks at the same height, since that may trigger a slashing event. The signature scheme proposed for this is called **leanXMSS.**

B. Stateless signatures are a good fit for the execution layer (EL), so that account owners are protected from mistakes in state management. The signature scheme proposed for this is called **leanSPHINCS.** It is a variant of the SPHINCS+ standard from NIST that has a shorter signature and is more amenable to aggregation, as discussed next.

The need for signature aggregation.

Both the consensus and execution layers generate many signatures per block. To save space and verification time there is a strong desire to aggregate many signatures into a short string that can be quickly verified, and if valid, proves that all the aggregated signatures are valid. The plan is to use a succinct proof system (SNARK) to generate a short proof that is fast to verify, where this short proof convinces the verifier that all the provided messages were property signed by the intended signer.

Thanks to this SNARK-based signature aggregation, the size of an individual signature is not a major concern. The only data posted onchain is a single aggregate signature, on the order of 128KB. At the execution layer, signature aggregation will be done by the block builder as part of zkEVM proving. Block builders are incentivised by transaction fees to perform aggregation. Since the aggregate signature is subsumed by zkEVM proofs, there is no data overhead. Moving to post-quantum signatures is actually a scalability increase for Ethereum L1, thanks to the removal of the per-transaction ECDSA data overhead.

At the consensus layer signature aggregation will be done by a subset of opted-in (non-designated) validator nodes with sufficient hardware resources. Those aggregators are spread across subnets for parallel aggregation, with multiple (e.g. 2 or 3) rounds of successive meta-aggregation (i.e. aggregation of aggregates). A validator on a high-end laptop CPU can aggregate roughly 1,000 leanXMSS per second. Similar to BLS aggregation, leanXMSS is altruistic by virtue of being cheap enough and being default-on for validators with sufficient hardware resources.

The proof system for signature aggregation will be built on the **leanVM**, a minimal zkVM for general-purpose hash-based cryptography. The underlying proof system is itself hash-based and the proof is about a hundred kilobytes. This is much bigger than a BLS aggregate signature, which is less than a hundred bytes, and hence this transition incurs some increase to the block size. We stress that to accelerate proof generation, the hash-based signature schemes mentioned above will likely use a non-standard SNARK-friendly hash function called Poseidon (designed in 2019), whose security is the focus of the **Poseidon Initiative** as well as the $1M Poseidon Prize **(launching soon).**

With this in place, let us briefly discuss the mitigations being considered for the four issues listed at the beginning of this section.

Eliminating ECDSA from the execution layer.

The goal is to enable account abstraction for every EOA so that it is controlled by a smart contract wallet (see **EIP 8141** for the latest proposal). This means that the EOA will be controlled by code that decides what is a valid transaction, and this code can implement whatever authentication mechanism the user wants, including a hash-based or even a lattice-based signature scheme such as ML-DSA. Towards this goal, **EIP 8051** proposes to add ML-DSA verification as an EVM precompile. The benefit of this approach is that users can keep their existing addresses, without moving all their assets to a new post-quantum address. We stress that this transition requires user action: the user must sign a single transaction giving the smart contract wallet control over the user's EOA. Moreover, for security, the chain must stop accepting ECDSA signatures for such an EOA, possibly as outlined in **EIP 7851.**

A post-quantum consensus layer.

This will be done by having validators attest to each block using a stateful hash-based signature scheme, and all the attestations on a specific block will be aggregated into a single proof using a hash-based succinct proof system.

Mitigating the last two issues listed at the beginning of this section is a bit more technical. First, the recent **EIP 8141** proposal includes an approach for replacing the EVM precompiles for validating a Groth16 proof with a post-quantum proof validation. Second, post-quantum updates to the data availability layer are more complex and will be discussed elsewhere.

