MARIN IVEZIC

QUANTUM COMPUTING FOR CYBERSECURITY PROFESSIONALS

No Physics Degree. No Hype. No Lies-to-Children.

Quantum Computing for Cybersecurity Professionals

No Physics Degree. No Hype. No Lies-to-Children.

Marin Ivezic

PostQuantum.com

Published by PostQuantum.com

Version 2.0, July 2026

This ebook is the companion edition of the 11-part online series *Quantum Computing for Cybersecurity Professionals*. The latest version of every chapter, along with updates as the field moves, lives at:

https://postquantum.com/quantum-computing-for-cybersecurity/

Resource estimates, hardware figures, and standards status throughout this book were reviewed on July 12, 2026. Quantum computing moves quickly; where a number matters to a decision, verify it against the online edition.

How to read this book

This book assumes you know classical cryptography, hashing, XOR, and big-O notation, and assumes nothing about physics. No linear algebra appears anywhere. Every quantum claim is built from arithmetic you can check by hand, and the few places where checking requires effort come with the numbers laid out.

Five kinds of boxes recur throughout, each with its own color:

THE ARITHMETIC

The arithmetic

Hand-checkable math. When a chapter claims two amplitudes cancel, this box is where you verify it with nothing harder than addition.

WHERE THIS BREAKS

Where this breaks

Every analogy in this book ships with a note on where it stops being true. Read these; the failure modes of an analogy are usually where the interesting physics lives.

FOR THE RECORD

For the record

Exact numbers, sources, and precision notes behind a claim made more loosely in the running text.

MYTH AUTOPSY

Myth autopsy

A popular claim, dismantled at the exact moment you have acquired the tools to see through it yourself.

ATTACKER'S-EYE VIEW

Attacker's-eye view

What the physics in the chapter means for someone trying to break your systems, and what it means for you defending them.

Contents

What a Quantum Computer Is Not

Sometime this week, three items will cross your feed. A lab will announce a quantum processor with a record number of qubits (physical qubits, a qualifier that will matter a great deal by Chapter 10). An article will explain that quantum computers work by trying every combination at once. And a colleague will forward both with the subject line "should we be worried about AES?"

Of the two factual claims in that pile, exactly one is accurate. The qubit count. Your colleague's email is the right question aimed at the wrong target, and by Chapter 9 you will know why.

This series is for the person who has to answer that email, and its bottom line is this: quantum computers threaten exactly one category of cryptography, your deadline is set by your data's lifetime rather than by anyone's prediction, and "a faster computer" is the wrong mental model entirely. Leadership, auditors, customers, and eventually regulators are going to ask you questions about quantum computers whose correct answers depend on details the popular story gets backwards. So the next eleven parts come with a contract. No lies-to-children: nothing you learn here will need to be unlearned later. Every analogy ships with a note on where it breaks. Any claim that can be checked with arithmetic gets an optional sidebar where you can check it, using math no harder than working out a subnet mask. And at no point will you be asked to accept "quantum weirdness" as an explanation for anything.

The wrong axis

Start by deleting the most common frame: that a quantum computer is a faster computer.

Your laptop, a rack of GPUs, and the largest supercomputer on earth are all the same kind of machine at different sizes. Anything one of them can compute, the others can too, given enough time; they differ only in scale. A quantum computer sits off that ladder entirely. It runs on different physical rules, and those rules make it dramatically better at a narrow class of problems with exploitable mathematical structure, while leaving it unremarkable to useless at nearly everything else. It will not run your SIEM faster, sort your logs, train your models on a budget, or mine Bitcoin at a profit. Nobody will ever issue one to the help desk.

> FOR THE RECORD
>
> **Tractable versus computable.**
>
> A classical computer can simulate a quantum computer perfectly; the simulation just takes exponentially long as the quantum machine grows. So quantum computing moves certain problems from "possible in principle, absurd in practice" into practical reach, and it adds nothing to the set of problems that can be solved at all. This is also why the casualty list below is decided by the mathematical structure of each algorithm, never by raw speed.

The right axis

Swap in a frame your profession already owns.

There are two ways to defeat a cipher. The first is to buy more hardware and try keys faster. That attack is quantitative: double the budget, halve the time, and the defender answers by adding a single bit to the key, staying comfortably ahead forever. The second way is to find structure in the underlying mathematics that lets you bypass key-trying entirely. That attack is qualitative. Cryptographers call it a break, no key length rescues a broken primitive, and the only remedy is retirement.

Quantum computers, in the cases where they matter at all, belong to the second category. Shor's algorithm, the reason this series

exists and the subject of Chapter 8, is a cryptanalytic break of the mathematical structure underneath RSA, elliptic curves, and Diffie-Hellman that happens to require an exotic machine to execute. Given a large, fault-tolerant quantum computer (two qualifiers examined in detail in Chapter 10), those primitives end the way FEAL ended when differential cryptanalysis arrived.

WHERE THIS BREAKS

A break with a countdown.

The cryptanalytic-break frame has one important limit. A classical break is a paper, and the primitive dies the day the paper is published. Shor published his algorithm in 1994; what has been missing for three decades is the hardware to run it at scale. So this is a break with a prerequisite, and therefore a countdown, which is exactly what turns it into a risk-management problem instead of an incident-response one. The mathematics is already lost. Only the machine is pending.

MYTH AUTOPSY

"Quantum computers are millions of times faster."

The phrase fails twice. First, "faster" without a named problem is meaningless. Faster at what? Today's quantum processors execute gate operations at rates measured in the millions per second at best (thousands, on some platforms) against your CPU's billions, and error correction will slow the useful rate further. On almost every task you care about, the quantum machine loses, badly, and permanently. Second, where quantum computers do win, speed is the wrong category altogether. They win the way a break beats brute force, by making the hard step unnecessary rather than by doing it faster. Treat any headline of the form "X times faster" the way you treat a vendor claiming military-grade encryption: ask at what, exactly.

> **Say instead:** quantum computers make certain structured problems tractable that are classically intractable; on everything else they are no better, and usually worse.

What a quantum computer actually is (thirty seconds, no weirdness)

A quantum computer is a device that computes with amplitudes: quantities that behave like the probabilities you already use to reason about unknown bits, except that they can be negative, which means they can cancel each other out. That cancellation is called interference, and it is the entire trick. A quantum algorithm is a computation arranged so that the amplitudes of paths leading to wrong answers cancel while the amplitudes of paths leading to the right answer reinforce, so that when you finally read the machine, the right answer is what you are likely to get. The machine's cells are called qubits; a precise definition is Chapter 3's job.

That is the whole preview. It contains nothing about being in two places at once, nothing about parallel universes, nothing about consciousness. One new rule, probabilities that can cancel, and the rest of this series is consequences. Chapters 2 through 4 build it properly, starting from an object you already trust completely: an ordinary bit you have not looked at yet.

Why quantum computing is your problem, on a schedule

The reason this material is a professional obligation and no mere curiosity: the narrow class of problems with exploitable structure contains, almost as if by design, the public-key cryptography holding up everything else. Conventional TLS key exchange, most PKI and code signing, many VPN profiles, and much of DNSSEC all rest on the exact mathematics that Shor's algorithm breaks on a fault-tolerant machine. Hybrid post-quantum deployments have

begun changing that picture, which is the migration working, and they are nowhere near universal.

And the threat does not wait for the machine. An adversary recording ciphertext today only needs the machine to exist before the confidentiality lifetime of that data expires, the pattern known as harvest now, decrypt later (HNDL). Your deadline is therefore Q-Day minus your data's shelf life minus your migration time, and two of those three numbers are yours to measure. I take no position on dates here. Chapter 11 will equip you to read the hardware roadmaps yourself, and the rest is risk management, because regulators, insurers, investors, and clients have already set the deadlines whether or not anyone's prediction turns out to be right. The clock is already running.

FOR THE RECORD

Mosca's inequality.

The standard formalization comes from Michele Mosca, and I've written a CISO's guide to Mosca's theorem if you want the full treatment. In symbols: if $x + y > z$, where x is the number of years your data must stay confidential, y is the number of years your migration will take, and z is the number of years until a cryptographically relevant quantum computer exists, you are already late. The industry shorthand for that machine is CRQC: a quantum computer big and reliable enough to run Shor's algorithm against real-world key sizes. All three quantities are estimates. Only the first two are under your control.

The casualty list

Everything in this table is asserted now and earned in Chapters 8 and 9.

Primitive	Verdict	Settled in
RSA	Broken by Shor's algorithm on a fault-tolerant machine	Chapter 8
Diffie-Hellman (finite-field)	Broken by the same algorithm on the same machine	Chapter 8
Elliptic-curve everything (ECDH, ECDSA, EdDSA)	Broken, at even smaller machine sizes than RSA requires	Chapter 8
AES	Survives; prefer 256-bit keys, since Grover's algorithm manages only a quadratic dent	Chapter 9
SHA-2, SHA-3, HMAC	Survive with margin	Chapter 9

To put it bluntly: the bottom two rows are the least-reported fact in mainstream quantum coverage. Symmetric cryptography and hashing are fine. The quantum threat applies with precision to the structured mathematics that makes public-key cryptography possible. Structure is what this machine eats.

Four different things called "quantum"

> **MYTH AUTOPSY**
>
> **"Quantum encryption."**
>
> There is no product called quantum encryption. There are four unrelated things that marketing crams into that phrase, and your first question to any vendor saying "quantum-safe" is: which one do you mean?

Quantum computing (**QC**) is the threat side, the machine this series is about. It breaks structure-based public-key cryptography, and nobody sells it as a security product.

Post-quantum cryptography (**PQC**) is the fix: new classical algorithms built on mathematical problems believed hard for classical and quantum attackers alike. Pure software, running on the hardware you already own. NIST finalized the first standards in August 2024: FIPS 203, ML-KEM (formerly CRYSTALS-Kyber), for key establishment, plus FIPS 204, ML-DSA (formerly CRYSTALS-Dilithium), and FIPS 205, SLH-DSA (formerly SPHINCS+), for signatures; my 2025 standardization update covers the full picture. When anyone says quantum migration, this is what migrating means.

Quantum key distribution (**QKD**) is a physics-based method for two fixed endpoints to agree on key material over a dedicated optical link; I've written a QKD 101 for cybersecurity professionals if you're curious. It distributes keys; it does not encrypt data, and on its own it cannot authenticate the endpoints, so it needs classical authentication to resist man-in-the-middle. It is also point-to-point only. The NSA's published position is that PQC, and never QKD, is the migration path for national security systems, and the UK's NCSC takes the same view for general use.

Quantum random number generation (**QRNG**) is a hardware entropy source that harvests randomness from quantum processes; here's my introduction to QRNG. Legitimate, useful, and mundane: it is an entropy source, and it offers no protection against a quantum computer. Nothing about the threat model above changes.

One machine that attacks, one software fix, one niche link technology, one entropy widget. Four products share a single adjective; hence the confusion.

How to read this series

Eleven parts. Chapters 2 through 7 build the machine's logic from the ground up: probabilistic bits, amplitudes, interference, entanglement, and the shape of a quantum algorithm. Chapters 8 and 9 cash it out against cryptography. Chapters 10 and 11 cover the hardware reality and how to read the field like a professional. Every part stands alone, but the arguments compound in order.

Along the way you'll meet five kinds of sidebar, each with one job. THE ARITHMETIC holds optional numbers, self-contained and skippable; the prose is always complete on its own, but when you want to verify a claim by hand, that box is where you do it. WHERE THIS BREAKS states every analogy's honest limits at the point of use. FOR THE RECORD holds precision notes and references, for when you need to defend a claim upstream. MYTH AUTOPSY dissects a popular misconception at the exact moment you've gained the tools to see through it; two are already behind you. And every part closes with an ATTACKER'S-EYE VIEW, because that is the seat this series assumes you think from.

ATTACKER'S-EYE VIEW

What the rational adversary is doing right now.

Nothing about a well-resourced adversary's 2026 posture requires a quantum computer to exist. It requires packet capture. Recorded TLS sessions whose keys were agreed with ECDH are an appreciating asset: unreadable today, readable in full the day a CRQC runs. The targeting logic writes itself. Prioritize ciphertext whose value outlives the machine's plausible arrival: diplomatic and intelligence traffic, health and genomic records, legal files, weapons and industrial design data, source code. Note the mechanism carefully, because this is where most threat models go wrong. Forward secrecy does not rescue a recorded session; the attacker breaks the ephemeral key-exchange values sitting in the captured transcript. Long-lived private keys are the richer prize wherever static key transport or key wrapping still depends on them.

> A mirror threat aims at authenticity instead of confidentiality. Any signature whose trust must outlive the machine's arrival (firmware and code signing, root and long-lived certificates, notarized records) becomes forgeable once the underlying mathematics gives way. Be precise about what that means: the attacker cannot rewrite the past, they mint new signatures that verify under the old public key. Trusted timestamps, transparency logs, and append-only records can still prove what existed before the break, which is exactly why systems resting on bare signature verification are the exposed ones. I've written about this pattern as trust now, forge later: HNDL steals yesterday's secrets, while its twin steals tomorrow's trust in yesterday's signatures.
>
> The defensive consequence: your clock is set by two measurable quantities, your data's shelf life plus your migration time; nobody's qubit roadmap enters into it.

What to remember

A quantum computer is a different kind of machine, not a faster one; where it matters, it wins the way a cryptanalytic break beats brute force. The break is confined to structured public-key mathematics (RSA, elliptic curves, Diffie-Hellman), while symmetric ciphers and hashes remain secure with larger parameters. The fix, post-quantum cryptography, is classical software on hardware you already run; QKD and QRNG are different products answering different questions. Harvest-now-decrypt-later and its signature-forging twin tie your deadline to your data's lifetime rather than to anyone's Q-Day prediction. And nothing ahead requires accepting weirdness: from Chapter 3 onward, one new rule, probabilities that can cancel, explains everything.

Next, in Chapter 2, "The Classical Bit You Don't Know Yet": before a single qubit appears, we spend time with an object you trust completely, an ordinary bit whose value you have not looked at. We'll write down everything you believe about it. Every one of

those beliefs is reasonable. In Chapter 4, we'll use them to spring a trap.

The Classical Bit You Don't Know Yet

Right now, in a hardware security module you will never see, bit number 17 of somebody's 256-bit master key sits in silicon. No human knows its value. The vendor doesn't. The administrators who ran the key ceremony don't. The auditor who signed off on the ceremony certainly doesn't. And yet you would stake your career on one claim about that bit: it has a value. A definite one. It is a 0 or it is a 1, and your not knowing which changes nothing inside the chip.

This article is about that conviction. The plan is to take the beliefs you already hold about classical bits you haven't looked at, write them down precisely, and prove the most consequential one. Nothing here is exotic, and everything here is true. If you've read Chapter 1, you know where this is heading; if you haven't, this part stands on its own. Either way, you will leave holding four beliefs in writing, and I want you to keep the receipt. Chapter 4 intends to collect.

The classical bit and the numbers in your head

Start with what "256 bits of entropy" actually asserts. It is a claim about attackers, and no claim at all about the key. The key is a fixed string; every bit of it has been definite since the moment the RNG output settled. Entropy measures how little you know, which is why the same key can carry 256 bits of entropy for the outside world and zero for the HSM that holds it.

That observation generalizes into the first belief. To describe a bit you haven't seen, you write down a pair of numbers: the probability it's 0 and the probability it's 1. Sure of a zero: $(1, 0)$. Completely ignorant: $(1/2, 1/2)$, the pair I'll call a fair coin from here on. The

numbers are non-negative and they sum to one, because something has to be true. And the pair lives in your head rather than in the hardware. The admin who watched the RNG log holds $(1, 0)$ about the very same bit you hold $(1/2, 1/2)$ about, and neither of you is wrong. The bit has a value; observers have probabilities.

Belief 1: a classical bit always has a definite value. Probability pairs describe observers, and different observers can hold different pairs about the same bit.

What operations do to your numbers

You rarely leave bits alone, so the second belief covers operations, and it splits cleanly in two.

Deterministic operations move values around without touching your ignorance. NOT a bit you hold at $(1/2, 1/2)$ and you now hold $(1/2, 1/2)$. The labels swapped; your knowledge didn't. XOR with a constant you know: same story, which is the two-line version of why obfuscation is not encryption, a sentence your team has said in code review at least once.

Randomized operations are the interesting ones. Take the simplest: flip the bit with probability q, leave it alone otherwise. Your new pair is a weighted average of your old one, and the arithmetic box below holds the two lines that say so. What matters in prose is the shape of the rule: without looking at the bit, every operation you can perform turns your pair into an average over where the bit could have gone. Averages of non-negative numbers pile up. Nothing in the rule ever subtracts.

Belief 2: operating on an unseen bit updates your pair by averaging. Contributions add; nothing cancels.

THE ARITHMETIC

Run the update rule yourself.

A pair is (p_0, p_1) with $p_0 + p_1 = 1$. The operation "flip with

probability q" updates it to:

$$p_0' = (1-q)\,p_0 + q\,p_1 \qquad p_1' = q\,p_0 + (1-q)\,p_1$$

Set $q = 1/2$, the fairest possible randomizer, and start from certainty, $(1, 0)$. You get $p_0' = (1/2)(1) + (1/2)(0) = 1/2$: one application takes certainty to the fair coin $(1/2, 1/2)$. Apply it again: $p_0'' = (1/2)(1/2) + (1/2)(1/2) = 1/2$. Still the fair coin. A third pass changes nothing, and neither does a thousandth.

The general version takes two lines. Any operation R acts on your pair by weighting what it does to each definite input: the output is $p_0 \cdot R(0) + p_1 \cdot R(1)$. If R turns each definite input into the fair coin, so that $R(0) = R(1) = (1/2, 1/2)$, the output is $(p_0 + p_1)(1/2, 1/2) = (1/2, 1/2)$ for every possible starting pair. R's output carries no trace of its input. Feeding R's output back into R just hands it another pair, and every pair comes out as the fair coin.

One family of operations does manufacture certainty, and it needs ruling out as a loophole. Overwrite the bit with 0 and your pair becomes $(1, 0)$ no matter what it was. Certainty achieved, the boring way: by destroying the input. (Incident responders who arrive after the wipe know this form of certainty well.) An overwrite never turns a definite input into a fair coin, so it never has anything to un-mix. The loophole worth worrying about is different, and it's next.

Mixing is a one-way street

Suppose a process turns each definite input into a fair coin. By that very fact, its output does not depend on its input; the arithmetic box shows the dependence washing out in one line. So running the process twice buys you nothing. The second run receives a fair coin, ignores it the way it ignores everything, and emits a fair coin. Cer-

tainty in, coin out, coin forever after. Could some cleverer randomizing gadget escape this? No. The argument used nothing about the gadget except that both definite inputs come out uniform, and any escape would need a negative number somewhere to cancel a contribution, which probabilities do not offer.

