Shor's Algorithm and Quantum Cryptography

Last time

A unitary U induces phases on its eigenvectors:

$$U|\psi\rangle = e^{2\pi i\theta}|\psi\rangle .$$

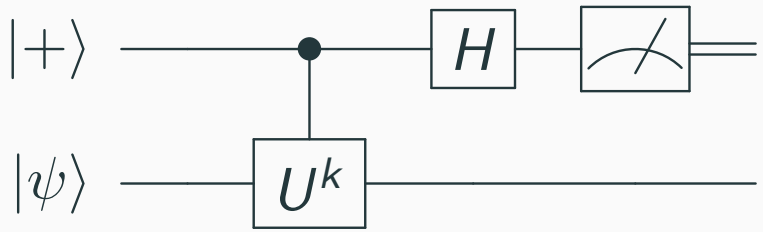
The goal of **Phase Estimation** is to estimate θ.

Last time

A unitary U induces phases on its eigenvectors:

$$U|\psi\rangle = e^{2\pi i\theta}|\psi\rangle\,.$$

The goal of **Phase Estimation** is to estimate θ.



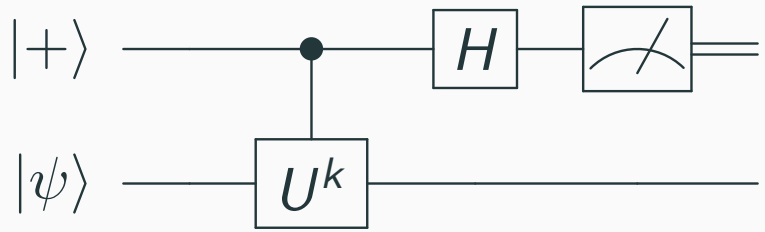
We use different powers of U to learn more bits of the phase.

Last time

A unitary U induces phases on its eigenvectors:

$$U|\psi\rangle = e^{2\pi i\theta}|\psi\rangle .$$

The goal of **Phase Estimation** is to estimate θ.



We use different powers of U to learn more bits of the phase.

For factoring, we constructed a unitary whose phases contain information about an **order**.

Order finding

Input: Integers N, x such that $1 \leq x < N$ and $\gcd(x, N) = 1$.

Goal: Find the smallest positive integer r such that

$$x^r = 1 \pmod{N}.$$

This is called the **order** of x modulo N.

Order finding

Input: Integers N, x such that $1 \leq x < N$ and $\gcd(x, N) = 1$.

Goal: Find the smallest positive integer r such that

$$x^r = 1 \pmod{N}.$$

This is called the **order** of x modulo N.

Example: $x = 7$, $N = 15$.

$$7^1 = 7, \qquad 7^2 = 4, \qquad 7^3 = 13, \qquad 7^4 = 1 \pmod{15}.$$

Therefore the order is $r = 4$.

Modular multiplication unitary

Fix x, N with $\gcd(x, N) = 1$. Define

$$U\,|y\rangle = |xy \bmod N\rangle\,.$$

Example: $x = 7$, $N = 15$.

$$|1\rangle \xrightarrow{U} |7\rangle \xrightarrow{U} |4\rangle \xrightarrow{U} |13\rangle \xrightarrow{U} |1\rangle\,.$$

Modular multiplication unitary

Fix x, N with $\gcd(x, N) = 1$. Define

$$U|y\rangle = |xy \bmod N\rangle .$$

Example: $x = 7$, $N = 15$.

$$|1\rangle \xrightarrow{U} |7\rangle \xrightarrow{U} |4\rangle \xrightarrow{U} |13\rangle \xrightarrow{U} |1\rangle .$$

Last time we found eigenvectors $|v_0\rangle, \ldots, |v_{r-1}\rangle$ satisfying

$$U|v_s\rangle = e^{2\pi i s/r}|v_s\rangle .$$

Phase estimation on $|v_s\rangle$ gives an estimate of s/r.