5.3: Solana

Solana created a new vault type, called the Solana Winternitz Vault. The Winternitz signature scheme is a hash-based signature scheme that has a manageable signature size, although signatures are two orders of magnitude bigger than ECDSA signatures. Solana token holders can move their assets to a new Winternitz-based address. Once the move is complete, the assets are no longer exposed to a quantum attacker.

5.4: Algorand

Algorand is among the first blockchain platforms to deploy post-quantum (PQ) signature schemes in production across both consensus-related mechanisms and the execution layer, following a staged roadmap toward full quantum readiness.

At the consensus level, Algorand's State Proof framework employs NIST-selected FALCON signatures (to be standardized as FN-DSA in FIPS 206) to produce quantum-resistant attestations of blockchain state by compressing approximately 256 rounds of block headers into succinct certificates verifiable by light clients and external chains. This mechanism secures the historical integrity of the ledger against future quantum attacks. However, core consensus operations remain partially reliant on classical cryptography: block proposals and committee voting use Ed25519 signatures, and the verifiable random functions (VRFs) used for sortition-based committee selection remain vulnerable to quantum attacks. Algorand has acknowledged these limitations and is actively researching approaches to progressively secure its consensus core.

At the transaction and execution layers, Algorand already provides the cryptographic tools necessary to support quantum-resistant accounts. The network **recently executed its first post-quantum transaction on mainnet using FN-DSA**, integrating FN-DSA verification as a native virtual-machine primitive. Through logic signatures that verify FN-DSA signatures over transaction identifiers, users can create quantum-resistant accounts without requiring protocol modifications.

5.5: Sui

Sui has outlined a number of strategies for migrating to a post-quantum secure chain. It is not yet clear which of these strategies will be deployed.

5.6: Aptos

Aptos is well positioned for the transition to post-quantum secure transactions. In Aptos a user's address is not derived from the hash of the user's public key. Instead, the user's public key (called an authentication key) is stored as metadata associated with the user's account. Thus, users who want to become post-quantum secure need only sign a transaction that updates their authentication key to a post-quantum public key. There is no need to move assets to a new account.

In AIP-137, Aptos outlined their plans to add support for SLH-DSA-SHA2-128 signatures and public keys. These are hash-based signatures. Once this AIP is implemented and deployed, Aptos users can simply update the authentication key associated with their account. No other action is required.

5.7: Layer 2 chains

Some layer 2 chains, such as Optimism, Arbitrum, Base, and others, have also announced a post-quantum plan.

For example, [Optimism announced that their post-quantum plan for the superchain](#) is a transition from EOA accounts to smart contract wallets. These smart contract wallets can be secured by a post-quantum public key, and they can manage assets stored in the EOA account using the EIP-7702 delegation mechanism. As a result, users need not move their assets away from the EOA address.

However, the ECDSA key controlling the EOA account needs to be deprecated, and to do so Optimism has announced a flag day, planned for January 2036, after which the ECDSA signing key will no longer be able to control EOA assets. After that date, the assets in the EOA account can only be controlled by a smart contract wallet.

5.8: What to do about abandoned assets?

Every chain that is planning to become post-quantum secure faces a critical decision:

- **Option (1): announce a "flag date" after which all assets that are protected by a quantum vulnerable public key will be revoked, and the assets will be lost forever.**
- **Option (2): no "flag day" meaning that assets that do not move to a post-quantum public key will remain active forever. These assets may be at risk of being stolen by a quantum attacker at some point in the future.**

In the case of Bitcoin there is a third option to handle (presumably) abandoned UTXOs whose public key is exposed on chain, most notably, the so called Satoshi coins discussed at the beginning of the section. Recall that there are about 1.7M such BTC held in P2PK outputs. Instead of revoking these UTXOs outright, the [Hourglass spending rule](#) proposal limits spending a P2PK output to at most 1 BTC per block. This way, an attacker who can break the P2PK public keys is greatly limited in the rate that they can spend these abandoned assets they now control. Moreover, any such transaction by the attacker will alert the community to an insecurity of ECDSA, thereby using the Satoshi coins as a canary for the existence of a quantum adversary, while limiting the negative consequences to Bitcoin.