Your profession already pays for this fact in several currencies. Hashing a weak password adds no entropy to it; the ignorance you can extract is capped by the ignorance you put in. Whitening the output of a broken RNG dresses the problem without fixing it, which is why NIST's SP 800-90B insists on assessing the raw noise source: conditioning functions can spread entropy around, and they cannot create it. Mixing moves ignorance. Only a genuine source makes more of it.

Belief 3: a recordless process that turns each definite input into a fair coin cannot be run again to recover certainty. Averaged-away information stays away.

That word "recordless" is doing real work, and the sharpest readers already have their objection loaded.

FOR THE RECORD

The one-time-pad objection.

XOR a message bit with a uniformly random pad bit and, to any attacker, the result is a perfect fair coin; Shannon's perfect secrecy, which vendors love to abuse, rests on exactly that. Yet XOR the same pad bit in again and the message returns. Randomization, undone. Doesn't that contradict Belief 3?

It does not, and the reason is the belief's most important word. The pad is a record. The XOR never destroyed the message's information; it moved it into the correlation between ciphertext and pad, and whoever holds the record can steer a second application to run the mixing in reverse. Belief 3 constrains processes that keep no record anywhere: fresh randomness, used once, then gone. Un-mixing is possible exactly when, and only where, the randomness was remembered.

Hold on to that word, recordless. Chapter 4 exhibits a device that un-mixes with no record in existence, which classical probability flatly forbids, and the same bookkeeping question, who kept a record of what, resurfaces in Chapter 10 as the reason quantum computers demand such extravagant engineering.

WHERE THIS BREAKS

This model's scheduled fracture.

For classical bits, it doesn't break. The picture above is exactly right, the information age is built on it, and you could retire on Beliefs 1 through 4. The reason this series spends a whole part on it is different: every rule here leaned, quietly, on the numbers in your pair being non-negative. Averages of non-negative numbers only fill in; they never cancel out. The next part introduces numbers that describe uncertainty yet can go negative, and Chapter 4 shows you a real device that those numbers describe and this model cannot. When the crack appears, it will run through the non-negativity you just relied on.

Looking without touching

The last belief covers the act your whole discipline is named for: observation.

Read the bit and your pair jumps to $(1, 0)$ or $(0, 1)$. The bit itself never moves. Nothing about reading disturbs classical information, and nothing about copying does either; a forensic image taken behind a write blocker is bit-perfect, repeatable, and invisible to the disk. The same holds on the wire. A passive tap produces a flawless copy while the endpoints exchange packets in blissful ignorance, which is why lawful intercept works, why harvest now, decrypt later is the patient attacker's rational strategy, and why

your IDS hunts for an interceptor's side effects: the interception itself emits nothing to find.

Belief 4: reading a classical bit updates your numbers and never the bit. Copying is free, perfect, and silent.

There exists information for which every clause of Belief 4 fails, information that reading changes and that cannot be copied at all. The next part introduces it, and one commercial technology you met in Chapter 1's disambiguation sidebar, QKD, sells exactly the difference: on a quantum link, the tap is the disturbance. For today, though, the classical picture stands in full. The bit never notices.

ATTACKER'S-EYE VIEW

Why passive attacks are silent.

Every silent attack in your threat model is silent because of Beliefs 1 through 4. Packet capture and disk imaging work at scale and without a trace because classical information copies perfectly and reads without disturbing (Belief 4). Recorded ciphertext keeps its value indefinitely because the definite plaintext behind it is going nowhere (Belief 1). And the defender's side of the ledger is written by Belief 3: no computation, however large, un-mixes a properly generated key, so brute force is never a reversal of your mixing, only an enumeration of the attacker's ignorance.

Which is why real attackers hunt the record instead. Seeds, RNG states, pads, key ceremonies: the places where randomness was remembered. When Debian's OpenSSL package shipped with a crippled entropy source in 2008, SSH and TLS keys across the internet became enumerable overnight. Nobody un-mixed anything; the mixing had drawn from a pool of a few tens of thousands of possibilities, and attackers simply listed them. The lesson generalizes. Against classical cryptography done right, the paths to certainty run through records and side channels, never through reversing randomness. Hold that map in mind. The information in the next part redraws it.

What to remember

A classical bit always has a definite value; probability pairs like (1/2, 1/2) measure an observer's ignorance, and different observers can hold different pairs about the same bit. Operations on unseen bits update those pairs by averaging, and averages of non-negative numbers add without ever cancelling. A recordless process that turns definite inputs into fair coins cannot be run again to recover certainty; the one-time pad escapes only because the pad is the record. Reading a classical bit changes your numbers and never the bit, which is why copying is silent and passive attacks are undetectable in principle. All four beliefs are true, provable, and now on the record, and Chapter 4 will break exactly one of them in front of you.

Next, in Chapter 3, "Quantum Amplitudes: The One New Rule": we change a single assumption in everything above. The numbers describing your uncertainty will be allowed to go negative. That one permission, handled honestly, is the entire difference between the machine on your desk and the machine this series is about.

Quantum Amplitudes: The One New Rule

In the summer of 1926, with his paper already at the printer, Max Born added a footnote. The paper proposed that the strange new quantum theory predicts probabilities, and that the probability of finding a particle somewhere is given by the size of a quantity called the amplitude. The footnote corrected one word: by the *square* of the amplitude. Physics has run on that correction ever since. It earned Born the Nobel Prize in 1954, twenty-eight years later, and it will run every quantum computer that ever threatens your cryptography.

It is also the only new rule this article needs. Chapter 2 described a bit you haven't seen as a pair of non-negative probabilities summing to one, and put four beliefs about that picture on the record. This article introduces quantum amplitudes by changing that pair in exactly one way, and the change fits in a sentence: the numbers may now be negative. By the end you will know what a qubit's state actually is, what measurement does and what it costs, why "0 and 1 at the same time" was never right, and where the new physics hides: in a sign that no single measurement can see. One honest warning: nothing in this part disproves your classical instincts. That demolition is scheduled, and it is the next part's entire job.

From probabilities to quantum amplitudes

Take the pair from Chapter 2 and relax one constraint. A qubit's state is a pair of amplitudes, (a_0, a_1): real numbers, positive or negative, whose squares sum to one. To predict a measurement, apply Born's footnote. The outcome is 0 with probability a_0^2 and 1 with probability a_1^2. That is the entire rule, in modern dress.

Notice how much of Chapter 2 carries over untouched. Two numbers describe the qubit. The squares behave exactly like your old probabilities, non-negative and summing to one, because something has to happen. And the state itself is perfectly definite. There is nothing vague about $(0.6, -0.8)$; it is as definite as a row in a config file. What the state leaves undetermined is the future: which outcome the next measurement will produce.

A qubit, then, is any physical system with two reliably distinguishable configurations whose behavior demands this bookkeeping: the direction of a current in a superconducting loop, the energy level of a trapped ion, the polarization of a photon. My earlier introduction to qubits for cybersecurity professionals surveys the hardware menagerie, and the engineering is the subject of Chapter 10. For this series, the qubit is the amplitude pair.

One difference from Chapter 2 deserves its own paragraph. Probability pairs lived in observers' heads; the admin and the outsider held different pairs about the same bit, and both were right. The amplitude pair is fixed by the preparation instead. Everyone who knows how the qubit was set writes down the same pair, signs included, which makes it feel less like opinion and more like a dial setting. Whether something definite hides underneath the dial is a fair question, and Chapter 2's Belief 1 says your instincts vote yes. Hold that thought. It has an appointment.

THE ARITHMETIC

Born's rule, by hand.

The state $(3/5, 4/5)$ is legal, since $(3/5)^2 + (4/5)^2 = 9/25 + 16/25 = 1$. Measure it: you get 0 with probability $9/25 = 36\%$ and 1 with probability $16/25 = 64\%$.

The state $(1/\sqrt{2}, 1/\sqrt{2})$, where $1/\sqrt{2} \approx 0.707$, gives $1/2$ and $1/2$: a perfect fair coin.

Now the ones that matter. $(1/\sqrt{2}, -1/\sqrt{2})$ squares to the same $1/2$ and $1/2$, because squaring erases the sign. So does $(-3/5, 4/5)$ reproduce the statistics of $(3/5, 4/5)$. Different

states, identical measurement outcomes. Whatever the minus sign is doing, a single measurement cannot see it.

FOR THE RECORD

Real amplitudes are a deliberate simplification.

In full quantum mechanics, amplitudes are complex numbers and the Born rule squares their magnitude. This series uses real amplitudes throughout, because anyone can check a sign by hand and nothing we cover requires more. A complex phase is the grown-up version of a minus sign; every argument you will read here works unchanged in the general theory. Where the difference could ever matter, I will say so.

Measurement, and the price of looking

Here is what measurement does, stated as rules you can hold me to. Measuring a qubit in state (a_0, a_1) returns a single classical bit, 0 or 1, with the Born probabilities. Immediately afterward, the qubit's state is the definite pair matching the outcome: $(1, 0)$ if you got 0, $(0, 1)$ if you got 1. Measure again and you get the same answer, consistently, forever. The original amplitudes are spent. No further measurement of that qubit recovers them.

Compare that with Chapter 2's Belief 4, which said reading updates your numbers and never the bit. The first half still holds; the second half fails outright. Measuring an amplitude pair changes the pair, and the change is violent: whatever information lived in the balance and signs of (a_0, a_1) is reduced to one classical bit, with the rest destroyed in the act of asking. How brutally this caps what a register of qubits can tell you is Chapter 5's subject. For now, the accounting for a single qubit is stark enough. One bit out. Everything else gone. Looking, here, is spending.

MYTH AUTOPSY

"Measurement requires a conscious observer."

None of the definitions above mentioned a mind, and none needs to. A measurement is any interaction that leaves a record of the outcome in something else: a detector's counter, a log line, a stray photon that bounced off the system and carried the news away, a single air molecule that scattered differently because the outcome was 0 rather than 1. The Geiger counter clicking in an empty lab is measuring. The SIEM writes the log line whether or not an analyst ever reads it. Chapter 2 taught you the operative word, and it was record, never mind.

The consciousness framing entered through decades of loose popularization, and it adds exactly nothing: no prediction changes, no equation acquires a new term. It is also why this series says measurement and never "observation," a word that smuggles a watcher into physics that has no seat for one. In Chapter 10 the record-making runs wild under its proper name, decoherence: the environment measures your qubits constantly, uninvited, which is most of why building this machine is so hard.

Say instead: a measurement is any interaction that leaves a record anywhere. Minds are optional and always were.

What superposition actually says

With the rules on the table, the vocabulary word can finally be defined without mysticism. A superposition is any state with both amplitudes nonzero. $(3/5, 4/5)$ is a superposition. So is the fair-coin state $(1/\sqrt{2}, 1/\sqrt{2})$. That is the entire definition. It describes one definite mathematical object, two numbers, no fog.

MYTH AUTOPSY

"A qubit is 0 and 1 at the same time."

The most popular sentence in quantum journalism fails three separate ways. First, it predicts something that has never been seen. If the qubit were both values at once, some measurement somewhere should have caught both; in a century of experiments, every measurement of every qubit has returned exactly one bit. Second, the slogan is blind to the sign. $(1/\sqrt{2}, 1/\sqrt{2})$ and $(1/\sqrt{2}, -1/\sqrt{2})$ would both be "0 and 1 at once," yet they are different states that a well-chosen operation treats oppositely; a description with no room for the minus sign has missed the engine of the entire subject. Third, it confuses the description with the value. The state is one definite thing. The value, before measurement, is what does not yet exist; there are two amplitudes, and there are zero values.

Say instead: a qubit holds a weighted combination of 0 and 1, and the weights can cancel.

The other popular framing has earned a stay of execution. "It is really 0 or 1, we just don't know which" is Chapter 2's Belief 1 wearing quantum clothes, and nothing above refutes it. A single qubit, prepared and measured, is indistinguishable from a biased coin; you could model everything above with hidden definite values and secret probabilities, and today's evidence could not convict you. Your skepticism is earned. Keep it loaded. The next part exists because there is an experiment your model must survive, and it will not.

Two states, one coin

Look again at the pair of states from the arithmetic box: $(1/\sqrt{2}, 1/\sqrt{2})$ and $(1/\sqrt{2}, -1/\sqrt{2})$. The Born rule squares both into the same fair coin. Prepare either one, measure a million

copies, and the two tallies are statistically identical. The sign exists in the description and vanishes in the statistics.

So is the minus sign physics, or is it notation? The theory's answer is blunt: the two states are as distinguishable, in principle, as 0 and 1, because there exist operations that treat them in opposite ways. To see the difference you must let amplitudes meet other amplitudes, signs intact, before any measurement squares them away. Signed numbers only reveal themselves when they are added, because adding is the one thing squares can't undo: $+1/\sqrt{2}$ and $-1/\sqrt{2}$ make zero, and zero is a probability you can notice. The sign needs a collision.

Building the collider takes one operation and four lines of arithmetic, and it produces the device Chapter 2's beliefs cannot describe: a coin that randomizes once and un-randomizes twice, with no record anywhere. Signs are invisible until they collide.

ATTACKER'S-EYE VIEW

The end of the silent tap.

Chapter 2 closed on why passive attacks are silent: classical information reads without disturbance and copies without a trace. The Born rule ends that. Measurement changes the state, so interception acquires a fingerprint; run an intercept-and-resend attack against a BB84-style quantum key exchange and you corrupt roughly 25% of the bits you touch, which the endpoints detect by comparing samples. The entire commercial premise of QKD, from Chapter 1's disambiguation sidebar, is selling that fingerprint.

The one-shot rule cuts against the attacker too. A stolen qubit is nothing like a stolen file: one measurement, one bit, and the rest of the state burns. No imaging, no replay, no second pass. And the sign channel should interest anyone who thinks about covert storage: information parked in the signs of amplitudes is invisible to an adversary who only measures, and extracting it requires the interference machinery of the next part, aimed correctly.

> The defender's version of the same fact is sobering. Everything that leaves a record measures, and the universe leaves records promiscuously; a stray photon is an eavesdropper with no agenda. Keeping amplitudes alive means keeping everything's hands off the qubit, and the bill for that isolation, in Chapter 10, explains most of the distance between today's machines and the one that breaks RSA.

What to remember

A qubit's state is a pair of amplitudes: numbers that behave like probabilities except they can be negative, with squares summing to one. The Born rule turns state into prediction: measurement returns 0 or 1 with probability equal to the squared amplitude, after which the pair is replaced by the definite pair for the outcome, and the original amplitudes are spent. Measurement means any interaction that leaves a record anywhere; minds are optional and always were. Superposition is one definite state whose weights can cancel, never "0 and 1 at the same time," while the "really definite, just hidden" reading remains standing, for now, on borrowed time. And two states can share identical statistics while differing by a single sign, which only a collision between amplitudes can reveal.

Next, in Chapter 4, "The Impossible Coin": we build the collider. One operation, applied twice, will return certainty from a perfect fair coin with no record in existence, and Chapter 2's receipt comes due. Bring the four beliefs. Exactly one leaves the room unchanged.

The Impossible Coin

A courier delivers a sealed device to your lab. One input port, one output port, no manual. The job is the one your profession trains for: characterize the black box.

You feed it a definite 0 and measure the output: 0 about half the time, 1 the other half, and over ten thousand trials the tally is a textbook fair coin. You feed it a definite 1. Fair coin again. By every test Chapter 2 taught you, the verdict writes itself: the device is a randomizer, apparently the recordless kind, and Belief 3 has already told you everything it will ever do.

So you chain two of them in series and feed the pair a definite 0. Out comes 0. Again: 0. Ten thousand runs, ten thousand zeros. Feed the chain a 1 and it returns 1 with the same monotony. One copy of this device manufactures perfect randomness. Two copies in a row manufacture perfect certainty, and nowhere in the glass and silicon is any record of what passed through.

Chapter 2 proved that impossible. Not unlikely. Impossible, by arithmetic you checked yourself. This article is about the cleanest quantum interference experiment ever devised, and it does four things: replays the impossibility proof, shows the amplitude arithmetic that predicts the bench perfectly, closes the one loophole your training will reach for, and then collects on the receipt. To put it plainly: if you verify one thing in this whole series by hand, make it the box below.

Three doors out

The classical argument, compressed from Chapter 2's arithmetic box: an operation using fresh, unremembered randomness acts on your probability pair by averaging. If it turns each definite input into a fair coin, its output carries no dependence on its input, so a second application receives a coin, ignores it, and emits a coin.

Certainty in, coin out, coin forever. The bench says certainty comes back. Something in the setup has to give, and there are exactly three candidates.

Door one: fraud. The demonstration is rigged, the device is a stage prop. Except this is among the most repeated experiments in physics; it runs on optics benches in undergraduate labs, and in software it is the first instruction of nearly every quantum program ever written. Names, dates, and a Nobel Prize are waiting in a box below. Door one is closed.

Behind door two, the respectable candidate: a hidden record. Chapter 2 allowed exactly one escape from irreversibility, and the one-time pad taught it: un-mixing is possible where the randomness was remembered. Maybe the device keeps a pad. This door deserves a real test rather than a wave of the hand, and it gets one, two sections down.

And door three: the model. Perhaps a probability pair was never the right description of what travels between the two stages. That door leads somewhere.

Quantum interference, by the numbers

Chapter 3 ended on a cliffhanger built for this moment: two states, $(1/\sqrt{2}, 1/\sqrt{2})$ and $(1/\sqrt{2}, -1/\sqrt{2})$, identical under measurement, different by one invisible sign. Here is the resolution. Those two states are this device's two outputs. In amplitude language, the device maps a definite 0, the state $(1, 0)$, to $(1/\sqrt{2}, 1/\sqrt{2})$, and a definite 1, the state $(0, 1)$, to $(1/\sqrt{2}, -1/\sqrt{2})$. Measure either output once and the Born rule squares away the sign: fair coin, both times, exactly as your first session found. The device manufactures the invisible sign.

The second stage is where the sign stops being invisible. The device acts on amplitudes by the same linearity Chapter 2 established for probabilities: feed it a general state and it processes each component, then adds. Except these components carry signs, and signed contributions meeting at the same output can do the

one thing probabilities never could. Cancel. Run the second stage on $(1/\sqrt{2}, 1/\sqrt{2})$ and the two routes into the "1" output arrive carrying $+1/2$ and $-1/2$: they annihilate, and the output-1 amplitude is exactly zero. The two routes into "0" arrive carrying $+1/2$ and $+1/2$: they reinforce, to exactly one. The state after the second stage is $(1, 0)$. Certainty, reconstituted.