Getting the eigenvectors

How do we get our hands on the eigenvector $|v_s\rangle$?

Getting the eigenvectors

How do we get our hands on the eigenvector $|v_s\rangle$?

While it is hard to construct $|v_s\rangle$ individually, the superposition

$$\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |v_s\rangle = |1\rangle$$

is easy to prepare! **(Exercise.)**

Getting the eigenvectors

How do we get our hands on the eigenvector $|v_s\rangle$?

While it is hard to construct $|v_s\rangle$ individually, the superposition

$$\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |v_s\rangle = |1\rangle$$

is easy to prepare! **(Exercise.)**

Running phase estimation on $|1\rangle$ gives an estimate of

$$\frac{s}{r}, \qquad \text{for a uniformly random } 0 \leq s < r.$$

We do not need to know r to prepare the input.

Summary of Order Finding algorithm

1. Run Phase Estimation with U and its powers, starting with $|1\rangle$.

Summary of Order Finding algorithm

1. Run Phase Estimation with U and its powers, starting with $|1\rangle$.
2. Get an estimate $\widetilde{\theta}$ of s/r for a random s.

Summary of Order Finding algorithm

1. Run Phase Estimation with U and its powers, starting with $|1\rangle$.
2. Get an estimate $\widetilde{\theta}$ of s/r for a random s.
3. Use the **Continued Fractions Algorithm** to find a nearby fraction a/b in lowest terms.

Summary of Order Finding algorithm

1. Run Phase Estimation with U and its powers, starting with $|1\rangle$.
2. Get an estimate $\widetilde{\theta}$ of s/r for a random s.
3. Use the **Continued Fractions Algorithm** to find a nearby fraction a/b in lowest terms.
4. Check whether $x^b = 1 \pmod{N}$. If not, repeat.

Continued fractions is a fast **classical** algorithm. We will not cover it here.

With a sufficiently accurate estimate, a/b is the reduced form of s/r. If $\gcd(s, r) = 1$, its denominator is r.

Summary of Factoring via Quantum Computers

1. If we can factor, we can break RSA.

Summary of Factoring via Quantum Computers

1. If we can factor, we can break RSA.
2. **Fact**: If we can solve Order Finding efficiently, we can factor integers efficiently.

Summary of Factoring via Quantum Computers

1. If we can factor, we can break RSA.
2. **Fact**: If we can solve Order Finding efficiently, we can factor integers efficiently.
3. Phase Estimation with the modular multiplication unitary gives estimates of s/r.

Summary of Factoring via Quantum Computers

1. If we can factor, we can break RSA.
2. **Fact**: If we can solve Order Finding efficiently, we can factor integers efficiently.
3. Phase Estimation with the modular multiplication unitary gives estimates of s/r.
4. The remaining steps are efficient classical computations.

The algorithm takes $\mathrm{poly}(\log N)$ time.

We implement U^k by computing x^k mod N using repeated squaring, **not** by applying U k times.

Cryptography after Shor's algorithm

Why haven't we broken RSA already?

The circuits we have drawn assume that gates work perfectly.

Real qubits are noisy. A long computation needs **error correction**.

Why haven't we broken RSA already?

The circuits we have drawn assume that gates work perfectly.

Real qubits are noisy. A long computation needs **error correction**.

- A **logical qubit** is a protected qubit used by the algorithm.
- **Physical qubits** are the actual hardware used to store and protect it.

Why haven't we broken RSA already?

The circuits we have drawn assume that gates work perfectly.

Real qubits are noisy. A long computation needs **error correction**.

- A **logical qubit** is a protected qubit used by the algorithm.
- **Physical qubits** are the actual hardware used to store and protect it.

Having enough qubits to write down N is not enough. We must also perform a very long computation reliably.

Gidney–Ekerå (2021); Gidney (2025).

How far away is Shor's factoring algorithm?