Going back to options (1) and (2) above, there are good arguments for both options. Arguments for a flag day (i.e., canceling exposed assets before a quantum computer is built):

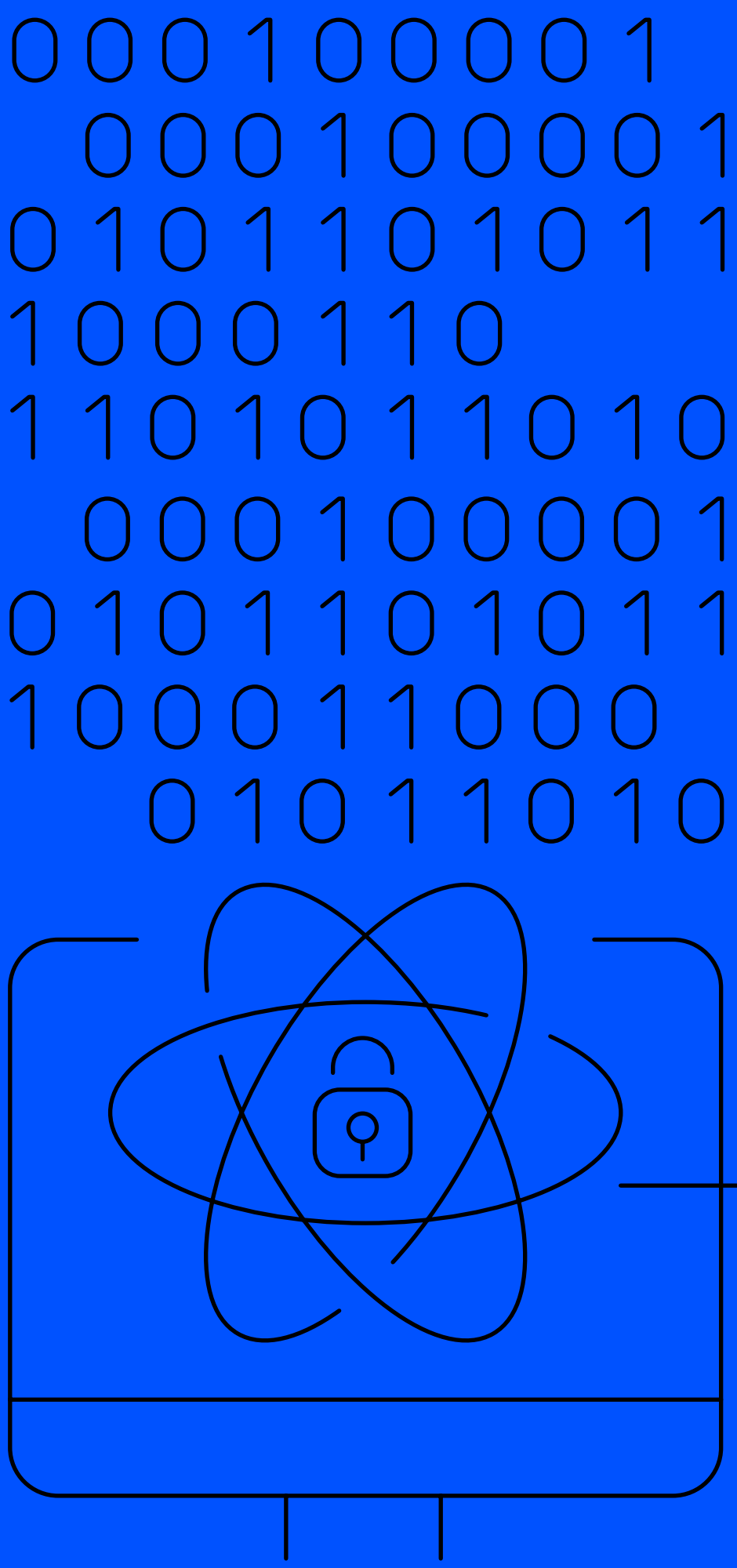
- **Without a flag day, a sanctioned entity or rogue state may be the first to develop a quantum computer, and use it to steal assets to fund its illicit operations.**
- **Without a flag day, once a quantum computer is built, it could be used to steal dormant exposed assets. This could greatly increase the supply of tokens, and potentially crash the token value.**