This collision has a name, quantum interference, and you have just watched the only mechanism by which a quantum computer ever beats a classical one. Wrong answers cancel; right answers reinforce. That was Chapter 1's thirty-second preview of the entire field, and it just happened in front of you in four lines.

THE ARITHMETIC

The four lines that matter most.

The device's rule on definite inputs: $(1, 0) \to (1/\sqrt{2},\ 1/\sqrt{2})$ and $(0, 1) \to (1/\sqrt{2},\ -1/\sqrt{2})$. On a general state it acts by linearity, exactly as in Chapter 2's box: process each component, add the results. So the new amplitudes are:

$$a'_0 = \frac{a_0 + a_1}{\sqrt{2}} \qquad a'_1 = \frac{a_0 - a_1}{\sqrt{2}}$$

Start from a definite 0, state $(1, 0)$. First pass: $a'_0 = 1/\sqrt{2}$, $a'_1 = 1/\sqrt{2}$. Second pass: $a''_0 = (1/\sqrt{2}+1/\sqrt{2})/\sqrt{2} = 1$ and $a''_1 = (1/\sqrt{2}-1/\sqrt{2})/\sqrt{2} = 0$. Final state $(1, 0)$: the output is 0 with certainty. Start from $(0, 1)$ instead and the signs flow through to give $(0, 1)$ back. Check it; it is six additions.

Now the peek. Measure between the stages and the Born rule fires: you get 0 or 1, evens, and the state becomes the matching definite pair. The second stage then does what it does to every definite input: produces a fair coin. Peeked chain: 50/50. Unpeeked chain: certainty. Your own arithmetic predicts that looking in the middle changes the answer at the end.

The record loophole, tested

Door two remains, and it is the respectable one. Chapter 2's one-time-pad box established that apparent un-mixing is possible when the randomness is remembered somewhere, so the professional hypothesis is that the device pads: it secretly records what it did on pass one and undoes it on pass two. The hypothesis is testable, because a record is a physical thing, and you can add one of your own.

Install a detector between the two stages, any mechanism at all that registers which value went past. The bench result is unambiguous, and it has been since the interference experiments of the 1980s: the certainty vanishes. The chain's output becomes a fair coin, precisely the peeked-chain arithmetic from the box. And the detail that should stop you mid-coffee is that nobody needs to read the detector. Its record can sit unexamined forever; the mere existence of the information, anywhere, in anything, produces the 50/50 outcome. That is Chapter 3's definition of measurement doing live ammunition work: an interaction that leaves a record is a measurement, minds optional.

So run the logic. If the device relied on an internal record to un-mix, an extra record should not hurt; a pad plus a bystander's photocopy of the pad still decrypts. Instead, any record kills the effect. The un-mixing exists only in the total absence of records, which is the exact configuration Chapter 2 proved cannot un-mix. Classically, the record was your only road back. Here, a record is what closes the road. The reversal runs on silence.

MYTH AUTOPSY

"It's really 0 or 1, we just don't know which."

The most respectable myth in this series was granted a stay of execution in Chapter 3, because single measurements could not refute it. The stay expires here. Give the story every courtesy and let it speak precisely, about the moment between the two stages, in a chain now certified record-free.

Premise: between the stages, the qubit is secretly definite, 0 or 1, even odds, and the pair $(1/\sqrt{2}, 1/\sqrt{2})$ merely describes your ignorance. Established behavior: this device turns each definite input into a fair coin; that was your first characterization session, ten thousand trials a side. Prediction: the second stage receives a definite value and therefore outputs a fair coin. Observation: the second stage outputs the original input, every time.

The premise did the predicting, and the premise is dead. Between the stages there is no secret value; there is the state $(1/\sqrt{2}, 1/\sqrt{2})$ itself, one definite mathematical object whose halves met signed halves and cancelled. Belief 1's picture, a value plus ignorance, cannot produce a minus sign, and the minus sign is measurably there: it is the difference between the certainty you observe and the coin the story predicts.

Say instead: a superposition has no underlying value, and the proof is operational: treating it as a hidden value predicts the wrong statistics for an experiment you can run.

FOR THE RECORD

This is a real bench, and also line one of every program.

On an optics table the device is a half-silvered mirror, and two of them bracket a Mach-Zehnder interferometer; the "coin" is which path the photon takes. Grangier, Roger, and Aspect ran the single-photon version in 1986, one photon at a time through the apparatus, and the interference held with near-perfect visibility. Aspect later shared the 2022 Nobel Prize for the experimental program, with entangled photons, that closed out the hidden-variable question at large; that larger story, Bell's theorem included, arrives with entanglement in Chapter 6.

For completeness: hidden-variable theories that survive this experiment do exist. De Broglie and Bohm built one by letting

the amplitudes steer hidden values nonlocally, which means the amplitudes stay load-bearing either way, and nothing in this series changes under that reading. The casualty here is the specific, common-sense version your Chapter 2 receipt encoded.

In software, the device is the Hadamard gate, typically the first instruction of any quantum program you will ever read. When Chapter 8 breaks RSA, the opening move is this article, applied to every qubit in the register.

WHERE THIS BREAKS

The coin has no faces.

The impossible coin is this series' keystone analogy, so its limits get stated plainly. Between measurements there is nothing two-sided in flight; the "sides" are amplitudes, and heads-or-tails imagery smuggles Belief 1 back in through the fire exit. And the clean numbers above are the noiseless limit. A real bench leaks, every leak is a partial record, and partial records buy partial coins: real devices return the input most of the time rather than all of it. The distance between "most" and "all," multiplied by a few million qubits, is Chapter 10's entire subject.

Stamping the receipt

Chapter 2 closed with four beliefs on the record and a promise that exactly one would leave this room unchanged. The audit:

Belief 1, dead for qubits. There is no definite value under the pair; the hidden-value premise predicts a coin where the bench delivers certainty. (Your classical bits keep the belief. Scope matters.)

Belief 2, dead. Contributions can subtract. The update rule keeps its shape, linearity and all, except it now runs on amplitudes, and $+1/2$ met $-1/2$ in front of witnesses.

Belief 3, dead on its own terms. A recordless process un-mixed, which Chapter 2 proved no probability process can do, and the escape clause inverted: records no longer enable the reversal, they are the one thing that prevents it.

Belief 4, unchanged today. This experiment never touched it. Chapter 3 already bent its reading clause, and Chapter 5 takes the copying clause. It should not feel safe.

What stands in the wreckage is the description that predicted everything: the amplitude pair of Chapter 3, now promoted from bookkeeping to the thing the machine actually computes with. The state between the stages is $(1/\sqrt{2}, 1/\sqrt{2})$, definite, valueless, and cancellable, and every quantum algorithm you will ever hear about is a scheme for arranging such states so that the wrong answers cancel. The receipt is stamped.

ATTACKER'S-EYE VIEW

One trick, at scale.

Everything on Chapter 1's casualty list is this bench trick, industrialized. Shor's algorithm in Chapter 8 arranges a computation over an astronomical number of routes so that amplitudes flowing toward wrong answers meet with opposing signs and annihilate, exactly like the $+1/2$ and $-1/2$ you just watched, while routes agreeing on the hidden structure of the key reinforce. The general choreography, and why it works only for problems with structure, is Chapter 7's subject.

The record test cuts the other way too, and defenders of future systems should file it. An interference computation dies the moment any record of its intermediate values exists anywhere, so one planted probe, one leaky coupling, one overcurious diagnostic is a denial-of-service attack on a quantum computer; measuring the machine is breaking the machine. The environment mounts this attack for free, continuously, with no adversary required, and holding it off is the engineering mountain of Chapter 10.

> One more consequence worth sixty seconds of paranoia: the sign channel from Chapter 3 is now a working covert channel. Information parked in relative signs is invisible to any inspector who only measures, and extracting it takes an interferometer aimed by someone who knows the encoding. Storage the auditor cannot audit, readable only by the party holding the layout: file that under both threats and tools.

What to remember

A single device turns each definite bit into a fair coin, yet two in series return the original input with certainty and no record anywhere, which Chapter 2's arithmetic proves impossible for any classical random process. The amplitude description predicts it in four additions: the device manufactures Chapter 3's invisible sign, and the second pass makes signed halves collide, cancelling the wrong output and reinforcing the right one. Any record of the intermediate value, read or unread, destroys the effect, so the reversal exists precisely where classical probability forbids it. The hidden-value story, granted every courtesy, predicts a coin where the bench delivers certainty; between the stages there is no secret value, only a superposition. And this cancellation is the sole engine of quantum computational advantage, the same move that eventually breaks RSA.

Next, in Chapter 5, "Vast State, Narrow Door": we scale from one qubit to many, meet the 2^n amplitudes that make quantum states vast, and the n-bit measurement that keeps them almost unreadable. The "tries every answer at once" myth finally gets its autopsy, and Belief 4 loses its last clause: quantum information cannot be copied at all.

Vast State, Narrow Door

Sometime this quarter, a vendor will tell you that their 300-qubit processor "occupies more states than there are atoms in the observable universe." The sentence is true. 2^{300} is around 10^{90}, and the atom count comes up roughly ten orders of magnitude short. It is also the setup for the most durable con in quantum journalism, because the sentence that follows is always some variant of "so it can try every answer at once," and that sentence is false in a way you can now prove.

Chapter 4 left three of your four classical beliefs dead on the bench and one, Belief 4, untouched and nervous. This article scales the amplitude picture from one qubit to many, and the scaling settles everything left open. You will meet the vastness honestly, because it is real and it is the reason this field exists. You will meet the door it all has to fit through, which is the qubit information limit: what n qubits hold, and the far smaller amount they will ever tell you. The obvious hack for widening the door will fail in four lines of arithmetic, taking Belief 4 with it, and then the myth this series was founded to kill finally holds still long enough for an autopsy.

Two qubits, four numbers

The generalization from Chapter 3 is mechanical. One qubit carried two amplitudes, one per possible value. Two qubits carry four amplitudes, one per possible two-bit string: an ordered list $(a_{00}, a_{01}, a_{10}, a_{11})$, real numbers, positive or negative, squares summing to one. The Born rule reads the same way it always has. Measure both qubits and you receive one two-bit string; the probability of string xy is a_{xy}^2, and afterward the register sits in the definite state for the string you got.

Three qubits: eight amplitudes. Ten: 1,024. The rule is one amplitude per bit-string, so n qubits are described by 2^n signed numbers,

and the growth does what exponentials do. At fifty qubits the description runs to about 10^{15} amplitudes, roughly ten petabytes just to write down. At three hundred, the description has more entries than the universe has atoms. Every word of the vendor's sentence checks out, and none of the arithmetic so far says anything about what you can get back out. That is the door's job.

The narrow door on qubit information

Here is the rule, and it is the same Born rule wearing bigger shoes. Measuring an n-qubit register returns exactly one n-bit string, chosen at the odds the squared amplitudes dictate, and the register's state is spent in the asking: whatever lived in the balance and signs of 2^n numbers is gone, replaced by the definite state of the string you received. Run the numbers on the vendor's processor. The state going in: 2^{300} amplitudes. The payout coming out: 300 bits, about 38 bytes, and no second withdrawal. The door is n bits wide.

The general theorem behind this has a name and a surprising date. Alexander Holevo proved in 1973 that n qubits can never be made to yield more than n classical bits, no matter how cleverly you encode and no matter how exotically you measure. That is two decades before anyone said the word "qubit," a bound on quantum computing published before quantum computing was proposed. There is exactly one loophole, and it is not the one hype needs: if sender and receiver already share an entangled pair, one qubit can carry two classical bits, a trick called superdense coding. It buys a factor of two and consumes a pair it had to distribute in advance. Two, not 2^n. I consider Holevo's theorem the most security-relevant physics result that nobody in security has heard of, because it is the ceiling on every "quantum computers read databases instantly" claim you will ever be sold.

Both facts belong together, because the pairing is the whole picture. The state is vast to describe and narrow to query: astronomical internal workings behind a single destructive n-bit interface. Your profession ships a product with exactly that shape. An HSM holds key material you can never export, only invoke through a narrow

API, and n qubits are nature's HSM, with a stingier API than anything FIPS 140-3 ever certified: one call, n bits, and the state clears on read.

WHERE THIS BREAKS

Nature's HSM, and where the analogy stops.

Two honest differences. An HSM's keys are ordinary definite bits, and a sufficiently funded attacker with lab equipment and patience can, in principle, extract them, because they are physically there to find. Amplitudes are not there in that sense. Chapter 4 closed the hidden-value reading, and no attack, however invasive, reads out the amplitudes of a single unknown state; "invasive" is just another word for measuring, and measuring is the one call you get. Second, an HSM's internal state persists across a million queries. A qubit register grants exactly one, then holds only the aftermath. The analogy earns its keep on the shape, vast-inside and narrow-outside, and should be retired beyond that.

The copy loophole

Every security person reading the last section had the same thought within a paragraph: fine, one measurement per state, so copy the state first. Clone the register a thousand times, measure the copies against different questions, and reconstruct the amplitudes, signs included, by statistics. If copying works, Holevo's door swings wide open.

Copying does not work, and you have owned the proof since Chapter 2. Operations act on amplitudes by linearity, and linearity dictates what a copier does to a superposition with no room for negotiation. The arithmetic box below runs it: build the best copying machine physics allows, one that duplicates definite inputs perfectly, and feed it the fair-coin state. Out comes something that behaves nothing like two copies. Measure the output pair and the two qubits agree with certainty, 00 or 11 and never anything else,

while genuine independent copies would disagree half the time. The machine did produce a strange and useful object, and naming it is the next part's job. What it did not produce is a copy.

This is the no-cloning theorem, put on formal footing by Wootters and Zurek in 1982, with Dieks arriving at it independently the same year: an unknown quantum state cannot be copied, full stop, by any machine, ever. It is a two-line consequence of linearity, which means it is exactly as negotiable as arithmetic. And with it, the last clause of the last classical belief goes down. Chapter 2's receipt is now fully spent:

Belief 4, dead. Reading disturbs (Chapter 3), and copying is impossible outright (today). Nothing about classical information's silence carries over. The bit that never noticed has been replaced by a state that cannot help noticing.

THE ARITHMETIC

The copier that entangles instead.

Two qubits, four amplitudes, ordered $(a_{00}, a_{01}, a_{10}, a_{11})$. The copier's spec on definite inputs: write qubit one's value onto a blank qubit two. So "0, blank" maps to "00": $(1,0,0,0) \to (1,0,0,0)$, and "1, blank" maps to "11": $(0,0,1,0) \to (0,0,0,1)$. Both required, both reasonable, and on definite inputs this machine copies flawlessly.

Now feed it the fair-coin qubit plus a blank: the state $(1/\sqrt{2},\ 0,\ 1/\sqrt{2},\ 0)$. Linearity processes each component and adds:

$$\frac{1}{\sqrt{2}}(1,0,0,0) + \frac{1}{\sqrt{2}}(0,0,0,1) = \left(\frac{1}{\sqrt{2}},\ 0,\ 0,\ \frac{1}{\sqrt{2}}\right)$$

Compare that with what two genuine copies of the fair coin would be: each qubit independently 50/50, all four strings equally likely, the state $(1/2, 1/2, 1/2, 1/2)$. The machine's actual output puts probability $1/2$ on 00, $1/2$ on 11, and

exactly zero on 01 and 10. Measure real clones a thousand times and they disagree about half the time. Measure this machine's output a thousand times and the two qubits never disagree once. Not copies. Something else, with a perfect correlation the copies would lack, and a name waiting in Chapter 6.

MYTH AUTOPSY

"It tries every answer at once."

The myth this series exists to retire gets its full hearing, both halves. The honest half first: put n qubits each through Chapter 4's device and the register is left in an equal superposition over all 2^n bit-strings, and a function evaluated on that register acts across every amplitude in one pass. The exponential workspace is real. Nobody serious disputes it, and Chapter 7 will use it.

The fraudulent half is the next word: "...and then you read the answers." You do not. One measurement, n bits, one Born-random string, state destroyed, and Holevo guarantees no cleverness improves the exchange rate. No copies for a second look either; the box above just closed that door. Total up naive quantum parallelism and the yield is one uniformly random input-output pair, which a classical computer produces by evaluating the function once on one random input. You built a dilution refrigerator to reimplement random.choice.

The maze version of the myth fails the same audit. The picture promises a machine that explores every corridor and reports the good one; unaided by anything, it reports a random corridor. What turns the workspace into an advantage is the one mechanism this series has shown you, interference, aimed by problem structure so the wrong corridors cancel before anyone looks. That choreography is Chapter 7's subject, and the myth's replacement is already on your shelf.

> **Say instead:** it computes with the amplitudes of every possibility, then uses interference so that mostly the right answer's amplitude survives measurement.

What the vastness is actually for

The two-front honesty this series owes you: after an article spent capping the hype, the denial deserves its turn against the wall. The 2^n workspace is physically real and computationally decisive, and the cleanest proof is the mirror image of everything above. If describing fifty qubits takes ten petabytes, then simulating fifty qubits classically takes ten petabytes, and every added qubit doubles the bill. That wall is where classical simulation of quantum systems goes to die, and it is the founding observation of this entire field: Richard Feynman's 1982 argument was precisely that classical machines pay exponentially to imitate quantum physics, so the honest move is to compute with quantum physics directly.

So the vastness is the machine's workspace, where amplitudes flow, meet, and cancel by the millions in patterns no classical memory could afford to track. What it will never be is retrievable storage, because the door stays n bits wide and the contents cannot be copied out. Every real quantum algorithm lives in the gap between those two facts, compressing an exponential computation through a linear door by making interference do the compression. A workspace, never a warehouse.

> ATTACKER'S-EYE VIEW
>
> **There is no dd for qubits.**
>
> Run the exfiltration math, and be precise about what physics forbids. An attacker who seizes a quantum register can carry it off, store it, transform it, and choose when to measure it; possession is not prohibited. What they cannot do is keep the original while manufacturing perfect duplicates to test

against many incompatible questions. A stolen classical drive is a perfect, silent, infinitely replayable copy. A stolen quantum register grants one destructive read and no replay. That single difference deletes the mechanism behind an entire attack class: harvest now, decrypt later works because classical ciphertext can be copied and shelved for the decade it takes the decryption capability to arrive, and a quantum state cannot be copied to the shelf at all. Today's quantum memories are also short-lived and lossy, which is engineering rather than theorem, but it compounds the point.

The defender inherits the same physics as a burden. An unknown quantum state cannot be forensically imaged into a perfect independent copy while the original sits untouched, so every compliance instinct built on "capture first, examine later" meets a system where capture spends the evidence. Auditing quantum systems will lean on controlled measurements, protocol transcripts, and repeated preparations rather than a bit-for-bit image. File that now; it matures into real operational questions the day quantum links and quantum memories enter your estate.