Two resource estimates for factoring a **2048-bit RSA integer**:

	Physical qubits	Time
Gidney–Ekerå (2019/2021)	20 million	8 hours
Gidney (2025)	< 1 million	< 1 week

Same basic hardware assumptions: a square grid, 0.1% gate errors, 1 μs error-correction cycles, 10 μs control reaction time.

How far away is Shor's factoring algorithm?

Two resource estimates for factoring a **2048-bit RSA integer**:

	Physical qubits	Time
Gidney–Ekerå (2019/2021)	20 million	8 hours
Gidney (2025)	< 1 million	< 1 week

Same basic hardware assumptions: a square grid, 0.1% gate errors, 1 μs error-correction cycles, 10 μs control reaction time.

These are **resource estimates**, not demonstrations or predictions of a particular year.

Better algorithms can reduce the resources, too.

Gidney–Ekerå, Quantum 5, 433 (2021; preprint 2019); Gidney, arXiv:2505.15917 (2025).

Why prepare before the machine exists?

Harvest now, decrypt later:

1. Record encrypted communication today.
2. Save it.
3. Try to decrypt it when a sufficiently powerful quantum computer becomes available.

Why prepare before the machine exists?

Harvest now, decrypt later:

1. Record encrypted communication today.
2. Save it.
3. Try to decrypt it when a sufficiently powerful quantum computer becomes available.

Some information must stay secret for many years.

Replacing cryptography in existing software and hardware also takes time.

We cannot wait until the attack works before thinking about a replacement.

NIST, What Is Post-Quantum Cryptography?

What will replace RSA?

Post-quantum cryptography: classical cryptosystems designed to be secure even against quantum computers.

What will replace RSA?

Post-quantum cryptography: classical cryptosystems designed to be secure even against quantum computers.

- Alice and Bob still use ordinary classical computers.
- The adversary may have a quantum computer.
- Security relies on problems believed to be hard for **both** kinds of computers.

What will replace RSA?

Post-quantum cryptography: classical cryptosystems designed to be secure even against quantum computers.

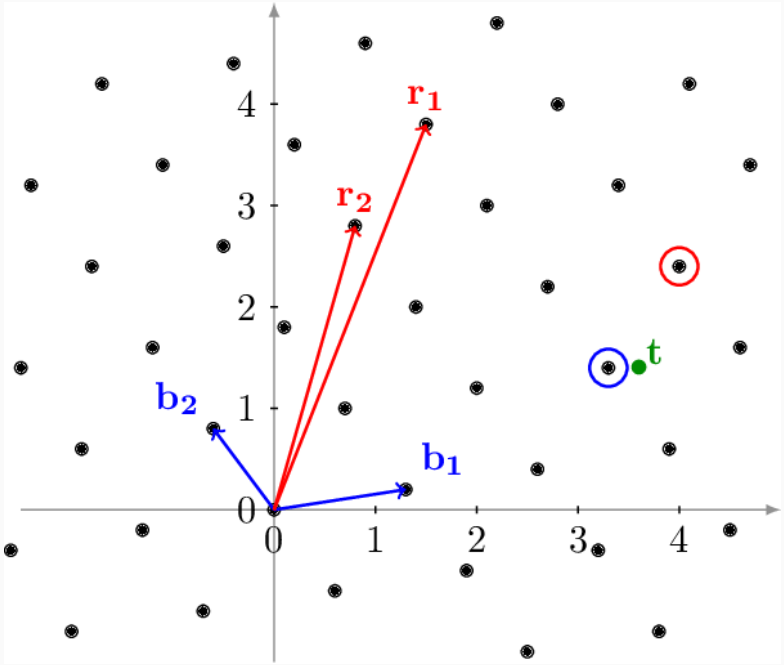
- Alice and Bob still use ordinary classical computers.
- The adversary may have a quantum computer.
- Security relies on problems believed to be hard for **both** kinds of computers.