Arguments against a flag day (i.e., leaving exposed assets exposed)

- **No matter how much information will be published about the flag day, some people will be unaware of it, and will not move their assets to a post-quantum address. After the flag day, those people will lose their assets, leading to unhappy users and potential litigation.**
- **The exposed UTXOs function as an incentive to develop a quantum computer, which may have positive applications in other areas of human life, as discussed in the first section. They also function as a canary to inform us when a large scale quantum computer has been developed.**
- **A quantum computer will give people who lost their tokens a way to get their tokens back. For example, if a person or entity could provide convincing evidence that a particular Ethereum wallet belongs to them, then a company who has a capable quantum computer could help them recover their locked assets. Here the assumption is that the first capable quantum computer will be developed by an honest player. In fact, this may be one of the first commercial applications for a quantum computer.**

Chains that cannot reach consensus as to which of the two options to adopt may fork: one fork will implement option (1) and the other fork will implement option (2). Market forces will then decide which fork will have the greater value. Since the decision is mostly in the hands of token holders, they will likely favor the fork that implements option (1), canceling exposed tokens, because that option reduces the supply of tokens, making the tokens in the hands of token holders more valuable. Given this likely turn of events, we hope that chains can avoid the ordeal of forking the chain.

We encourage blockchain communities affected by this issue to recognize that market uncertainty is always a problem, and the uncertainty regarding abandoned assets is already causing damage. Thus, making a decision and communicating it publicly should be a priority for major blockchains.



6

Post-Quantum Security Beyond Signing

coinbase Independent Advisory Board on Quantum Computing and Blockchain

6.1: Post-Quantum Security Beyond Signing

In an earlier section we discussed the impact of quantum computing on transaction signing. In this section we look at other cryptographic primitives, beyond signatures, that will need to be replaced.

1 Threshold signatures.

Threshold signatures are used to protect signing keys throughout the blockchain ecosystem. Using a t-out-of-n signature scheme ensures that even if t-1 parties are compromised, the attacker learns nothing about the secret signing key. Yet any t parties can sign a message. A pre-quantum signature scheme such as BLS supports a very simple (one-round) threshold signing protocol. The Schnorr signature scheme supports a threshold signing protocol, but it requires **two** or **three** rounds of communication between the parties to sign a message.

There are a number of ways to support threshold signing in a post-quantum setting. The first option is to implement a threshold signing protocol for the ML-DSA signature scheme. Recall that ML-DSA is a lattice-based analogue of the Schnorr signature scheme. While threshold signing protocols for ML-DSA are more involved than for Schnorr, there already exist protocols (e.g., **this paper**) that is efficient and needs only three rounds of communication between the signing parties. Future research is expected to deepen our confidence in the security of existing protocols and to develop new protocols that further improve performance.

For applications that need a stronger security assurance, one can use a hash-based signature scheme such as either variant of SLH-DSA. These signature schemes are much harder to thresholdize directly: see for example **here** and **here.** Instead, one can apply a **generic technique** for converting a non-threshold signature scheme into a threshold scheme using a succinct proof system, also called a SNARK. Here, each of the n parties generates a public/private key pair on its own. Then all the public keys are placed at the leaves of a Merkle tree, and the resulting Merkle root becomes the public key for the scheme. Now, to sign a message m, some set of t parties sign on their own using their signing key. The final signature is a SNARK proof where the witness is the set of t signatures along with the Merkle proofs for the corresponding t public keys. In such a scheme the public key is short (a Merkle root), a signature is as short as a SNARK proof, and verification time is the same as the time to verify a SNARK proof. This scheme supports what is called a "silent" setup: there is no multi-round distributed key generation (DKG) protocol, and the set of n signing parties and the threshold t can be easily changed dynamically as needed. Moreover, this scheme can either operate as a private threshold signature scheme (where the final signature reveals nothing about the threshold t and the set of parties who signed) or as an accountable multisignature (where the final signature identifies the set of t parties that signed). This is the current plan for **post-quantum consensus signatures in Ethereum.** The downside is that signatures are relatively long --- as long as a post-quantum SNARK proof --- and that the signing process relies on a SNARK prover which is complex.