And the door disciplines the attacker's crown jewel. Shor's algorithm must squeeze an exponential workspace through those same n bits, which is why it works only on problems whose answer is one compact global pattern, a period, that interference can concentrate into the single string the door lets out. The narrow door is why your symmetric keys get to live and your RSA keys do not.

What to remember

n qubits are described by 2^n signed amplitudes, one per bit-string, and the Born rule now pays out strings instead of bits. One measurement returns n classical bits and spends the state, and Holevo proved in 1973 that no encoding or measurement scheme ever beats that exchange rate. Copying is not an escape: linearity forces any copier fed a superposition to produce perfectly correlated

non-copies instead, which kills Belief 4's last clause and finishes Chapter 2's receipt. Naive parallelism therefore yields one random sample, no better than classical guessing, and quantum advantage lives entirely in interference aimed by problem structure. The vastness is real, decisive, and unreachable: workspace beyond any classical memory, storage for nothing.

Next, in Chapter 6, "Correlations Without a Mechanism": the failed copier left us holding two qubits that agree with certainty and never once disagree, no matter how far apart you carry them. Einstein spent his last decades objecting to what that state implies. We will define it honestly, see what it can and cannot do, and put the "entangled qubits communicate" myth where the others went.

Correlations Without a Mechanism

In May 1935, Albert Einstein, Boris Podolsky, and Nathan Rosen published four pages in Physical Review arguing that quantum mechanics, whatever its successes, had to be incomplete. Their exhibit was a pair of particles prepared so that measuring one appeared to settle the other, instantly, at any distance. Surely, they argued, the answers were fixed all along, and the theory was simply blind to the ledger. Einstein later compressed the objection into the phrase that has sold a million bad headlines: spooky action at a distance.

The argument ran for eighty-seven years. It became arithmetic in 1964, laboratory data in the 1980s, was settled loophole-free in a Delft basement in 2015, and collected the Nobel Prize in 2022. The disputed phenomenon is entanglement, and you are already holding the disputed object: Chapter 5's failed copier manufactured it, two qubits described by $(1/\sqrt{2},\ 0,\ 0,\ 1/\sqrt{2})$, agreeing forever, disagreeing never. This part defines what that state is, grants Einstein's instinct its strongest modern form, breaks it with arithmetic you can check at your desk, and closes the file on the last myth standing: that entanglement lets qubits communicate.

The state the copier left behind

Start with what the four numbers say. The state puts amplitude $1/\sqrt{2}$ on the string 00, the same on 11, and exactly zero on 01 and 10. Measure both qubits, anywhere, any time, and the Born rule pays out 00 half the time, 11 the other half, and a disagreement never. Separate the qubits by a corridor, a continent, or 1.3 kilometers of Dutch campus, and the agreement holds. So far, striking but not yet strange; correlated randomness is an old story.

The strangeness is in what the state refuses to be. Two independent qubits, one holding the pair (a_0, a_1) and the other (b_0, b_1), have a joint description you can always build by multiplication: the amplitude of each string is the product of its parts. Run that requirement against our state and it fails in four lines; the arithmetic box below has them. No assignment of individual pairs multiplies out to $(1/\sqrt{2}, 0, 0, 1/\sqrt{2})$. That is the definition, and it deserves stating in full: **a joint state is entangled when it cannot be factored into separate descriptions of its parts.** The pair has an amplitude description. Neither member has one of its own.

Sit with that sentence, because nothing in classical engineering prepares you for it. Every distributed system you have ever operated is exactly the sum of its node states plus correlations you could log. Here, the complete description of the whole exists, is four numbers long, and contains no page for either half. Ask what qubit A alone is doing and the honest answer is a shrug wearing mathematics. Physics does hand you a local description, but it is a statistical one, a fair coin in every direction you can measure, carrying none of the correlation. The amplitude pair, the thing this series has used to describe a qubit since Chapter 3, simply does not exist for either half.

THE ARITHMETIC

The factorization test.

Suppose the state did factor: qubit A carries (a_0, a_1), qubit B carries (b_0, b_1), and joint amplitudes are products. Matching our state requires four equations: $a_0 b_0 = 1/\sqrt{2}$, $a_0 b_1 = 0$, $a_1 b_0 = 0$, $a_1 b_1 = 1/\sqrt{2}$.

Take the second equation. A product is zero only if a factor is zero, so $a_0 = 0$ or $b_1 = 0$. If $a_0 = 0$, the first equation reads $0 = 1/\sqrt{2}$. If $b_1 = 0$, the fourth reads $0 = 1/\sqrt{2}$. Both branches contradict. No factorization exists, for this state or for anything within reach of it. Four lines, one shrug, and a new kind of object.

The gloves objection

Your instinct has an answer ready, and it deserves the floor, because it is Einstein's answer. Take a pair of gloves, seal the left in one box and the right in another, ship one to Geneva and one to Singapore. Open either box and you know, instantly, at any distance, what the other contains. No spookiness, no signal, no physics beyond a shared past: the answers were fixed at the source, and "instantly" describes your bookkeeping, never the gloves. John Bell himself liked to make the point with a colleague's famously mismatched socks: see one pink sock, deduce the other, alert no journalists.

Now be precise about what the gloves model claims for our qubits. At the source, the pair flips one coin and writes the result into both qubits: both carry 0, or both carry 1. Every measurement then reads a pre-written answer. Run this model against everything in the previous section and it passes in full. Perfect agreement: reproduced. Fifty-fifty marginals: reproduced. Distance-proof correlation: trivially reproduced, the way your two backup tapes agree without a network between them. The factorization failure does not rescue us either, since the gloves model was never a pair of amplitude pairs to begin with; it is Chapter 2's picture, definite values plus shared history, and Chapter 4 killed that picture for one qubit's interference, never for two qubits' correlations.

Honesty first, as always in this series: for every experiment described so far, the gloves win. If entanglement only ever produced agreement on one fixed question, Einstein would be right, the state would be a fancy ledger, and this article would end here. So far, the gloves win.

Bell's arithmetic at 120 degrees

What John Bell saw in 1964, in a paper physics took years to notice, is that the gloves model signs a contract it cannot keep once the qubits can be asked more than one question. The upgrade costs nothing we do not own. Chapter 3 measured every qubit against

one fixed question; a rotated question is available inside the same real-number arithmetic, and measuring at angle θ means comparing the state against the tilted pair $(\cos\theta, \sin\theta)$. One imported fact, derivable in four lines from the linearity you already use: measure the two halves of our state at angles α and β, and they agree with probability $\cos^2(\alpha - \beta)$. Same angle, agreement always, matching everything above. The correlation cares only about the difference between the questions.

Bell's move, in the three-question form David Mermin later popularized, is to allow each side three questions and force the recipe to commit. The arithmetic box runs it in full, and the totals are these. The qubits, asked two different questions, agree one time in four. Any conceivable pre-written recipe, asked two different questions, agrees at least one time in three. A third against a quarter. The gap is small, unforgiving, and measured.

Laboratories spent three decades closing the escape hatches, from the Paris experiments of the early 1980s to the loophole-free test at Delft in 2015, which violated the bound with entangled electron spins 1.3 kilometers apart, far enough that no light-speed signal could coordinate the answers. The 2022 Nobel Prize went to Aspect, Clauser, and Zeilinger for that experimental program. Einstein's instinct, given every courtesy and fifty years of engineering, loses to the pigeonhole principle. The recipe model is dead.

THE ARITHMETIC

One-third versus one-quarter.

Three questions, at angles 0°, 120°, and 240°. Geneva and Singapore each pick one at random for every entangled pair and log the answers.

The quantum tally for different questions: the angle difference is 120° or 240°, and $\cos^2(120°) = (-1/2)^2 = 1/4$, same for 240°. Agreement rate: one in four.

The recipe's best effort. To reproduce the observed perfect agreement whenever the questions match, both qubits must

leave the source carrying identical pre-written answers to all three questions: a shared triple, like (0, 1, 1). Now the pigeonhole: a triple has three entries and only two possible values, so some two entries are equal, meaning at least one of the three question-pairs agrees in advance. Check the cases yourself. The triples 000 and 111 agree on all three pairs. Every mixed triple, like (0, 1, 1), agrees on exactly one pair of the three. There is no third kind of triple. So when two different questions are chosen at random, pre-written answers agree with probability at least $1/3$.

Floor for any recipe: one in three. Measured rate: one in four. You cannot write a recipe that loses to its own pigeonhole, and nature does. (The $\cos^2$ correlation is the one fact imported here on credit; it follows from Chapter 3's rules in four lines, and every published Bell test is, in effect, the universe grading that homework.)

WHERE THIS BREAKS

The gloves, retired with honors.

The gloves analogy is the correct teaching tool for what entanglement is *not*, which is why this article leaned on it, and it must be retired the moment the 120-degree data arrives. Gloves are Chapter 2's world: definite properties, fixed at the source, revealed on inspection. Bell's arithmetic shows the qubits' correlations exceed what any such ledger can produce, so the one thing the analogy exists to suggest, answers written in advance, is the one thing the experiment excludes. Real tests carry real fine print: detector efficiency, timing, and locality loopholes occupied experimentalists for decades, which is exactly why the 2015 loophole-free results, and the Nobel that followed, closed an argument rather than a seminar.

MYTH AUTOPSY

"Entangled qubits communicate."

The correlation is real, instantaneous-looking, and distance-proof, so the myth writes itself: measure here, affect there, message sent. It fails on arithmetic you can now do in one line. Whatever angle Geneva chooses, and whether Geneva measures at all, Singapore's local statistics never move: add the joint outcomes that leave Singapore reading 0 and you get $\cos^2(\alpha-\beta)/2 + \sin^2(\alpha-\beta)/2 = 1/2$, for every α on the dial. Singapore sees a fair coin before, during, and after anything Geneva does. No experiment on one side detects the other side's choice, which means no channel, no bandwidth, no bit.

The second failure is about control. Even granting the correlation, nobody chooses their outcome; Geneva gets a Born-random answer and so does Singapore, and you cannot encode a message in a value you do not select. The agreement is only visible when the two logs meet, and the logs travel by classical means, at light speed or slower. Quantum teleportation, despite the branding, pays the same toll: moving one qubit's state consumes one entangled pair and requires sending two ordinary classical bits, so nothing outruns light there either. Relativity files no complaint.

Say instead: entangled qubits are correlated in ways no pre-shared classical recipe can reproduce; nothing travels between them.

What entanglement is actually for

Strip the mysticism and entanglement is a resource with a job description. Inside a quantum computer it is the fuel: two-qubit gates create it, and a computation that never entangles anything can be simulated cheaply on the laptop you already own, which means the exponential workspace of Chapter 5 only opens for algorithms that entangle at scale. The register states inside Shor's algorithm

are entangled beyond any classical bookkeeping, and that is a requirement, never a side effect.

Entanglement also enforces the strangest access-control property in your future stack: monogamy. If two qubits are maximally entangled with each other, physics forbids either from being entangled with anything else; there is no BCC on a Bell pair. That exclusivity is why entanglement-based key distribution can promise eavesdropper detection with a straight face, and why a measured Bell violation can certify that randomness is fresh and unshared even when the hardware vendor is untrusted. Verification by physics rather than by attestation: your audit instincts should be circling that sentence.

And the same resource, pointed the wrong way, is the machine's failure mode. A qubit that entangles with a stray photon, a phonon, a fluctuating field, has formed a Bell-flavored bond with something nobody controls, and its interference dies exactly as Chapter 4's which-path test predicted. Under its proper name, decoherence, this is Chapter 10's central villain. Fuel one way, failure the other.

ATTACKER'S-EYE VIEW

No exploit in the correlation.

Clear the fantasy first, because it is sold in real rooms: there is no covert FTL channel here, no undetectable comms layer, no "quantum radio." No-signaling is an arithmetic identity, so any pitch involving instantaneous communication through entanglement, and such pitches exist at the confluence of defense budgets and physics envy, fails at the marginal-probability line above. File under Chapter 1's rule: ask at what, exactly.

The real security content runs the other way. Monogamy makes entanglement the one resource whose exclusivity is enforced by physics: an eavesdropper who entangles with a distributed pair necessarily degrades the pair's Bell score, so the security test and the physics test are the same number. That is the engine of entanglement-based QKD,

of which the E91-family protocols are the commercial descendants, and of device-independent randomness, where a Bell violation certifies fresh entropy even from a box you refuse to trust. I consider Bell-certified randomness the most under-appreciated security primitive of the coming decade, precisely because it moves a root-of-trust question from the vendor's documentation into the universe's arithmetic.

And for the defender of quantum systems, the villain has a face now: decoherence is uninvited entanglement, the environment forming Bell-flavored bonds with your register at gigahertz rates. Every shield, filter, and millikelvin in Chapter 10 exists to stop the universe from doing to your computer what Bell did to Einstein.

What to remember

Entanglement is a joint state that cannot be factored into individual descriptions; the pair has four amplitudes, and neither member has two of its own. The gloves model, shared answers fixed at the source, reproduces the easy correlations perfectly and deserves the respect of a precise statement, because Bell turned precisely that statement into arithmetic. Asked three questions at 120-degree spacing, any pre-written recipe agrees at least one time in three by pigeonhole, the qubits agree one time in four by the Born rule, and loophole-free experiments side with the qubits. Nothing communicates: local statistics never move, outcomes cannot be chosen, and the correlation exists only in logs that travel classically. In the machine, entanglement is fuel, monogamy is access control, and the same bond formed with the environment is the failure mode called decoherence.

Next, in Chapter 7, "The Shape of a Quantum Algorithm": everything is finally on the table: amplitudes, interference, a vast state, a narrow door, and entanglement to bind them. Chapter 7 assembles the pieces into the only recipe that works, prepare, imprint, interfere, measure, and shows why the machine that does everything at

once is useless until a problem's structure determines which outcomes cancel.

The Shape of a Quantum Algorithm

In 1985, David Deutsch published the first quantum algorithm. It answered a question nobody was asking, about a function nobody cared about, one bit at a time. It was also the blueprint for breaking RSA. Four decades of quantum algorithms later, from that toy to the machine your migration program exists to outrun, every single one of them is still Deutsch's moves in the same order.

This part assembles the toolbox into that shape. You hold all the pieces: amplitudes that carry signs, the collision that makes signs count (Chapter 4's engine), a workspace of 2^n amplitudes behind an n-bit door (Chapter 5), and entanglement to bind registers together. What follows is how quantum algorithms work: the four-move template every one of them obeys, a complete worked example small enough to run at your desk, and the reason the recipe pays out only when a problem brings structure to the table. By the end, the casualty list from Chapter 1 stops being a claim and becomes a mechanism.

Four moves

Prepare. Hadamards on every qubit spread amplitude evenly across all 2^n input strings. This is the honest half of the parallelism myth, and it costs almost nothing: one gate per qubit and the workspace is open.

Imprint. Run the problem through the register once, writing what the problem knows into the amplitudes, typically as a pattern of signs. A circuit that evaluates a function f can be compiled to flip the sign of every input string where f says 1. After this move the register holds the problem's entire truth table, encoded in signs. It also holds, per Chapter 3, nothing you can see: signs are invisible

to measurement, and if you measured now you would draw one random input, exactly the dilution-refrigerator-as-random.choice fiasco of Chapter 5's autopsy. Two moves in, nothing has been gained. This is worth saying plainly, because vendors never do.

Interfere. The closing transform. A second layer of gates makes the signed amplitudes collide, and here is the entire art form: the transform is chosen so that computational routes toward strings describing the problem's *global pattern* arrive in agreement and reinforce, while routes toward everything else arrive in conflict and cancel. Chapter 4's $+1/2$ against $-1/2$, scaled to millions of colliding routes. The transform is fixed in advance, built from the problem's *form* alone, because nobody knows the answer yet.

Measure. The narrow door opens once and pays out n bits. If the choreography was right, those bits name the global pattern, and the state is spent buying them.

That is the whole genre.

The whole game in one qubit

Deutsch's problem sounds like a pentest constraint. You are handed an oracle for an unknown one-bit function f: feed it 0 or 1, it returns $f(0)$ or $f(1)$. There are four possible functions: two constant (always 0, always 1) and two balanced (the identity, and NOT). Your question: is f constant or balanced? Your budget: one call.

Classically, one call is hopeless. Learn $f(0) = 0$ and the function could still be constant-zero or the identity; the question compares two values, and you have one. Any classical strategy needs both calls, full stop. The quantum run takes one, and it is the four moves verbatim. Prepare: a Hadamard puts the qubit at $(1/\sqrt{2}, 1/\sqrt{2})$. Imprint: one oracle call flips the sign of each input's amplitude where f outputs 1. Interfere: a second Hadamard. Measure: the bit reads 0 for constant, 1 for balanced, with certainty. The arithmetic box runs all four cases; it is sixteen additions, and I recommend actually doing them, because this is the smallest

complete quantum algorithm that will ever exist and it fits on an index card.

Two details in the outcome repay attention. First, you have run this algorithm before: Chapter 4's impossible coin was Deutsch's algorithm applied to the laziest function on the menu, constant-zero, whose oracle does nothing. The impossible coin was never a stunt; it was the genre's first rehearsal. Second, look at what you learned and what you paid. The output bit answers the *comparison*, constant versus balanced, a global property of f. Ask the algorithm what $f(0)$ actually was and it cannot tell you; that information fed the interference and was consumed by it. One call bought one bit, Holevo smiling in the background, and the algorithm's genius was spending its single bit on the question that needed two classical calls: one call, one bit, certainty.

THE ARITHMETIC

All four functions, sixteen additions.

The oracle rule: flip the sign of the amplitude at input x wherever $f(x) = 1$. Start every run at $(1, 0)$, Hadamard to $(1/\sqrt{2}, 1/\sqrt{2})$, apply the oracle, Hadamard again using Chapter 4's rule $a'_0 = (a_0 + a_1)/\sqrt{2}$, $a'_1 = (a_0 - a_1)/\sqrt{2}$, then measure.

Constant zero, no flips: $(1/\sqrt{2}, 1/\sqrt{2}) \to (1, 0)$. Output 0.

Constant one, both flipped: $(-1/\sqrt{2}, -1/\sqrt{2}) \to (-1, 0)$. The Born rule squares: output 0 with certainty. An overall minus sign in front of everything is invisible; only *relative* signs ever fight.

Identity, second amplitude flipped: $(1/\sqrt{2}, -1/\sqrt{2}) \to (0, 1)$. Output 1.

NOT, first amplitude flipped: $(-1/\sqrt{2}, 1/\sqrt{2}) \to (0, -1)$. Output 1, certainty again.

Constant functions output 0, balanced functions output 1, one oracle call each. (One import on credit, in this series'

> tradition of labeling them: turning an ordinary circuit for f into a sign-flipper is a standard compilation, one helper qubit and a trick called phase kickback, at the cost of exactly one evaluation of f.)