"Post-quantum" does **not** mean that Alice and Bob send qubits.

NIST, Post-Quantum Cryptography project; adapted from the 2023 Quantum Crypto lecture.

Post-quantum cryptography from lattices

Many proposed replacements are based on **lattice problems**: finding short vectors in high-dimensional lattices.



A related problem looks like solving linear equations with small errors:

$$b = As + e \pmod{q}.$$

Given A, b, recover the secret s. The errors e make the problem much harder than ordinary linear equations.

For suitable parameters, these problems are believed to be hard for quantum computers, too.

Lattice figure and framing: October 30, 2025 slides. Noisy-equation example: Spring 2022, Lecture 8.

Post-quantum cryptography standards

The US National Institute of Standards and Technology (NIST) ran a multiyear process to select post-quantum cryptosystems.

Three standards finalized in **August 2024**:

Standard	Purpose	Based on
ML-KEM	Establishing a shared key	Lattices
ML-DSA	Digital signatures	Lattices
SLH-DSA	Digital signatures	Hash functions

Post-quantum cryptography standards

The US National Institute of Standards and Technology (NIST) ran a multiyear process to select post-quantum cryptosystems.

Three standards finalized in **August 2024**:

Standard	Purpose	Based on
ML-KEM	Establishing a shared key	Lattices
ML-DSA	Digital signatures	Lattices
SLH-DSA	Digital signatures	Hash functions

NIST recommends beginning the transition now.

These are classical algorithms. No quantum communication is needed.

NIST: FIPS 203 (ML-KEM), FIPS 204 (ML-DSA), FIPS 205 (SLH-DSA).

What if someone finds a better attack?

It is an important research problem to find better evidence that lattice problems are hard for quantum computers.

But there is always the possibility that someone finds a better algorithm.

What if someone finds a better attack?

It is an important research problem to find better evidence that lattice problems are hard for quantum computers.

But there is always the possibility that someone finds a better algorithm.

AI can help search for attacks in two different ways:

1. Find mistakes in the software implementing a cryptosystem.
2. Find mathematical weaknesses in the cryptosystem itself.

What if someone finds a better attack?

It is an important research problem to find better evidence that lattice problems are hard for quantum computers.

But there is always the possibility that someone finds a better algorithm.

AI can help search for attacks in two different ways:

1. Find mistakes in the software implementing a cryptosystem.
2. Find mathematical weaknesses in the cryptosystem itself.

A proof that a scheme is secure *assuming a problem is hard* does not prove that the problem is hard.

Adapted from October 30, 2025 slides; Anthropic, Discovering cryptographic weaknesses with Claude (July 28, 2026).

An AI-assisted attack on HAWK

HAWK was a candidate **post-quantum digital signature scheme**.

An AI-assisted attack on HAWK

HAWK was a candidate **post-quantum digital signature scheme**.

In July 2026, researchers using Claude reported a faster attack on HAWK's mathematical construction.

The designers withdrew HAWK from the NIST process on **July 29, 2026**.

An AI-assisted attack on HAWK

HAWK was a candidate **post-quantum digital signature scheme**.

In July 2026, researchers using Claude reported a faster attack on HAWK's mathematical construction.

The designers withdrew HAWK from the NIST process on **July 29, 2026**.

This exploited structure specific to HAWK. It was **not** a general algorithm for solving lattice problems, or a break of ML-KEM.