Recommendation:

The field of post-quantum threshold signature schemes is still evolving. If possible, it is best to wait until better schemes emerge. If one cannot wait, then SNARK-based threshold SLH-DSA is a reasonable choice.

2 Collision Resistant and one-way hash functions.

Collision resistant hash functions are used everywhere on the blockchain. From Merkle trees and Patricia trees to hash-based proof systems and signatures. Similarly, one-way hash functions are used in hash timelock contracts (HTLC).

Our current state of knowledge suggests that collision resistance is unaffected generically by a quantum computer. That is, a quantum computer is no better at finding collisions on a generic hash function than a classical computer. **However, there are two caveats:**

- **Not enough analysis has gone into understanding the quantum collision resistance of *specific* hash functions such as SHA256, SHA3, and Poseidon. While unlikely, there may be a clever quantum algorithm that can find a collision for these functions in less time than the time to find a collision for a generic function.**
- **There is some evidence that there may yet be a better *generic* quantum collision finder than what is currently known. However, this is currently an open problem. In particular, the known faster collision finders require a large number of qubits, making them impractical.**

One way hash functions, as used in HTLC contracts, are impacted by Grover's algorithm in theory. In practice, Grover's quadratic speedup does not translate to a real speedup. Nevertheless, for post-quantum security, the output size of these hash functions could be doubled to eliminate any concerns about an attack by Grover's algorithm.

Proof of work hash functions.
In theory, the quadratic speed-up provided by Grover's algorithm suggests that a quantum computer should be able to solve the Bitcoin proof of work faster than a classical computer. In practice, however, the overhead of Grover's algorithm dwarfs the quadratic speedup for real-world proof-of-work parameters, and no speed-up is obtained. Hence, with our current state of the art there is no change needed to PoW systems.

Succinct proof systems (a.k.a ZK proofs or SNARKs).
These proof systems are used in ZK-rollup systems and bridges. Generally speaking, there are three types of proof systems: **pairing-based, hash-based, and lattice-based.**

- **Pairing-based proof systems become unsound in the presence of a large-scale quantum computer and will need to be deprecated. In particular, the popular Groth16 proof system will need to be deprecated as part of the transition to post-quantum blockchains.**
- **Hash-based proof systems are largely unaffected by a quantum computer. However, these proof systems usually rely on the Fiat-Shamir transform, and the soundness of this transform is impacted by the theoretical quadratic speedup of Grover's algorithm. While this speedup may only be a theoretical concern, it is advisable that these proof systems shrink their soundness error to account for this risk.**

Recommendation:

ZK-rollup systems and ZK-bridges will need to prepare to transition away from Groth16. This is not an easy transition because of the much larger size of post-quantum proofs. Future research may yield better solutions, but as of now, experimentation with post-quantum proofs should begin.

3 Zero knowledge proof systems (a.k.a ZKZK proofs or zk-SNARKs).

These systems are used in privacy systems, such as private transaction systems (e.g., Aleo and Aztec) and in mixers (e.g., Railgun and PrivacyPools). The zero knowledge property of these systems is usually information theoretic and is therefore unaffected by a quantum computer.