Credit where the history puts it: Deutsch's 1985 version answered with certainty only half the time, saying "don't know" otherwise; the crisp one-call form above is the 1998 refinement by Cleve, Ekert, Macchiavello, and Mosca. That last name is doing double duty in this series; you met Mosca's inequality in Chapter 1, setting your migration deadline. The man tuning the first quantum algorithm and the man timing your response to it are the same person, which tells you something about how small and how serious this field is. Deutsch and Jozsa had already scaled the problem to n-bit functions in 1992, one call replacing up to $2^{n-1}+1$, and the template never changed.

Why structure is the entry fee

Now the constraint that shapes the entire casualty list. The interfere move is a fixed transform, designed before anyone knows the answer, from nothing but the problem's form. For the design to exist at all, right answers must be distinguishable from wrong ones by their *relationship to the whole input space*: a period, a parity, a constant-versus-balanced split, some global regularity that a pre-committed transform can convert into concentrated amplitude. Deutsch's problem is the minimal case, one bit of global pattern. Shor's algorithm is the maximal one, where the pattern is the period hiding inside modular arithmetic, and the closing transform is a Fourier transform, the mathematical instrument whose one job is finding periods.

Take the structure away and the recipe starves. A generic search problem, find the preimage, guess the key, satisfy the formula, offers no global regularity: the right answer relates to the wrong ones by nothing but being right. No pre-committed transform can aim cancellation at "everything except the answer" without knowing

the answer, so the choreography has nowhere to stand, and the only quantum recourse is a generic amplitude-nudging trick worth a quadratic speedup and no more, which is Chapter 9's whole story. This is the mechanism under the sentence Chapter 1 asserted on credit: structure is what the machine eats. The same fact is a compliment your field has earned. Symmetric cryptography holds up in the quantum era for the same reason it held up in the classical one: cryptographers spent half a century sanding every exploitable pattern off it. AES resists the choreography because there is nothing regular for the choreography to aim at. No pattern, no algorithm.

The myth's replacement is now complete, in the final form Chapter 5 promised. The machine really does imprint every input's contribution at once, and that costs one evaluation and buys nothing. Advantage exists exactly when a closing transform can convert the spread into one readable global fact, and the door lets exactly that fact out. Parallelism is free and worthless. Interference, aimed by structure, is the entire product.

WHERE THIS BREAKS

Two analogies, retired on schedule.

Choreography has carried this series since Chapter 1, and its limit is that dancers improvise while gates never do: every transform is a fixed, deterministic, linear operation, committed before the music starts. The art lives entirely at design time, in the mathematician's choice of transform; at runtime the machine executes arithmetic with the creative latitude of a payroll system. And the noise-cancelling headphone, the other intuition on loan here, borrows a medium that does not exist: sound waves cancel in air, while amplitudes cancel in nothing at all. What interferes in a quantum computer is possibility itself, with no substance doing the waving, which is precisely why no classical wave gadget, however clever, reproduces Chapter 4's bench.

FOR THE RECORD

What the machine cannot do, complexity edition.

One box of complexity theory, as promised, and then the series drops the subject. The problems quantum computers handle efficiently form a class called BQP. It contains everything classical computers do efficiently, and it is believed, on the strength of factoring, to reach beyond. It is *not* known, or seriously believed, to contain the NP-complete problems: scheduling, routing, generic SAT, the travelling salesman. No proof exists either way, but four decades of trying have produced no structured quantum attack on any NP-complete problem, and Chapter 9's quadratic ceiling is the best generic result anyone has. When a pitch deck promises quantum computers will "solve optimization," apply Chapter 1's reflex: at what, exactly, and where is the closing transform?

FOR THE RECORD

The parallel-universes clause.

David Deutsch, who built the first algorithm above, holds that the honest reading of all this arithmetic is that the computations occur across parallel universes, and he argues it seriously; many-worlds is a real position held by serious physicists. This series takes no side, for an operational reason rather than a diplomatic one: interpretations are stories about what amplitudes *mean*, engineered to make identical predictions, and every one of them must reproduce the sixteen additions in the box above, digit for digit. The universes, if you keep them, are a picture attached to the arithmetic, never a mechanism added to it; nothing computes faster because a story got grander. Whatever story you attach, the table is the table.

ATTACKER'S-EYE VIEW

A targeting doctrine, and a fraud detector.

The template doubles as an evaluation framework, and I suggest running every "quantum threat to X" claim through it. Two questions: what global structure does X expose, and what closing transform converts that structure into a readable answer? RSA answers both, period and Fourier transform, and dies in Chapter 8. AES answers neither and gets a quadratic dent in Chapter 9. When the next paper claims a quantum attack on some protocol, find the imprint and find the transform; where there is no transform, there is no threat, only a press release wearing equations.

The same lens catches the most common accounting fraud in quantum claims. I read every quantum-speedup paper with one finger on the oracle, because the imprint step is where costs hide: if "loading the data" into the register quietly costs what the classical computation cost, the exponential speedup is a shell game, and a noticeable fraction of quantum machine-learning claims over the past decade have died in exactly that audit. The oracle is an assumption wearing a lab coat. Check its invoice.

What to remember

Every quantum algorithm is four moves: prepare a uniform superposition, imprint the problem into the signs at the cost of about one evaluation, interfere with a transform fixed in advance, and measure the one global fact the choreography concentrated. Deutsch's problem runs the entire shape in one qubit, answering in one call a question that classically takes two, and paying for it precisely: the algorithm learns the comparison and forfeits the values. Advantage requires structure, because the closing transform must aim cancellation using the problem's form alone, which is why RSA's periodic arithmetic is fatal exposure and AES's deliberate patternlessness is quantum resistance by fifty years of habit. Parallelism is

free and buys nothing; interference aimed by structure is the entire product. And no interpretation, universes included, adds a single addition to the arithmetic.

Next, in Chapter 8, "Shor: Cryptanalysis by Physics": Deutsch asked one bit about one toy function. Chapter 8 aims the same four moves at a 2048-bit modulus, where the global pattern is a period hiding inside modular exponentiation and the closing transform is the quantum Fourier transform. The casualty list stops being a table and becomes a walkthrough.

Shor: Cryptanalysis by Physics

The security of RSA rests on a bet: that multiplying two large primes is easy and reversing the product is hopeless. The bet has held for nearly fifty years against the finest classical mathematicians alive, and it still holds against every classical computer on Earth. In 1994, Peter Shor showed the bet was never made against the right opponent. His algorithm does not factor faster than classical methods by brute persistence. It reaches factoring through a side door, period-finding, and that door happens to be exactly the shape of the machine Chapter 7 built.

This is where the casualty list from Chapter 1 stops being a table you take on faith and becomes a mechanism you can follow. The four moves are unchanged: prepare, imprint, interfere, measure. Only the target grew, from Deutsch's one-bit toy to a 617-digit modulus. This part shows how factoring becomes period-finding, how the quantum Fourier transform reads a period the way a tuner reads a pitch, why the same move flattens elliptic curves and Diffie-Hellman even faster than it flattens RSA, and what it honestly takes to run, in physical qubits, as of 2025. No new physics past this point. Only the target.

The number-theory bridge, in one example

Shor's first move is pure classical mathematics, and it converts a factoring problem into a hunt for a repeating pattern. This part is not quantum at all, which is exactly why it matters: it explains why factoring, of all things, exposes the periodic structure the machine feeds on.

Take a number to factor, and call it N. Pick any number a smaller than N that shares no factor with it, and start raising it to powers, reducing modulo N at each step: $a^1, a^2, a^3, \ldots$, always taking the remainder after dividing by N. Because there are only finitely

many possible remainders, the sequence must eventually repeat, and once it repeats it cycles forever. The length of that cycle (the gap after which the powers of a start over, and the only quantity in this whole attack that needs a quantum computer to find) is called the period, written r. The arithmetic box works a small case by hand.

Here is the bridge, and it is elementary modular arithmetic, not quantum mechanics: the difference of squares $a^r - 1 = (a^{r/2} - 1)(a^{r/2} + 1)$. When the period r is even and $a^{r/2} \not\equiv -1 \pmod{N}$, at least one of $a^{r/2} - 1$ and $a^{r/2} + 1$ shares a real factor with N, which you pull out in seconds by taking $\gcd(a^{r/2} \pm 1,\ N)$ with Euclid's algorithm, the oldest algorithm in the book. When either condition fails you pick a new a and rerun; random choices succeed with substantial probability. Factoring, reduced to a single unknown: the period r. Everything hard about breaking RSA now lives in one question, and it is a question about the length of a repeating cycle.

Classically, that reduction buys nothing, because finding r for a cryptographic modulus means marching through a cycle longer than the age of the universe measured in machine cycles. The period is there. It is simply too long to walk. What Shor found is that a quantum computer does not walk it. It listens for it.

THE ARITHMETIC

A period you can find by hand.

Factor $N = 15$ with $a = 2$. Raise and reduce mod 15:

$2^1 = 2, \quad 2^2 = 4, \quad 2^3 = 8, \quad 2^4 = 16 = 1, \quad 2^5 = 2, \ldots$

The sequence is $2, 4, 8, 1, 2, 4, 8, 1, \ldots$, repeating every four steps. The period is $r = 4$, which is even, and $a^{r/2} = 2^2 = 4 \not\equiv -1 \pmod{15}$, so both success conditions hold. Now take gcds: $\gcd(4 - 1,\ 15) = 3$ and $\gcd(4 + 1,\ 15) = 5$. Both are genuine factors of 15, and $3 \times 5 = 15$. The whole difficulty was finding that the cycle length is 4. For a 2048-bit N the cycle is astronomically long, the schoolbook steps are identical,

and only the period-finding needs a quantum computer.

Reading a period with interference

The period-finding move is the four-move template of Chapter 7, and now the abstract description acquires flesh. Watch the amplitudes.

Prepare. Fill a large input register with an equal superposition of the exponents $1, 2, 3, \ldots$, far enough to cover many full cycles of the unknown period. Amplitude spreads across every exponent at once, the honest half of parallelism.

Imprint. In one pass, compute a^x mod N into a second register. This is the expensive step, the modular exponentiation, and it is where nearly all the qubits and nearly all the runtime go; the closing sections are honest about that cost. The two registers are now entangled: each exponent in the first is bound to its power's remainder in the second. The periodic structure of a is now physically imprinted, because exponents exactly one period apart produced identical remainders and are therefore tied to the same value in the second register.

Interfere. Apply the quantum Fourier transform to the input register. Every exponent that shares a remainder, meaning every exponent standing one period apart from its neighbors, sends amplitude into the transform, and those contributions reinforce only at output values that match the period's rhythm. Every other output receives contributions that point in conflicting directions and cancel. Chapter 4's $+1/2$ against $-1/2$, now playing out across the whole register at once. The wrongness cancels; the period survives.

Measure. The input register is read, and it does not hand you r directly. It yields a sample sharply peaked near an integer multiple of the pattern's fundamental frequency. One more piece of classical arithmetic, the continued-fractions algorithm, turns that reading into a candidate for r, which may be the period or a divisor of it. You check the candidate by modular exponentiation and

rerun the quantum subroutine when it misses. Hand a good r to the difference of squares, run Euclid, and RSA is factored.

WHERE THIS BREAKS

The tuner, and where the metaphor stops.

The quantum Fourier transform earns the tuner analogy honestly: a guitar tuner samples a messy sound wave and reports one number, the pitch, because a Fourier transform converts a signal from "value at each moment" into "strength at each frequency," and a repeating signal lights up at its own frequency. The QFT does exactly that to the periodic pattern in the register, which is why Chapter 7's claim that the closing transform must match the problem's structure is literal fact here: factoring hides a period, so the period-finder is the tool.

First, a tuner's microphone can sample the wave as often as it likes; the QFT gets Chapter 5's single destructive read, which is why the algorithm reruns to collect enough peaks for the continued-fractions step to lock on. Second, and more fundamental, nothing is oscillating. A guitar string really does vibrate in air; here the "signal" is a pattern in amplitudes, the "frequency" is a mathematical property of modular arithmetic, and no physical wave exists anywhere in the machine. The tuner is how the math feels. It is not what the hardware is doing.

Why the whole family falls together

The lethal generality of Shor's method comes from what periodicity is: the mathematical signature of the hidden-subgroup problem, and every classical public-key system in wide deployment today, RSA and the discrete-log family alike, is built on a member of it. Break the pattern-finding and you break all of them, at a stroke, on the same machine. The post-quantum algorithms replacing them are public-key too, built on different hard problems Shor

does not touch, which is the entire point of the migration.

RSA hides a period in modular exponentiation, as above. Finite-field Diffie-Hellman and DSA rest on the discrete logarithm problem, which is period-finding wearing a different hat; Shor's algorithm solves it with the same interference machine, no new ideas required. And elliptic-curve cryptography, ECDH and ECDSA, the long-standing default behind conventional TLS handshakes, device identity, and the key material in your phone's secure enclave, rests on the elliptic-curve discrete logarithm. Shor breaks that too. Deployments have already begun wrapping ECDH in hybrid post-quantum key exchange, Signal's PQXDH among them, which is exactly the right response and also proof that the industry is not waiting for the machine. And here is the cruel twist your threat model needs to internalize: where classical ECDH is still in use, elliptic curves fall to *smaller* quantum computers than RSA does.

That inverts the classical security story completely. Against classical attackers, a 256-bit elliptic-curve key is comfortably stronger than a 3072-bit RSA key while being twelve times shorter, which is precisely why the industry has spent fifteen years migrating to it. Against a quantum attacker, the elliptic-curve group's tidy mathematical structure, the very compactness that made it the efficient choice, makes it a cheaper target. Elliptic curves' greatest classical strength is a quantum liability. The migration that hardened you against the last threat model positioned you slightly worse for this one, which is the kind of irony cryptography specializes in and a clean illustration of why "quantum-resistant" is a property no amount of classical key length can buy.

What it actually takes

The gap between "the algorithm exists" and "your keys are readable" is the only thing standing between you and Q-Day, and it is measured in physical qubits.

Shor's algorithm needs roughly $2n$ high-quality logical qubits to factor an n-bit number: a few thousand flawless logical qubits for RSA-2048. No such machine exists in 2025; the largest processors

today carry a few hundred to low-thousands of *physical* qubits with error rates far too high to run Shor's thousands of sequential operations, a distinction that is Chapter 10's entire subject and the reason the casualty list comes with a "when" attached to the "whether." The translation from logical to physical is where the real number lives, because each logical qubit must be woven from many noisy physical ones through error correction.

That translation is a moving target, and reading it correctly means comparing like with like. The clean comparison is the surface-code line. In 2019, Gidney and Ekerå estimated 20 million noisy physical qubits and roughly eight hours. In May 2025, Gidney cut that to under one million physical qubits and under a week, on the same hardware assumptions, from better arithmetic and error correction alone. Nothing about the hardware improved between those two papers. The estimate improved, because people got smarter about the same machine.

By 2026 the frontier had moved again, and this time by changing the architecture rather than the assumptions. The Pinnacle preprint replaces surface codes with quantum LDPC codes and claims RSA-2048 with under 100,000 physical qubits at the same headline error and timing. A separate neutral-atom study from Cain and colleagues argues that cryptographically relevant Shor could run on as few as 10,000 reconfigurable atoms, trading qubit count for much longer runtimes. Both are theoretical preprints on architectures nobody has built, using different codes, connectivity, and time budgets, so they are not directly comparable with Gidney's surface-code baseline or with each other. What they establish is not a new threshold. It is that architecture and error-correction choices can move the number by another order of magnitude before a single qubit is added to a real machine.

Do not read any of these as a countdown. A fault-tolerant machine of even 100,000 high-quality qubits is a formidable distance from today's hardware, and I am on record against the quantum panic industry that reads every such paper as proof of imminent doom. Read the trajectory instead, the way a risk officer reads any hostile capability curve: the requirement has fallen by five orders of magnitude in fourteen years while qubit counts climb, and your

defensive timeline is not set by when the lines cross. It is set by harvest now, decrypt later, which put your long-lived secrets on the clock the day they were transmitted, and by the migration deadlines regulators and insurers have already written without waiting for the physicists. The math above is why RSA and ECC are on the list. The reason to act is on a different clock entirely.

FOR THE RECORD

Physical, logical, and the number that matters.

Three numbers get conflated in headlines, and the distinction is the difference between panic and planning. *Logical qubits*: RSA-2048 needs a few thousand, roughly $2n$ for an n-bit modulus. *Physical qubits*: how many noisy devices error-correct those logical qubits, and the ratio is not a constant. Gidney's 2025 surface-code estimate put it under one million; the 2026 Pinnacle qLDPC preprint claims under 100,000 by changing the code; a neutral-atom study reaches as few as 10,000 by accepting far longer runtimes. *Today's machines*: hundreds to low-thousands of physical qubits, error rates too high for deep circuits. When a headline says a processor "hit 1,000 qubits," those are physical, unfit for Shor, and short of even the most aggressive fault-tolerant estimate by one to two orders of magnitude. A "qubits required" number with no code, error model, and runtime attached has omitted most of the estimate. Re-verify when citing; the figure has moved three times in a decade and will move again.

ATTACKER'S-EYE VIEW

What to do before the machine exists.

The rational adversary already runs the profitable half of this attack, and it needs no quantum computer. Harvest now: capture and warehouse ECDH and RSA-encrypted traffic today, decrypt it the week a cryptographically relevant machine switches on. Every ciphertext whose value outlives

> the machine's arrival is already, in effect, compromised; the decryption is merely scheduled. That is not a future threat to model. It is a present exfiltration to assume, and it makes data with a long confidentiality lifetime, state secrets, health records, identity keys, the first thing to move.
>
> The defender's list writes itself from the mechanism. Inventory everything riding on RSA, finite-field Diffie-Hellman, and every elliptic curve, because all of it is on the same list and falls to the same machine; ECC is not a hedge against RSA's exposure, it is a slightly cheaper target. Prioritize by confidentiality lifetime, not by perceived Q-Day distance, since HNDL already erased that distinction. Demand crypto-agility in everything you procure, so the eventual swap to NIST's post-quantum standards is a configuration change rather than a decade-long forklift. The one defense the mechanism rules out is waiting for certainty, because certainty arrives the same week your keys become readable, and that is far too late to start.

What to remember

Shor's algorithm breaks RSA not by factoring faster but by converting factoring into period-finding, a conversion built from elementary modular arithmetic that turns the whole problem into one question: the length of a repeating cycle. That period is found by the four-move template, with modular exponentiation as the imprint and the quantum Fourier transform as the interference step that lets period-matching outputs reinforce while everything else cancels. The same method solves the discrete logarithm, so finite-field Diffie-Hellman, DSA, and all elliptic-curve cryptography fall to the same machine, with elliptic curves falling at smaller sizes than RSA because their compact structure is a quantum liability. Running it against RSA-2048 needs a few thousand logical qubits. On the like-for-like surface-code comparison, the physical qubit requirement dropped twentyfold from 2019 to 2025; architecture-specific 2026 preprints claim further reductions that are not yet

commensurable. Read the trajectory rather than any single number. The reason to migrate now is not that the machine is close; it is that harvest-now-decrypt-later and regulatory deadlines already run on a clock that ignores the physicists.