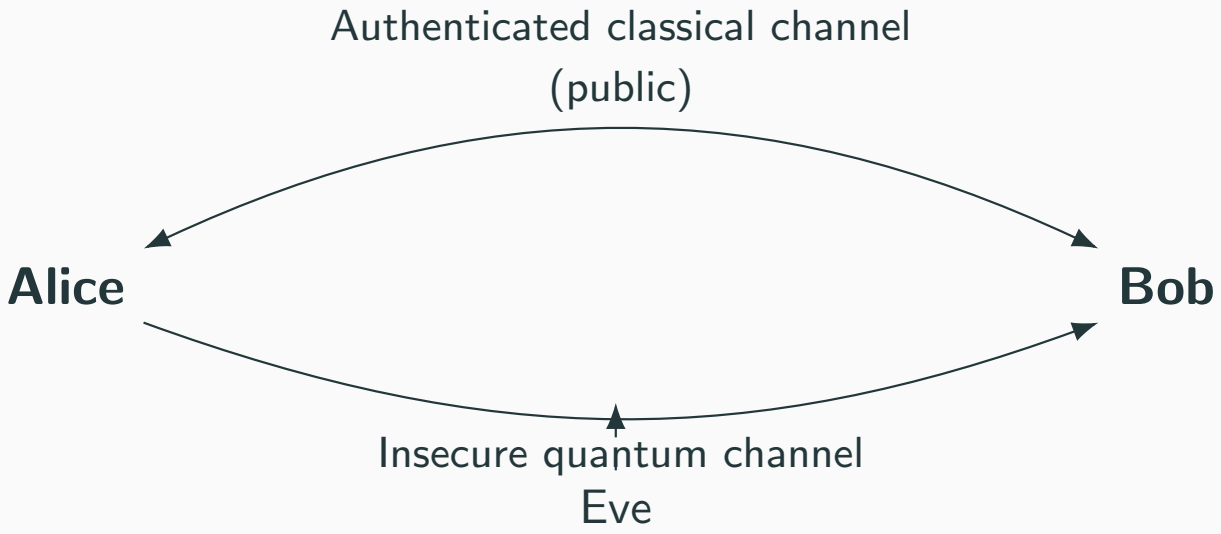
Our understanding of the best attacks can change.

Anthropic (July 28, 2026); HAWK designers' withdrawal announcement (July 29, 2026).

Quantum key distribution

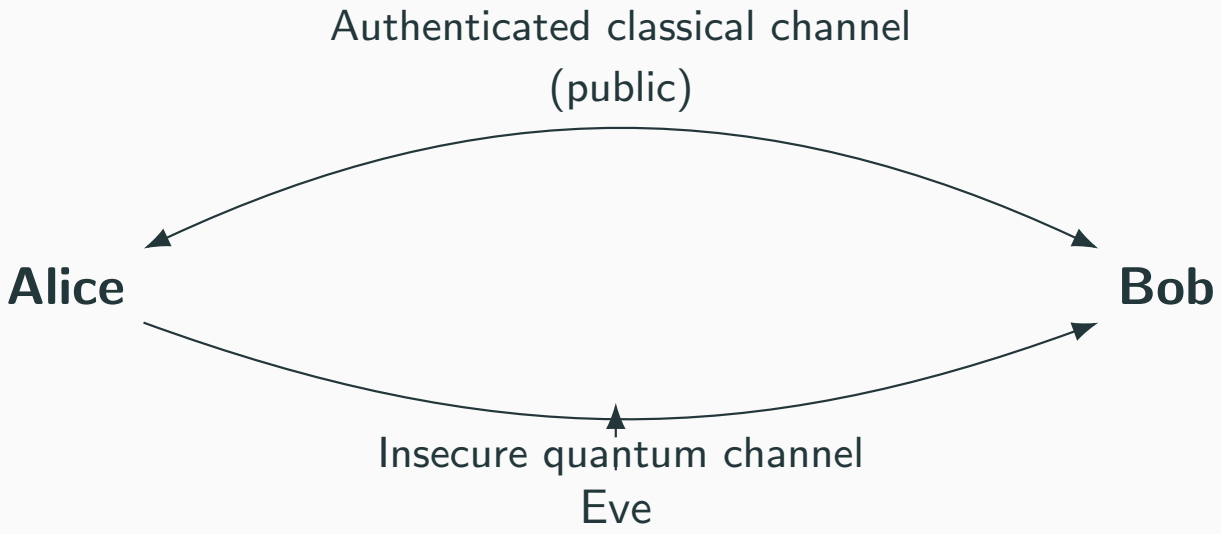
Quantum key distribution

Can Alice and Bob share a secret key even if Eve has **unlimited computational power**?