4 Public key encryption and the TLS protocol.

TLS is a two-party transport layer security protocol and is the most widely deployed security protocol on the Internet. Pre-quantum TLS is at risk of an attack called harvest-now-decrypt-later (HNDL) where an attacker records valuable encrypted TLS sessions today, so that they can be decrypted in a few decades once a quantum computer becomes available. The same applies to public key encrypted data, where a ciphertext created today may be decrypted by a quantum computer in the future. This is a considerable risk for trade-secrets, health records, financial data, and government confidential data. The risk of an HNDL attack is the primary motivation for the US government's requirement that products sold after 2035 use post-quantum encryption. We stress that 2035 was chosen as a reasonable timeline for industry to complete the transition to post-quantum encryption. It does not indicate that one expects a crypto-relevant quantum computer to be available by 2035.

In the blockchain space TLS is used, for example, to protect transactions en-route to a block builder. Public key encryption is used to protect transactions on their way to an encrypted mempool. In these settings, transaction confidentiality is important for a relatively short period of time. Typically, the risk that these transactions will be decrypted a decade later may not be a strong concern. Nevertheless, post-quantum TLS is already widely deployed on the Internet (as of February 2026, **over 60% of Cloudflare's Internet traffic** uses the hybrid post-quantum secure cipher suite X25519MLKEM768). When supported, migration to post-quantum TLS is relatively straightforward. We therefore recommend following the best practices of Internet service providers such as Cloudflare and Google.

In some blockchains public-key encrypted data needs to remain hidden in perpetuity. For example, a number of privacy chains and protocols use public key encryption to protect the content of transactions. A quantum attack on the encryption scheme would expose transaction data that is meant to remain hidden forever. This may reveal the identity of payers, payees, and amounts, despite the advertised privacy assurances of the relevant chains. The specific damage caused by a quantum computer depends on how the system is architected. In the context of Zcash, we refer to **Sean Bowe's writeup** for a discussion of how a quantum computer affects the privacy of Zcash transactions. For other privacy chains, such as Aleo and Aztec, a quantum attack on the public key encryption scheme may also impact privacy.

We stress that due to the risk of a harvest-now-decrypt-later (HNDL) attack, the transition to a post-quantum public key encryption scheme is more urgent than the transition to post-quantum digital signatures.

5 Trusted Execution Environments (TEE).

(TEE). TEEs are used widely in the blockchain space (e.g., for block building, some layer-2 systems, and trusted oracles). Existing TEEs, such as Intel's TDX rely on RSA signatures for remote attestation. All hardware deployments that rely on such TEEs will need to be upgraded to post-quantum signatures. Other capabilities of the processor, such as signed microcode updates, also rely on RSA signatures and will need to be upgraded.

Position Paper

7: Additional Reading

There are many papers that have been written on quantum computing, cryptography and blockchains. Below we provide links to a few sources that we believe are informative for those seeking more information:

1

Links regarding quantum computing and post-quantum cryptography in general:

- State of the post-quantum Internet in 2025
- The quantum era is coming. Are we ready to secure it?
- Quantum is unimportant to post-quantum
- Quantum computing: quantifying the current state of the art to assess cybersecurity threats
- Tracking the Cost of Quantum Factoring
- Dora Research - Blog posts on quantum computing

2

Links regarding quantum computing and its impact to blockchain:

- Quantum computing and blockchains: Matching urgency to actual threats
- Quantum computing: What, when, where, how
- Hardware wallets: the post-quantum upgrade problem

3

Academic papers on quantum and blockchain:

- Quantum Disruption: An SOK of How Post-Quantum Attackers Reshape Blockchain Security and Performance
- Assessing the Impact of Post-Quantum Digital Signature Algorithms on Blockchains
- Hash-based Signature Schemes for Bitcoin
- Hash-Based Multi-Signatures for Post-Quantum Ethereum

4

Recent research on size/time estimates:

- Shor's algorithm is possible with as few as 10,000 reconfigurable atomic qubits
- Securing Elliptic Curve Cryptocurrencies against Quantum Vulnerabilities: Resource Estimates and Mitigations