Next, in Chapter 9, "Grover and the Honest Speedup": Shor breaks the systems with structure. Chapter 9 covers the systems without it, where the only quantum attack is a generic search that runs quadratically faster than brute force, buys far less than the hype suggests once error correction is priced in, and explains precisely why your symmetric keys and hashes step into the quantum era with room to spare.

Grover and the Honest Speedup

Every time I walk a security team through Shor's algorithm, the same hand goes up before I finish. If quantum computers eat RSA and elliptic curves, what do they do to AES, and do we need 512-bit keys?

The textbook answer takes one line. Grover's algorithm searches an unstructured space quadratically faster, so a 128-bit key offers 64 bits of security against a quantum attacker and a 256-bit key offers 128. Double your keys and sleep soundly.

Now read what the NSA actually mandates. CNSA 2.0, the requirement set for US national security systems, holds every other primitive to the highest parameters on the menu: ML-KEM-1024, ML-DSA-87, SHA-384 or better. For symmetric encryption it asks for AES-256 and nothing more. No larger variant is demanded, no asterisk, no urgency. Notice that this is entirely consistent: even on the textbook accounting, AES-256 leaves a notional 2^{128} of work, which is a fine place to stand. What is missing is any sign that the agency treats the halving as a live operational problem, and the reason is that the halving figure was never an attack estimate in the first place.

It is a query count. This part is about the difference: what Grover does, why its speedup provably stops where it does, why almost no attacker can collect even that, and what your symmetric keys should actually be.

What Grover actually does

Grover's algorithm, published by Lov Grover in 1996, answers the question Shor cannot touch: finding a needle in an unstructured haystack. Given a function that says yes to exactly one input out of N and no to everything else, find the yes. Key search is the canonical instance: the function reversibly tests a candidate key k against

known plaintext-ciphertext pairs, enough of them to pin down one key (a single 128-bit block leaves far too many AES-256 keys consistent with it), and the haystack is every key in the space.

Classically, you have no traction. The right key relates to the wrong keys by nothing except being right, so you try candidates until one works, averaging $N/2$ attempts. Chapter 7 explained why a quantum computer cannot do to this problem what it does to factoring: with no global structure to imprint, there is no closing transform that concentrates amplitude onto the answer in one shot. AES is a highly structured algorithm. What it lacks, by fifty years of deliberate design, is structure an attacker can exploit to beat generic search, and that absence is the whole reason Grover is the best anyone has against it.

What Grover does instead is patient. The four moves are there, and the third one runs in a loop. Prepare the uniform superposition over all N keys, each carrying amplitude $1/\sqrt{N}$. Imprint by flipping the sign of the correct key's amplitude, which the oracle does without anyone knowing which key it flipped. Then interfere, using an operation called inversion about the mean: compute the average of all amplitudes and reflect every amplitude through it. The flipped one sits below the average, so reflection kicks it up; the rest sit slightly above, so reflection nudges them down. Repeat. Each pass, a little more amplitude migrates to the answer and a little more drains from everything else. Do this about $\sqrt{N}$ times and the answer's amplitude is close to one. Measure, and you get the key.

Nothing was tried at once. Amplitude was pumped, one careful rotation per iteration, until the Born rule had no choice.

THE ARITHMETIC

One Grover iteration, by hand.

Four keys, one correct, say key number four. Start uniform: every amplitude is $1/2$, since $4 \times (1/2)^2 = 1$.

$$\left(\tfrac{1}{2},\ \tfrac{1}{2},\ \tfrac{1}{2},\ \tfrac{1}{2}\right)$$

Imprint. Flip the sign on key four: $(1/2,\ 1/2,\ 1/2,\ -1/2)$.

Interfere. The mean of those four numbers is $(0.5 + 0.5 + 0.5 - 0.5)/4 = 1/4$. Reflect each amplitude through the mean, using $a \to 2(\text{mean}) - a$:

Keys one to three: $2(1/4) - 1/2 = 0$. Key four: $2(1/4) - (-1/2) = 1$.

Final state $(0, 0, 0, 1)$. One iteration, and the wrong keys have cancelled to nothing while the right key carries all the amplitude. Measure and the answer is certain.

Why $\sqrt{N}$ in general: each iteration rotates the state by a small fixed angle of roughly $2/\sqrt{N}$, and you need to travel a quarter turn to swing the state onto the answer. Divide $\pi/2$ by $2/\sqrt{N}$ and you get about $0.79\sqrt{N}$ iterations. For a 128-bit key, $\sqrt{N} = 2^{64}$. Hold that number.

Quadratic is the ceiling, and that is a theorem

Before the good news, the bad. Nobody is going to find a better generic search algorithm, because the quadratic limit is proved rather than assumed. In 1997, Bennett, Bernstein, Brassard, and Vazirani showed that no quantum algorithm can search an unstructured space in fewer than about $\sqrt{N}$ queries, and Christof Zalka tightened it in 1999 to show Grover already hits the bound exactly. This is the anti-denialism half of the ledger: the quadratic speedup is real, it is optimal, and it is not going away.

It is also the ceiling. A quantum computer cannot search a structureless space faster than the square root of its size, which means that against symmetric cryptography, the machine that eviscerates RSA is reduced to a modestly better key-guessing engine. Everything hangs on the word "modestly," and on a detail the textbook line drops.

Two to the sixty-four, one after another

That 2^{64} is not a count of operations you can spread across a data center. It is a count of iterations that must run in strict sequence, on one coherent machine, without interruption.

The reason traces to Chapter 3. Each iteration transforms the amplitudes produced by the previous one, and you cannot checkpoint the intermediate state, because reading it destroys it and copying it is forbidden. There is no save file, no map-reduce, no "throw a thousand machines at it." Split the keyspace across M quantum computers and each searches a smaller haystack, which sounds like a win until you do the arithmetic: each machine still needs $\sqrt{N/M}$ iterations, so the *speedup* scales as $\sqrt{M}$ rather than M, while the aggregate oracle work goes up. A thousand machines buy you a factor of thirty-two in wall-clock time. A million buy you a thousand. Parallelism, the resource that makes classical brute force frightening, is precisely the resource quantum search cannot use efficiently.

So the attack is a single quantum computer, error-corrected, running 2^{64} iterations back to back, each iteration executing a full AES circuit in superposition. Every one of those logical operations is built from a mountain of error-corrected physical ones (Chapter 10's subject), and the machine must hold coherence through all of them. Put plausible numbers on the clock, as the box below does, and the wall time for AES-128 comes out in the billions of years.

This is not my private skepticism. It is the reason NIST introduced a parameter called MAXDEPTH into the post-quantum standardization criteria: a cap on how long a serial quantum computation any real attacker could plausibly run. Impose that cap and the attacker must parallelize, and parallelizing surrenders the quadratic advantage. NIST's own security categories bake this in, and the agency stated plainly that its reference primitives give substantially more quantum security than a naive reading suggests. Grassl, Langenberg, Roetteler, and Steinwandt built the first detailed quantum circuits for AES to make these estimates concrete, and Jaques, Naehrig, Roetteler, and Virdia later costed depth-limited attacks

properly, finding that the required parallelization drives the gate cost up rather than down.

To put it bluntly: the halving figure counts ideal oracle queries, not the cost of an attack anyone knows how to mount, and repeating it as though the two were the same is how a conservative number turns into a frightening one. Cryptographer Filippo Valsorda made the same argument this year, and reached the same place: quantum computers are not a practical threat to 128-bit symmetric keys.

THE ARITHMETIC

One illustrative wall clock.

AES-128 under Grover needs about $2^{64} \approx 1.8 \times 10^{19}$ iterations, in sequence. Each iteration runs at least one AES-128 encryption as a reversible quantum circuit; published circuits are thousands of logical layers deep, so call it 10^4 sequential logical operations per iteration, which is generous to the attacker.

Total serial depth: $1.8 \times 10^{19} \times 10^4 \approx 1.8 \times 10^{23}$ logical operations, strictly one after another.

Now the clock. Take Chapter 8's hardware assumptions, a one-microsecond surface-code cycle, and grant the attacker a full logical operation per microsecond, which is wildly optimistic. Then:

$$1.8 \times 10^{23}\ \mu s = 1.8 \times 10^{17} \text{ seconds} \approx 5.7 \times 10^{9} \text{ years}$$

Under these stated assumptions, roughly six billion years of uninterrupted, coherent, error-corrected computation. The universe is 13.8 billion years old. Grant a logical clock a thousand times faster and you still need six million years. Try to escape with parallelism and the $\sqrt{M}$ penalty demands on the order of 10^{19} separate fault-tolerant machines to finish inside

a year. Change the oracle depth or the clock and the figure moves, so do not quote six billion years as *the* answer. The durable lesson is the shape of the problem: the iterations are serial, each one is a full cipher circuit, and buying more machines barely helps.

AES-256 needs 2^{128} iterations. There is no point continuing the arithmetic.

(These are order-of-magnitude figures meant to be checkable, not a published resource estimate. The careful circuit-level analyses cited above reach the same verdict by more rigorous routes.)

WHERE THIS BREAKS

Two burglars, two methods.

This series has used a lock analogy since Chapter 1; both halves now have their mechanisms. Shor is the burglar who studied the lock's blueprint and machines a key from the design; no amount of adding pins helps, because the exposure is in the mechanism. Grover is the burglar who tries keys faster than you thought possible, which you answer the way locksmiths always have, by adding pins until trying is hopeless.

Where the analogy stops: Grover's "faster" is a square root, and square roots have the inconvenient property of being cheap to outrun. Every bit you add to the key doubles the classical attacker's work and adds a full bit back against the quantum one; a 256-bit key restores what a 128-bit key would notionally lose. Defense wins that arms race permanently, on paper, before the sequential-depth argument above even enters the room. Which is why the honest question was never whether Grover threatens AES. It was whether anyone can run Grover at all.

Why AES-256 is the conservative answer

Return to the opening. CNSA 2.0 demands maximum parameters for the primitives it considers exposed and leaves AES-256 alone, which tells you the agency protecting material for fifty years is not treating symmetric encryption as a fire. I cannot read NSA's mind about how much of Grover's asymptotic advantage it thinks is collectible, and neither can anyone else. What I can read is NIST's published costing model, which prices depth-limited attacks and makes the same point in the open: the query count is not the attack.

So here is the recommendation, and it is the same one you would get from CNSA 2.0, arrived at by a route that lets you defend it in a meeting. Use AES-256. Not because AES-128 is on the verge of falling, since the arithmetic above says it plainly is not, but because the margin costs you almost nothing, the guidance is unambiguous, and cryptographic history punishes people who cut margins to the theoretical minimum. Where AES-128 already protects short-lived data with a mature key hierarchy, treat the upgrade as hygiene rather than emergency. Reserve your emergency budget for the public-key migration, which is the actual fire.

Then take the sentence you now own into your next steering committee. AES comes through intact.

What Grover does to hashes

The same logic, one step sideways. Finding a preimage of a hash is unstructured search, so Grover applies: SHA-256 preimage resistance drops from 2^{256} to a notional 2^{128}, with all the sequential-depth caveats above still attached. Comfortable.

Collision resistance is the subtler case, and it is where a piece of quantum folklore deserves puncturing. Yes, an algorithm published by Brassard, Høyer, and Tapp finds collisions in roughly the cube root of the search space, which sounds alarming until you notice it demands quantum memory on the order of that same cube root. For SHA-256, that means storing and querying

something like 2^{85} quantum-accessible records, an object no more buildable than a quantum computer the size of Belgium. Meanwhile classical collision search, parallelized across ordinary hardware in the way van Oorschot and Wiener worked out in the 1990s, remains the practical attack. The quantum "speedup" on collisions is an artifact of counting queries and ignoring memory. SHA-256 holds. SHA-384 gives margin, which is why CNSA 2.0 asks for it.

MYTH AUTOPSY

"Quantum computers will break all encryption."

The headline that launched a thousand vendor decks dies on the casualty list, which you can now derive rather than memorize. Public-key cryptography built on factoring or discrete logs falls, completely, because periodic structure is exactly what interference can concentrate. Symmetric ciphers and hash functions hold, because no known structure in them yields a Shor-style shortcut, which leaves Grover pumping amplitude one sequential iteration at a time against a haystack that grows exponentially with every key bit you add. The threat is precise, and precision is the opposite of what "all encryption" implies.

The claim also fails a simpler test: the fix is already standardized. NIST's post-quantum standards replace the broken public-key algorithms with new classical ones that run on the hardware you own today, and AES rides through the transition untouched.

Say instead: quantum computers break the public-key algorithms that rest on factoring and discrete logs. Symmetric ciphers and hashes hold, with larger parameters as a margin.

ATTACKER'S-EYE VIEW

Nobody will Grover your AES key.

Here is the operational lesson to take upstairs, and it reframes the whole question that opened this article. Consider how that AES-256 session key came to exist. In a TLS handshake, it was agreed using elliptic-curve Diffie-Hellman, then used to encrypt the session. A quantum attacker with a recorded transcript does not attack the AES. They attack the handshake with Shor, solve the recorded elliptic-curve relation, derive the session key, and read the plaintext without ever touching the cipher. How long that takes depends on the machine and the architecture, and no honest number exists yet. It does not need to be hours. Your symmetric key is unbreakable and irrelevant, because the wrapper it arrived in is made of glass. The same holds for the file encrypted under AES-256 whose data key sits in an envelope sealed with RSA-2048. Hybrid post-quantum key establishment closes this exposure, which is exactly why your inventory has to record the handshake mode rather than assume every TLS session is alike.

That is why the migration is a key-establishment problem and a signature problem, and why an inventory that catalogues cipher suites without cataloguing key-exchange and certificate chains has inventoried the wrong thing. It is also why harvest now, decrypt later works on traffic protected by ciphers no quantum computer will ever break: the adversary is not harvesting your AES, they are harvesting the handshake in front of it.

The residual Grover risk worth a line in your risk register is narrower and duller: legacy 64-bit and 80-bit symmetric keys and short MAC tags in embedded, industrial, and telecom estates, where a square-root speedup meets a keyspace that was already marginal. Those deserve attention. AES-256 does not.

What to remember

Grover's algorithm attacks the problems Shor cannot, and it does so by pumping amplitude toward the answer through about $\sqrt{N}$

sequential iterations, giving a quadratic speedup that is provably optimal and provably capped. Those iterations must run in strict sequence on one coherent machine, because intermediate states cannot be saved or copied, and parallelism buys only a square root of what it buys classically. Priced honestly, the notional attack on AES-128 needs billions of years of uninterrupted error-corrected computation, which is why NIST wrote MAXDEPTH into its standards and why NSA asks for AES-256 rather than AES-512. Hashes hold on the same reasoning, with the quantum collision advantage gone once you account for quantum memory. And the practical threat to your AES-256 traffic is not Grover at all: it is Shor, breaking the key exchange that delivered the key.

Next, in Chapter 10, "The Machine Itself": every threat in this series has assumed a machine that does not exist, and now we build it. Decoherence as the environment measuring your qubits uninvited, error correction that checks integrity without ever reading the data, and why a raw physical-qubit count tells you almost nothing about a machine's cryptanalytic capability.

The Machine Itself

Ten millikelvin. That is the temperature inside the dilution refrigerator holding a superconducting quantum processor, roughly one hundredth of a degree above absolute zero, hundreds of times colder than the cosmic microwave background that fills the rest of the universe. The chip sits behind nested magnetic shields, in high vacuum, its control lines attenuated and filtered at every thermal stage. Enormous engineering effort goes into cutting it off from the world.

Its qubits still forget everything they know in under a hundred microseconds.

That failure, and the two decades of engineering aimed at it, is the central barrier between every threat in this series and a machine that can execute it, though not the only one: scale, control fidelity, connectivity, decoding speed, and a full logical gate set all have to arrive too. Chapter 8 showed you the algorithm that breaks RSA. This part shows you the hardware that cannot yet run it, why not, and what would have to change. By the end you will know what decoherence actually is (you already have the concept; it has been hiding in this series since Chapter 2 under a different name), how a machine corrects errors in data it is forbidden to read, and why the qubit count in the headline is almost never the number that decides anything. Which makes this the part that changes how you read the news.

The environment never stops measuring

Recall the strangest result in Chapter 4. Two devices in series returned certainty from a fair coin, and installing a detector between them killed the effect, whether or not anyone read the detector. Any record of the intermediate value, anywhere, in anything, destroyed the interference. The reversal ran on silence.

Now stop thinking of that detector as laboratory equipment and start thinking of it as the world. A stray photon bounces off your qubit and flies away carrying news of its state. A vibration in the chip's substrate couples to the qubit's energy. A wandering magnetic field nudges its phase. A cosmic ray strikes the silicon. Each of these leaves a record of the qubit's condition somewhere in the environment, which is Chapter 3's definition of a measurement: an interaction that leaves a record, minds optional. Nobody reads these records. Nobody has to. The interference dies anyway, exactly as the which-path test predicted, and the computation dies with it.

That is decoherence, and it now has a precise description in the vocabulary this series built. The environment entangles with your qubit without being invited. Once qubit and photon are correlated, the qubit alone has no amplitude pair of its own, and the delicate signs that were going to cancel wrong answers are smeared into the universe, unrecoverable, because you cannot copy them back and reading what remains only spends it. Chapter 2 asked you to hold on to the word "recordless." Here is the invoice. Every millikelvin, every layer of shielding, every filtered control line in that refrigerator is there to buy silence, and the silence is never complete.

WHERE THIS BREAKS

Two analogies, one for the villain, one for the cure.

The environment as adversary. Decoherence behaves like a side-channel attack running continuously against your hardware, and the comparison earns its keep: information leaks into the surroundings through physical couplings nobody designed, and the defense is isolation, shielding, and filtering, exactly as with TEMPEST. Where it breaks: there is no adversary, no intent, and nobody who ever reads the leaked data. The universe is not attacking your computer. It is simply in contact with it, and in quantum mechanics contact is enough. That is worse than an attacker, because you cannot deter it, out-resource it, or wait for it to lose interest.

Integrity without plaintext. Error correction below checks parity relationships without reading the protected data, which will feel like verifying an HMAC without ever seeing the message. Where it breaks: an HMAC check is optional, and reading the message would merely be indiscreet. Here, reading the data is fatal to it, so the parity check has to be engineered so that measuring it cannot possibly disturb what it protects. The constraint is physics, and it dictates the entire architecture.