Quantum key distribution

Can Alice and Bob share a secret key even if Eve has **unlimited computational power**?



This goal is possible! Bennett and Brassard proposed the **BB84 protocol** in 1984.

We still need authentication and suitable devices.

What do we want a shared key for?

Suppose Alice and Bob share a uniformly random secret string $k \in \{0,1\}^n$.

To send $m \in \{0,1\}^n$, Alice sends

$$c = m \oplus k.$$

Bob computes $c \oplus k = m$.

What do we want a shared key for?

Suppose Alice and Bob share a uniformly random secret string $k \in \{0,1\}^n$.

To send $m \in \{0,1\}^n$, Alice sends

$$c = m \oplus k.$$

Bob computes $c \oplus k = m$.

Example:

$$\begin{array}{rr} m & 1011 \\ k & 0110 \\ \hline c & 1101 \end{array}$$

This is the **one-time pad**. The key must be secret, as long as the message, and used only once.

Why not just send random bits?

Alice chooses a random bit x and sends $|x\rangle$ to Bob.

Why not just send random bits?

Alice chooses a random bit x and sends $|x\rangle$ to Bob.

Eve measures in the standard basis.

$$|0\rangle \longrightarrow 0, \qquad |1\rangle \longrightarrow 1.$$

She learns the bit **without changing the state**.

Why not just send random bits?

Alice chooses a random bit x and sends $|x\rangle$ to Bob.

Eve measures in the standard basis.

$$|0\rangle \longrightarrow 0, \qquad |1\rangle \longrightarrow 1.$$

She learns the bit **without changing the state**.

So the no-cloning theorem, by itself, does not solve our problem.

Question: What should Alice change?

Wiesner conjugate coding

Two ways to encode a bit:

	Bit 0	Bit 1
Standard basis (S)	$\|0\rangle$	$\|1\rangle$
Diagonal basis (D)	$\|+\rangle$	$\|-\rangle$

$$|+\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}, \qquad |-\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}.$$

Wiesner conjugate coding

Two ways to encode a bit:

	Bit 0	Bit 1
Standard basis (S)	$\|0\rangle$	$\|1\rangle$
Diagonal basis (D)	$\|+\rangle$	$\|-\rangle$

$$|+\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}, \qquad |-\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}.$$

Alice chooses the bit and the basis **independently at random**.

These four states are often called the **BB84 states**.

Single qubit example

Alice sends $|+\rangle$: bit 0, encoded in the diagonal basis.

Single qubit example

Alice sends $|+\rangle$: bit 0, encoded in the diagonal basis.

Bob measures in D: He gets $|+\rangle$, so he records 0.

Single qubit example

Alice sends $|+\rangle$: bit 0, encoded in the diagonal basis.

Bob measures in D: He gets $|+\rangle$, so he records 0.

Bob measures in S: Since

$$|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle,$$

he gets 0 or 1, each with probability $1/2$.

Single qubit example

Alice sends $|+\rangle$: bit 0, encoded in the diagonal basis.

Bob measures in D: He gets $|+\rangle$, so he records 0.

Bob measures in S: Since

$$|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle,$$

he gets 0 or 1, each with probability $1/2$.

- Correct basis: correct bit.
- Wrong basis: a uniformly random bit.

BB84

1. Alice chooses a random bit string and a random basis string. She sends the corresponding BB84 states to Bob.

BB84

1. Alice chooses a random bit string and a random basis string. She sends the corresponding BB84 states to Bob.
2. Bob independently chooses a random basis for each qubit and measures it.