Correcting errors you are forbidden to look at

Classical error correction is old, understood, and everywhere: parity bits, ECC memory, RAID. All of it rests on two moves this series has already taken away from you. Copy the data, and compare the copies. No-cloning forbids the first. Measurement destruction forbids the second, since looking at a qubit to see whether it drifted is exactly the act that ruins it.

For a while this looked fatal, and serious physicists argued that quantum computing was thermodynamically impossible for precisely this reason. Then in 1995, Peter Shor published a scheme for protecting quantum memory from decoherence, one year after publishing the algorithm that made the machine worth building. The same man broke RSA and then rescued the machine that would do it, which is either a fine joke or a fair description of how small this field was in the nineties.

The trick is to stop asking about values and start asking about relationships. Spread one qubit's worth of information across several physical qubits in an entangled state, then design a measurement that reveals only whether the qubits still agree with each other, never what they agree on. Agreement is a parity relationship, and parity can be extracted through helper qubits that touch the data without exposing it. The answer that comes back is called a syndrome, and it says something like "qubits one and two disagree, qubits two and three agree," which pinpoints where an error struck

while telling you nothing whatsoever about the encoded state. Correct the flagged qubit and the logical information rides on, having never been read. My syndrome extraction writeup goes deeper for those who want the full machinery.

The bill for this is large. Real codes must catch both bit flips and phase flips (the second is a uniquely quantum failure, a silent inversion of the minus sign that has driven every result in this series), which multiplies the qubit budget. The surface code, today's workhorse, arranges physical qubits in a lattice and measures parities across neighboring plaquettes continuously, forever, at roughly a million cycles per second, with a classical decoder racing in real time to interpret each syndrome before the next arrives. The machine is not computing most of the time. It is checking itself, relentlessly, and stealing a little computation in the gaps.

THE ARITHMETIC

Finding an error without reading the data.

The simplest code, protecting against bit flips only. Encode one logical qubit into three physical ones, mapping the amplitude on 0 onto the string 000 and the amplitude on 1 onto 111. A qubit in the state (a_0, a_1) becomes a three-qubit state with a_0 on 000 and a_1 on 111, and nothing anywhere else.

This is not cloning. Three copies of (a_0, a_1) would put amplitude on all eight strings, including 010 and 101. This state has amplitude on two strings only, and Chapter 5's failed copier is exactly what built it.

Now flip the middle qubit. The state becomes a_0 on 010 and a_1 on 101. Measure two parities, using helper qubits that compute the answer without touching the data:

$\text{parity}(q_1, q_2) = 1$ (they differ) $\text{parity}(q_2, q_3) = 1$ (they differ)

Both checks fire, which happens for exactly one error: a flip on qubit two. Flip it back. Note precisely what you learned. You learned *where* the error was. You learned nothing about a_0 or a_1, because 010 and 101 have the same parity pattern, so the syndrome is identical whichever amplitude the state was carrying. The data stayed sealed. Check the other cases yourself: a flip on qubit one fires the first check only, a flip on qubit three fires the second only, and no error fires nothing.

That is quantum error correction in miniature. Real codes are bigger and catch phase flips too, and the logic is this logic.

Below threshold, and why the argument changed

Here is the objection that kept this field in the wilderness for a decade, and it is a good one. Error correction requires operations, and operations are themselves error-prone. Add more physical qubits and you add more places for things to go wrong. If the correction machinery introduces errors faster than it removes them, then scaling up makes the machine worse, and no amount of engineering rescues it.

The answer is the threshold theorem, and it is the most consequential result in the field after Shor. If the physical error rate per operation is below a critical threshold, then adding redundancy suppresses logical errors exponentially: each increase in code size makes the logical qubit dramatically more reliable. Above threshold, the machinery drowns. Below it, the machinery wins, and it wins faster than the qubit count grows. Everything depends on which side of that line the hardware sits, which is why "below threshold" is the only hardware milestone I consider load-bearing.

In December 2024, Google's Willow processor crossed it, and the result was published in Nature. Scaling the surface code from distance 5 to distance 7 cut the logical error rate by a factor of 2.14, in the direction the theorem predicts, and the distance-7 logical qubit

outlived the best physical qubit on the chip. That headline result used a high-accuracy decoder running offline, after the fact. A separate distance-5 memory on the same processor demonstrated the other half of the problem, keeping pace below threshold with a decoder running in real time.

This is where I have to be honest on both fronts at once, because the temptation to spin this result in either direction is enormous. To the denialists, who spent twenty years calling fault tolerance a fantasy: the central scaling prediction is now measured on hardware, adding redundancy lowered the error rate rather than raising it, and that moves the core question from "is this possible" to "how far does it go." To the panic industry, which read the same paper as a countdown: this was a memory, holding one idle logical qubit alive, not a logical gate, a non-Clifford operation, or a machine that runs an algorithm for days, and all of those remain undemonstrated. One distance-7 logical qubit costs 101 physical qubits, while Shor against RSA-2048 needs a few thousand logical qubits at error rates orders of magnitude below anything shown. Comparing Willow's memory-error-per-cycle directly against an algorithm's total gate budget would be comparing unlike quantities, so I won't put a single number on the gap. It is large on every axis that matters. Neither fact cancels the other; both belong in any honest assessment.

FOR THE RECORD

Willow's numbers, separated properly.

The Nature paper reports two distinct memories, and coverage that merges them gets the milestone wrong. The distance-7 surface code across 101 physical qubits reached a logical error of $0.143\% \pm 0.003\%$ per cycle, with error-suppression factor $\Lambda = 2.14 \pm 0.02$ for each increase of two in code distance, the signature of below-threshold operation, and a logical lifetime 2.4× its best physical qubit. That result was decoded offline. Separately, a distance-5 code held below-threshold performance with an integrated real-time decoder averaging 63 μs latency against a 1.1 μs cycle time, over runs up to a

> million cycles. High-accuracy scaling came from one code; real-time decoding came from the other.
>
> And the detail most coverage omitted, which Google reported plainly: running repetition codes out to distance 29, performance was limited by rare correlated error events striking roughly once an hour. Cosmic rays and similar bursts hit many qubits at once, which no code designed for independent errors handles gracefully. Fault tolerance at scale has to answer that, and it has not yet. This is what an honest milestone looks like: a real result, with the next problem named in the same paper.

The number in the headline is the wrong number

Which brings us to the distinction that determines what any qubit headline means. A physical qubit is a piece of hardware: a transmon circuit, a trapped ion, an atom in an optical tweezer. A logical qubit is an abstraction woven from many physical ones by error correction, and it is the only unit in which algorithms are written. Shor's resource estimates are quoted in logical qubits (a few thousand for RSA-2048) and paid for in physical ones, but the conversion is not a fixed rate. Gidney's 2025 surface-code analysis put RSA-2048 under a million physical qubits; 2026 preprints using qLDPC codes or reconfigurable neutral atoms claim under 100,000, or even 10,000 at the cost of far longer runtimes. The exchange rate is a property of the architecture, not a constant of nature.

So when a vendor announces 1,000 qubits, the professional question is not "is that a lot." It is: physical or logical, at what error rate, under what code, with what decoder, and sustained for how long. My CRQC Quantum Capability Framework exists precisely because a single number cannot answer that, and tracks the capabilities that actually gate a CRQC: error correction, syndrome extraction, below-threshold scaling, connectivity, logical gate

fidelity, magic states, decoder performance, continuous operation, and manufacturability. Qubit count moves none of them.

MYTH AUTOPSY

"They just hit 1,000 qubits, so we must be close."

The count in the headline is almost always physical, and physical qubits are the cheap part. Today's machines carry hundreds to a few thousand of them at error rates that make deep circuits impossible; RSA-2048 needs a few thousand *logical* qubits, each woven from many physical ones under an error-correcting architecture that determines the ratio, running a deep circuit without a fatal error. The headline number and the number that matters are separated by orders of magnitude and a great deal of engineering.

Now the deeper cut, and the one that will keep you sane through the next two years of press releases: **"logical qubit" is not a standardized unit.** Compare three results published within fifteen months of each other. Google's Willow spent 101 physical qubits on a single distance-7 surface-code logical qubit, decoded offline. QuEra published 96 logical qubits from 448 physical atoms in Nature in January 2026, using high-rate codes with a far better exchange rate. Quantinuum, on 98 trapped ions, reported up to 94 error-*detected* logical qubits and 48 error-*corrected* ones using iceberg codes, and the detection-versus-correction split is a different guarantee entirely: detection tells you an error happened and makes you discard the run, correction fixes it and lets the computation continue.

All three are real. None are commensurable. A logical qubit's worth depends on its code, its distance, its error rate, and whether it can survive a long algorithm rather than a demonstration. Anyone comparing logical-qubit counts across platforms without stating those parameters is selling something, possibly to themselves.

Say instead: ask which capability moved. Qubit counts are inputs, and the CRQC is gated by error rates, decoder speed, logical gate fidelity, and sustained operation.

Five ways to build a qubit

The modalities differ in ways that matter for reading roadmaps, and each trades the same currencies: speed, fidelity, connectivity, and manufacturability.

Superconducting circuits (Google, IBM, Rigetti) are fast, with gates in tens of nanoseconds, and they are lithographed like chips, which is a real manufacturing advantage. They are also the noisiest, needing dilution refrigerators and dense microwave wiring that becomes its own scaling problem. Trapped ions (Quantinuum, IonQ) are the opposite: exquisite fidelity and all-to-all connectivity, with gate speeds a thousand times slower, which bites hard when Shor needs ten billion sequential operations. Neutral atoms (QuEra, Pasqal, Atom Computing, Infleqtion) have surged, offering thousands of identical atoms held in optical tweezers and physically shuffled to create connectivity, which is exactly what produced QuEra's high-rate-code result. Photonic approaches (PsiQuantum, Xanadu) run at room temperature and lose photons instead of scrambling them. Silicon spin qubits (Intel, Diraq) are the smallest, betting that the existing semiconductor industry becomes decisive at scale.

Nobody knows which wins. It is entirely possible that more than one does, and equally possible that the winner is a modality nobody is funding today.

ATTACKER'S-EYE VIEW

How to read a quantum press release.

You will be forwarded these announcements by executives asking whether to panic. Six questions, in order, and the ma-

chine's own physics generates all of them.

Physical or logical? Almost always physical; if logical, under which code and at what distance. What is the error rate per operation, and is it below threshold? Does the machine hold below-threshold performance with a real-time decoder, or was decoding done offline afterward, which is a demonstration rather than a computer? For how long does it run: a burst of a few hundred cycles, or hours, since Shor needs days of coherent operation. Is the result error correction or error detection? And the question that dissolves ninety percent of the hype: does this move any capability that a CRQC actually requires, or does it move a number that is easy to grow?

There is also a defensive lesson buried in the hardware. Everything in this part says a quantum computer is a monstrously fragile instrument that any leaked record can ruin, and that fragility will not save you, because the adversary does not need a reliable machine for long. They need one machine, once, for a week, with your ten-year-old harvested traffic in the queue. Fragility delays the threat. It does not defuse it, and it certainly does not shorten your migration.

What to remember

Decoherence is the environment measuring your qubits without being invited, entangling with them and carrying off the signs that were going to cancel wrong answers, which is why the machine lives in a refrigerator colder than deep space and still fails in microseconds. Error correction works despite no-cloning and destructive measurement by checking parity relationships rather than values, so it learns where an error struck while learning nothing about the data it protects. The threshold theorem says that below a critical physical error rate, redundancy suppresses logical errors exponentially, and Google's Willow demonstrated exactly that in 2024, which moved fault tolerance from physics question to engineering problem while leaving a four-order-of-magnitude gap to a CRQC. Physical qubits are the cheap part and the headline part; logical

qubits are the unit that matters, and "logical qubit" is not yet a standardized unit across platforms. Ask which capability moved.

Next, in Chapter 11, "Reading the Field Like a Professional": the series closes by turning all of this into working equipment. How to read a vendor roadmap and a research paper, which milestones would actually change your migration timeline, what the resource estimates do and do not tell you, and the standing answers to the questions you will be asked for the rest of your career.

Reading the Field Like a Professional

Four numbers, fourteen years apart. In 2012, Austin Fowler and colleagues estimated that breaking RSA-2048 would take roughly a billion physical qubits. In 2019, Gidney and Ekerå brought it down to 20 million, running for eight hours. In May 2025, Gidney cut his own figure to under a million. By February 2026, the Pinnacle architecture claimed under 100,000, and a neutral-atom study argued the job could run on as few as 10,000 atoms.

A drop of five orders of magnitude in fourteen years, and none of it came from better hardware. Gidney's 2025 paper assumes the same gate error rates and cycle times as his 2019 paper; the 2026 papers change the error-correcting code and the machine's architecture, not the physical qubits underneath. The devices did not improve. The mathematics and the engineering of how to arrange them did.

That fact organizes this final part. Ten articles built you the physics; this one turns it into equipment. You will get the two clocks that govern your exposure and why you can only see one of them, the specific milestones that would move a timeline and the announcements that never will, the standing answers to the questions you will be asked for the rest of your career, and the arithmetic that sets your actual deadline. Which, as it happens, was never Q-Day.

The clock you cannot see

Every roadmap you will be shown tracks hardware. Qubit counts, error rates, delivery dates, a tidy line climbing to the right. Hardware is visible because it is expensive, physical, and announced by companies who want credit for it. You can track it, and Chapter 10 taught you how.

The second clock tracks the resource model, and it runs in silence. It ticks when a researcher finds a cheaper circuit for modular arithmetic, when a new code stores idle logical qubits more efficiently, when someone works out how to make magic states without a distillation factory the size of the rest of the machine. Gidney's twenty-fold drop came from exactly that: approximate residue arithmetic borrowed from Chevignard and colleagues, yoked surface codes, magic state cultivation. The 2026 papers moved the number again by swapping the whole error-correcting architecture, qLDPC codes in one case, reconfigurable atoms in another. No new hardware, no announcement schedule, no roadmap, no warning. A preprint appears (Gidney's appeared on a Wednesday in May, with no press cycle at all), and the number that governs your exposure changes by an order of magnitude overnight.

Your organization's exposure is a function of both clocks, and most executive information diets cover one of them. When a CISO tells me they are watching the hardware roadmaps closely, my honest reply is that they are watching the clock that has moved *least*. Real, demonstrated physical qubit counts have grown by roughly two orders of magnitude since 2012. The modeled requirement to break RSA-2048 has fallen by five. Those are different kinds of number, a device that exists against an estimate on paper, and you should never plot them on one axis. But the asymmetry in how fast each moves is the whole point: the clock nobody puts on a slide is the one setting the pace.

FOR THE RECORD

The trajectory, and the honest counterweight.

Published estimates of the physical qubits needed to factor RSA-2048:

Study	Estimate	Runtime	Architecture / status
Fowler et al., 2012	~1,000,000,000	~1 day	Surface code; peer-reviewed

Study	Estimate	Runtime	Architecture / status
Gidney-Ekerå, 2019	20,000,000	8 hours	Surface code; peer-reviewed (2021)
Gidney, 2025	<1,000,000	<1 week	Surface code, yoked storage; preprint
Pinnacle (Webster et al.), 2026	<100,000	trade-off	qLDPC; preprint
Cain et al., 2026	~10,000	much longer	Neutral-atom; preprint

Do not read this as one smooth trend line. Only the 2019-to-2025 step is like-for-like, same assumptions, so its 20× reduction is a clean measure of algorithmic progress. The rest change codes, connectivity, and runtimes, and the last three rows are unreviewed preprints on machines nobody has built. Google's public tracker follows the 2012-2025 surface-code sequence and predates the 2026 architecture studies.

Now the counterweight, because a curve extrapolated without friction is a fantasy. These estimates model idealized machines. They assume independent errors, and Chapter 10 showed you Google's own finding that correlated bursts strike roughly hourly and break codes designed for independence. They assume decoders keep pace, control electronics scale, cryogenic plumbing scales, and fabrication yields hold across a million qubits, none of which is demonstrated. Real engineering has a way of adding zeroes back. The trend is real and the friction is real, and anyone selling you only one of them is selling.

What would actually change my mind

Given both clocks, the professional question stops being "when is Q-Day" and becomes "what would I have to see." Here is my list, framed against the capabilities in my CRQC Quantum Capability Framework, and I offer it as a falsifiable commitment rather than an opinion.

A logical qubit that runs for hours rather than seconds, with real-time decoding holding the whole way, would move me. Shor against RSA-2048 needs days of unbroken operation, and nothing demonstrated so far runs anything like that long. A logical two-qubit gate measured at an error rate far below anything shown today would move me, and I want to see a gate error, not a memory error borrowed from an idle encoded qubit and quoted as though the two were the same number. A cheap solution to magic state production would move me, since non-Clifford gates currently dominate the machine's footprint and a large improvement there compresses every estimate at once. An architecture that handles correlated error bursts would move me, because that problem is unsolved and unpriced. And a genuine end-to-end Shor run on a number large enough that the circuit could not have been pre-simplified using the answer would move me most of all, because it would mean the integration problem is solved and only scale remains.

One of my tripwires has already fired, and I am obliged to say so rather than quietly revise the list. Two years ago I would have told you that an order-of-magnitude drop below Gidney's million physical qubits would be a significant event. Pinnacle and the neutral-atom studies delivered exactly that, on paper, within a single quarter. It did not move Q-Day, because none of those machines exists and none of those papers has been through review. It did move my estimate of how fast the number can move without any hardware changing.

Now the announcements that will be pushed at you as evidence. A new qubit-count record moves nothing on its own; those are physical qubits, the easiest number in the field to grow, and Chapter 10 explained why the headline number is the wrong number.

A "quantum supremacy" or random-circuit-sampling result carries no cryptanalytic content whatsoever, though it does test control and calibration at scale, so give it that much and no more. A factoring "record" achieved with annealing, variational methods, or Schnorr-style hybrid tricks moves nothing; those claims have appeared repeatedly for a decade and none has scaled, which is why implementations of Shor's algorithm have still factored nothing bigger than 15 and 21 and why a 48-bit factoring demonstration tells you nothing about RSA-2048. And a vendor roadmap slipping two years should update your confidence in that vendor, by rather less than a demonstrated capability would. Roadmaps are marketing documents with engineering attached.

WHERE THIS BREAKS

The two clocks are not independent.

The two-clock model is the most useful frame I know for this problem, and it has one limit worth naming. The clocks are coupled. Gidney's yoked surface codes assume a particular hardware geometry; magic state cultivation assumes particular gate fidelities; algorithmic wins are frequently made possible *by* hardware properties, and hardware roadmaps are steered *by* what the algorithms need. A better model is two runners tied at the ankle, sometimes helping and sometimes tripping each other. The operational lesson holds regardless: you are watching one runner and betting on both.

The standing answers

You will be asked these for the rest of your career, often by people who have just read something breathless. Each answer is one sentence, with the full argument a click away.

"Isn't a quantum computer just a much faster computer?" No, it is a different machine that wins on a narrow class of structured problems and loses everywhere else. (Chapter 1)

"Isn't a qubit 0 and 1 at the same time?" No, it holds a weighted combination of 0 and 1 whose weights can cancel, which is what "0 and 1 at once" cannot describe. (Chapter 3)

"Doesn't measurement require a conscious observer?" No, a measurement is any interaction that leaves a record anywhere, and minds have never entered into it. (Chapter 3)

"Isn't the qubit really 0 or 1, and we just don't know which?" No, and this one is settled experimentally: treating a superposition as a hidden value predicts a coin where the bench delivers certainty. (Chapter 4)

"Doesn't it try every answer at once?" It computes with the amplitudes of every possibility, then reads out one string, so parallelism alone yields a random guess; interference aimed by structure is where all the advantage lives. (Chapter 5)

"Can't entangled qubits communicate instantly?" No, local statistics never move no matter what the far end does, so there is no channel and no bit. (Chapter 6)

"Will quantum computers break all encryption?" No, they break public-key algorithms built on factoring and discrete logs, while symmetric ciphers and hashes hold. (Chapter 9)

"Do we need AES-512?" No, AES-256 is the answer, and the reasons why are more interesting than the answer. (Chapter 9)

"They just announced a thousand qubits, are we close?" Those are physical qubits, the easiest number to grow; RSA-2048 needs thousands of *logical* ones, and the number of physical qubits each takes depends entirely on the error-correcting architecture. (Chapter 10)

"Should we buy quantum encryption?" Ask which of four unrelated products they mean, and the answer for general systems is post-quantum cryptography, which is software. (Chapter 1)

Your deadline was never Q-Day

Everything above concerns a machine whose arrival nobody can date. Notice how little that matters.

Go back to Mosca's inequality from Chapter 1: if the years your data must stay secret, plus the years your migration will take, exceed the years until a cryptographically relevant machine exists, you are already late. Two of those three numbers belong to you. Only the third belongs to the physicists, and it is the only one anyone argues about.

Used properly, Mosca's inequality is not a doomsday clock. It is a sorting function. Run it per data class and it partitions your estate immediately: session tokens with a ninety-day life and a three-year migration are comfortable under any plausible timeline, while patient records with a fifty-year confidentiality requirement are already past due under every plausible timeline, and no argument about Q-Day changes either verdict. The arithmetic box below works both cases. What falls out is a priority order you can defend to a board without ever predicting anything.

And your true deadline is set elsewhere, which is the argument I have made at length and stand behind. Regulators have published dates. NIST IR 8547 remains an Initial Public Draft, but its proposed deprecation of quantum-vulnerable RSA and ECC after 2030 and disallowance after 2035 already has teeth: Executive Order 14412 and OMB M-26-15 treat those dates as federal compliance deadlines, and procurement, audit, and insurance pressure follows the dates regardless of the document's formal status. CNSA 2.0 binds national security systems and their supply chain on a schedule already running. Insurers are asking. Clients are asking, and their procurement questionnaires do not have a field for your Q-Day estimate. None of these actors will wait for a physicist to ring a bell, and every one of them can hurt you before the first CRQC boots.

THE ARITHMETIC

Solve it for your own estate.

Mosca: you are late when $x+y > z$, where x is your data's required confidentiality lifetime, y is your migration time, and z is the years until a CRQC.

Case one, session tokens. $x = 0.25$ years. $y = 3$ years for a contained TLS-layer migration. So $x + y = 3.25$. Is a CRQC more than three years away? Almost certainly. Verdict: comfortable, and it can wait its turn.

Case two, patient records. $x = 50$ years, because a genome does not expire. $y = 7$ years, which is realistic for a large hospital network with embedded devices and vendor dependencies. So $x + y = 57$. Is a CRQC more than fifty-seven years away? Nobody serious will make that claim. Verdict: you are already late, and were late when the data was first transmitted.

Neither verdict required a Q-Day prediction. The first is safe under every estimate and the second is doomed under every estimate, so the uncertainty that dominates public argument is irrelevant to both. That is the whole trick: the estimate for z only matters for data in the middle band, and for everything else your own two numbers decide it. Run the inequality across your data classes and the sorting does itself. My CRQC Readiness Benchmark lets you vary z under your own assumptions if you want to see the middle band move.

So the action list is short, and none of it depends on knowing anything about physics. Build a cryptographic inventory, because you cannot migrate what you cannot see, and remember from Chapter 9 that the exposure lives in key exchange and signatures rather than in the cipher suite. Sort the inventory by data lifetime, using the arithmetic above. Demand crypto-agility in every procurement from today onward, so the next transition is a configuration change. Start moving key establishment to the standardized algorithms, hybrid where prudent. And treat signatures as the slower, harder problem they are, since trust now, forge later attacks the roots you cannot swap quickly. My PQC Migration Framework is

free and covers the sequencing in detail.

ATTACKER'S-EYE VIEW

The adversary's project plan is already executing.

Assemble everything this series taught you and look at the problem from the other desk. A well-resourced adversary in 2026 has a program plan with three streams running in parallel, and none of them requires a quantum computer to exist.

Stream one, collection: capture and warehouse quantum-vulnerable handshakes and ciphertext whose value outlives the protection. Note the mechanism precisely, because this is where most threat models go wrong. Forward secrecy does not save you: a recorded TLS session using ordinary ephemeral ECDH is still readable later, because the attacker Shors the ephemeral public values sitting in the transcript. Long-term keys are the richest target wherever static key transport or key wrapping depends on them. Passive collection is cheap and effectively undetectable at the endpoint, which is why you should treat it as a present threat rather than wait for proof of a particular archive. Stream two, positioning: identify long-lived signing roots and code-signing keys whose trust must outlive the machine, because those become forgery opportunities the moment the mathematics gives way. Stream three, patience: fund the hardware, or simply wait for someone else to.

The streams do not all mature on one day, and the runtime is architecture-dependent rather than universal. What holds is the asymmetry. The adversary needs the machine to work once, unreliably, in a lab, with no service-level agreement and no uptime commitment. You need your migration finished before that run happens against data that still matters. The fragility of quantum computers, which is real and which Chapter 10 documented in detail, protects them and not you.

> Date uncertainty changes which assets you move first. It does not remove the need to start a migration whose own duration is measured in years.

What to remember

Two clocks govern your exposure: hardware, which you can watch, and the resource model, which you cannot. The second has moved faster, cutting the RSA-2048 estimate by five orders of magnitude in fourteen years with no hardware improvement at all, which means the number that determines your risk can change on any given Wednesday without warning. Judge announcements by which capability they move, since qubit records, sampling stunts, and annealing-based factoring claims move none. Mosca's inequality is a sorting function rather than a prophecy, and run across your data classes it produces a defensible priority order without predicting anything. Your real deadline comes from regulators, insurers, clients, and the shelf life of your own secrets, all of which are already set. And the adversary needs the machine to work once; you need to be finished before it does.

The three items, revisited

Chapter 1 opened with three things crossing your feed in a single week: a processor announcement with a record qubit count, an article explaining that quantum computers try every combination at once, and a colleague forwarding both with the subject line "should we be worried about AES?"

Of the two factual claims in that pile, exactly one was accurate, and I said so before you had any way to check.

You can check now. The qubit count is real and nearly meaningless, because those are physical qubits and the machine's capability is decided by logical error rates, decoder speed, and sustained operation. The article is wrong, and you can say precisely why: the

register does hold every possibility, and the readout returns one Born-random string, so parallelism alone is a very expensive coin toss and only interference aimed by structure converts it into an answer. And your colleague's email was the right question aimed at the wrong target. No, we should not be worried about AES. We should be worried about the elliptic-curve handshake that delivered the AES key, which a fault-tolerant machine breaks while never touching the cipher at all.

Eleven parts ago that reply would have needed a physicist. It needs you.

Glossary

Amplitude. The number a quantum computer computes with: every possible outcome of a register carries one, and it behaves like a probability except that it can be negative, so two amplitudes for the same outcome can cancel. Squaring an amplitude gives the probability of observing that outcome. In full generality amplitudes are complex numbers; the real-number treatment in this book is an exact slice of the theory, not an approximation.

Bell test. An experiment on entangled pairs in which each side chooses among several measurement settings. The observed correlations exceed what any pre-written-answer explanation can produce, which is how physics ruled out the gloves-in-boxes model. Loophole-free versions have run since Delft's 2015 experiment.

Below-threshold operation. The regime in which enlarging an error-correcting code makes the logical qubit better instead of worse. Google's Willow processor demonstrated it on a surface code in 2024; above threshold, error correction is a net liability.

Born rule. Square the amplitude, get the probability. The only bridge between the amplitudes a quantum computer manipulates and the classical outcomes you can read out.

CNSA 2.0. The NSA's Commercial National Security Algorithm Suite 2.0, the requirement set for US national security systems: ML-KEM-1024 and ML-DSA-87 for public-key work, AES-256 and SHA-384 for symmetric primitives, on a migration schedule that binds vendors and integrators as well.

Code distance. The size parameter of a quantum error-correcting code. Larger distance means more physical qubits per logical qubit and, below threshold, exponentially better error suppression.

Coherence time. How long a qubit holds its state before the environment effectively measures it. Superconducting qubits sit under a hundred microseconds; everything a quantum computer does must fit inside, or be error-corrected past, that window.

Continued-fractions algorithm. The classical arithmetic that converts Shor's measurement outcome, a sample near a multiple of the pattern's frequency, into a candidate period.

CRQC. Cryptographically relevant quantum computer: a fault-tolerant machine large and stable enough to run Shor's algorithm against deployed key sizes. No such machine exists; every migration deadline in this book was set without waiting for one.

Crypto-agility. The engineering property of being able to swap cryptographic algorithms without redesigning the systems that use them. The cheapest insurance available against both Q-Day and the ordinary breakage of classical algorithms.

Decoder. The classical computer that turns a stream of syndrome measurements into corrections while the quantum computation runs. Its latency is a first-class constraint: a decoder that falls behind is indistinguishable from no decoder.

Decoherence. The environment interacting with a qubit in a way that leaves a record, which is to say measuring it, uninvited. The central engineering barrier between the algorithms in this book and a machine that can run them.

Entanglement. A joint state of two or more qubits that cannot be factored into separate descriptions of its parts. The pair has an amplitude description; neither member has one of its own. It produces correlations no pre-written ledger can match, and it carries no signal.

Error-detected vs. error-corrected. Two different guarantees behind the phrase "logical qubit." Detection tells you an error occurred so you can discard the run; correction fixes the error and lets the computation continue. Vendor logical-qubit counts routinely mix the two.

Fault tolerance. Architecture in which errors are corrected faster than the correction machinery introduces new ones, letting long computations run on imperfect hardware. The qualifier on which every sentence about Shor's algorithm depends.

Forward secrecy. The property that compromising a long-term key does not expose past session keys. It survives classical key theft; it

does not survive Shor, because the attacker breaks the ephemeral key-exchange values recorded in the transcript itself.

Grover's algorithm. The optimal quantum algorithm for unstructured search: roughly the square root of the classical number of guesses, delivered as strictly sequential iterations. Parallelizing across many machines buys only a square-root speedup, which is why doubling symmetric key length closes the question.

Harvest now, decrypt later (HNDL). Recording quantum-vulnerable ciphertext today to decrypt when a CRQC exists. Cheap, passive, effectively undetectable, and the reason long-lived secrets are already on the clock.

Hidden-subgroup problem. The algebraic family whose members include factoring and the discrete logarithms. Shor-style period-finding breaks the family, which is why one algorithm ended several cryptosystems at once.

Holevo bound. Alexander Holevo's 1973 theorem: without preshared entanglement, n qubits yield at most n classical bits, however cleverly encoded and however exotically measured. Preshared entanglement buys a factor of two (superdense coding), never an exponential readout.

Hybrid key establishment. Running a classical and a post-quantum key exchange together and combining the results, so security holds unless both fail. The engineering-prudent default during migration; Signal's PQXDH is a deployed example.

Interference. Amplitudes for the same outcome adding when signs agree and cancelling when they differ. The one genuinely new computational phenomenon quantum mechanics offers, and the engine of every algorithm in this book.

Logical qubit. The abstraction algorithms are written in: one protected qubit woven from many physical ones by error correction. Not a standardized unit; a count means nothing without the code, the distance, the error rate, and whether errors are detected or corrected.

Magic state. A resource state consumed to perform the non-Clifford gates that error-correcting codes cannot apply directly.

Producing them at scale dominates the projected footprint of fault-tolerant machines.

MAXDEPTH. NIST's cap on the plausible sequential depth of a quantum attack circuit, written into its post-quantum evaluation criteria. The reason honest costings of Grover attacks diverge so far from the halves-your-key-length folklore.

Measurement. Any interaction that leaves a record of a qubit's state, minds optional, observers irrelevant. It returns one outcome with Born-rule probability and destroys the amplitude description that existed before.

ML-KEM and ML-DSA. NIST's standardized post-quantum key-encapsulation and signature algorithms (formerly CRYSTALS-Kyber and CRYSTALS-Dilithium), the workhorses of the migration this book keeps pointing at.

Modular exponentiation. Computing $a^x \bmod N$. The repeating sequence it produces is where RSA hides its period, and evaluating it in superposition is the expensive heart of Shor's circuit.

Mosca's inequality. If migration time plus the shelf life of your data exceeds the time to a CRQC, you are already late. Best read not as a prophecy but as a sorting function that ranks data classes by urgency.

Neutral-atom qubit. A qubit encoded in an individual atom held in optical tweezers, reconfigurable in ways solid-state platforms are not, and the basis of the 2026 estimate that Shor might run on as few as ten thousand atoms.

NIST IR 8547. NIST's draft transition guidance: quantum-vulnerable RSA and ECC deprecated after 2030 and disallowed after 2035. Still an Initial Public Draft as of this writing, but executive and OMB actions already treat the dates as federal compliance deadlines.

No-cloning theorem. An unknown quantum state cannot be copied. It forbids duplication, not possession: a stolen quantum register can be carried off and measured once, but never imaged for replay.

Oracle. A reversible circuit that packages a problem so a quantum algorithm can act on it, typically by flipping the sign of the amplitude on inputs that satisfy a condition. The oracle is where the problem's structure, or its absence, enters the machine.

Period finding. Determining the cycle length of a repeating sequence, which is the only step of RSA-breaking that needs a quantum computer. Shor reduced factoring to it; interference does the rest.

Physical qubit. A piece of hardware, whether a transmon circuit, a trapped ion, or an atom in a tweezer, with error rates far too high to run deep circuits unprotected. The number in every headline, and the wrong number to watch.

Post-quantum cryptography (PQC). Public-key cryptography built on problems with no known quantum shortcut, standardized precisely so it can run on today's classical hardware. The migration to it is the practical payload of this entire book.

Q-Day. Shorthand for the day a CRQC first breaks deployed public-key cryptography. This book's position: the date is unknowable, and your deadlines were set by regulators, insurers, and clients long before it.

qLDPC codes. Quantum low-density parity-check codes, which protect many logical qubits at a far better physical-to-logical exchange rate than the surface code, at the cost of harder connectivity and decoding. The basis of the Pinnacle estimate that cut projected RSA-2048 requirements below 100,000 physical qubits.

Quantum error correction (QEC). Encoding one logical qubit across many physical qubits so errors can be found and fixed without reading, and therefore destroying, the protected data.

Quantum Fourier transform (QFT). The interference pattern-reader at the end of Shor's circuit: it concentrates amplitude on the frequency of a repeating pattern, turning a global property no single measurement could see into a readable output.

Qubit. A physical system described by a pair of amplitudes. Reading it yields one classical bit and destroys the pair; the power of n

qubits lives in the 2^n amplitudes describing them jointly, behind a readout door n bits wide.

Record, recordless. This book's operational vocabulary for measurement. A process is recordless when no trace of the intermediate state exists anywhere in the universe; only recordless alternatives interfere. One stray record, in anything, and the quantum behavior is gone.

Shor's algorithm. Peter Shor's 1994 algorithm that factors integers and takes discrete logarithms in polynomial time on a fault-tolerant quantum computer, by converting each problem into period finding. The reason RSA, Diffie-Hellman, and elliptic-curve cryptography share a retirement date.

Superconducting qubit, transmon. A qubit built from a superconducting circuit operated at around ten millikelvin, the platform behind Google's Willow. Fast gates, short coherence, and an entire refrigerator per chip.

Superposition. A single definite quantum state that carries amplitudes for more than one measurement outcome. Not "0 and 1 at the same time": the qubit is in exactly one state; it is the outcomes that are plural until a record exists.

Surface code. The best-studied quantum error-correcting code: logical qubits encoded in a two-dimensional grid of physical qubits with only nearest-neighbor interactions. The baseline architecture of most published RSA-2048 resource estimates.

Syndrome. The output of the parity checks an error-correcting code runs continuously: a pattern that reveals where an error occurred without revealing, or disturbing, the encoded data itself.

Trapped-ion qubit. A qubit encoded in an individual charged atom suspended in electromagnetic fields. Slower gates than superconducting platforms, far longer coherence, and all-to-all connectivity within a trap.

Trust Now, Forge Later (TNFL). The signature-side twin of HNDL: signatures trusted today become forgeable once the underlying mathematics falls, because an attacker can mint new

signatures that verify under the old public key. Why long-lived signing roots belong at the front of the migration queue.

Unstructured search. Finding a needle in a haystack that offers no shortcuts: no ordering, no algebra, nothing to exploit but trying candidates. Grover's square-root speedup is provably the best a quantum computer can do here, which is most of why symmetric cryptography survives.

About the author

Marin Ivezic is the author of PostQuantum.com, where his analysis of quantum computing and post-quantum security draws over a million readers a month, and the founder and CEO of Applied Quantum, a quantum systems integration and consulting firm. He has written three books on quantum technology: *Quantum Ready*, the practitioner's guide to post-quantum cryptography migration; *Quantum Sovereignty*, on national strategy and the geopolitics of quantum technology; and *Quantum Systems Integration*, on the engineering of practical quantum computers. He maintains the free, open-source PQC Migration Framework at PQCFramework.com. His quantum work spans decades and two earlier ventures, Boston Photonics and PQ Defense, founded before the field went mainstream. A former Fortune Global 500 CISO and CTO who led practices at IBM, Accenture, PwC, and KPMG, he narrowed his focus to the question this book answers: what quantum computing actually means for the people defending real systems.

For the online edition of this series, the CRQC Readiness Benchmark, and ongoing coverage of the quantum threat:

https://postquantum.com