BB84

1. Alice chooses a random bit string and a random basis string. She sends the corresponding BB84 states to Bob.
2. Bob independently chooses a random basis for each qubit and measures it.
3. **After Bob has received and measured the qubits**, they publicly compare their basis strings.

BB84

1. Alice chooses a random bit string and a random basis string. She sends the corresponding BB84 states to Bob.
2. Bob independently chooses a random basis for each qubit and measures it.
3. **After Bob has received and measured the qubits**, they publicly compare their basis strings.
4. They discard the positions where their bases differ.

About half the bits remain. If there are no errors or eavesdropping, Alice and Bob now have the same string.

They announce the **bases**, not the bits!

BB84 example

One possible run, with no noise or eavesdropping:

Position	1	2	3	4	5	6	7	8
Alice's bit	0	1	1	0	1	0	0	1
Alice's basis	S	D	S	D	D	S	D	S
Bob's basis	S	S	S	D	S	D	D	S
Bob's bit	0	0	1	0	1	1	0	1

BB84 example

One possible run, with no noise or eavesdropping:

Position	1	2	3	4	5	6	7	8
Alice's bit	0	1	1	0	1	0	0	1
Alice's basis	S	D	S	D	D	S	D	S
Bob's basis	S	S	S	D	S	D	D	S
Bob's bit	0	0	1	0	1	1	0	1
Keep?	yes	–	yes	yes	–	–	yes	yes

They keep positions $1, 3, 4, 7, 8$. Both get **01001**.

BB84 example

One possible run, with no noise or eavesdropping:

Position	1	2	3	4	5	6	7	8
Alice's bit	0	1	1	0	1	0	0	1
Alice's basis	S	D	S	D	D	S	D	S
Bob's basis	S	S	S	D	S	D	D	S
Bob's bit	0	0	1	0	1	1	0	1
Keep?	yes	–	yes	yes	–	–	yes	yes

They keep positions $1, 3, 4, 7, 8$. Both get **01001**.

At position 5, the bits happen to agree. They still discard it: the **bases** did not agree.

What if Eve measures?

Alice sends $|+\rangle$. Eve measures in S and sends the resulting state to Bob.

What if Eve measures?

Alice sends $|+\rangle$. Eve measures in S and sends the resulting state to Bob.

Eve gets $|0\rangle$ or $|1\rangle$, each with probability $1/2$.

Bob measures in D. Recall that

$$|0\rangle = \frac{|+\rangle + |-\rangle}{\sqrt{2}}, \qquad |1\rangle = \frac{|+\rangle - |-\rangle}{\sqrt{2}}.$$

What if Eve measures?

Alice sends $|+\rangle$. Eve measures in S and sends the resulting state to Bob.

Eve gets $|0\rangle$ or $|1\rangle$, each with probability $1/2$.

Bob measures in D. Recall that

$$|0\rangle = \frac{|+\rangle + |-\rangle}{\sqrt{2}}, \qquad |1\rangle = \frac{|+\rangle - |-\rangle}{\sqrt{2}}.$$

In **either** case, Bob gets $|+\rangle$ or $|-\rangle$ with equal probability.

So Bob records the wrong bit with probability $\mathbf{1/2}$.

How often does Eve cause an error?

Suppose Eve measures **every** qubit, choosing S or D uniformly at random, and sends the resulting state to Bob.

Consider only the bits Alice and Bob **keep**.

How often does Eve cause an error?

Suppose Eve measures **every** qubit, choosing S or D uniformly at random, and sends the resulting state to Bob.

Consider only the bits Alice and Bob **keep**.

Eve's basis	Probability	Error probability
Same as Alice's	1/2	0
Different from Alice's	1/2	1/2

$$\Pr[\text{error on a kept bit}] = \frac{1}{2} \cdot 0 + \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}.$$

This analyzes **one particular attack**, not every possible attack.

Checking some of the bits

Alice and Bob publicly compare a **random sample** of their kept bits.

- If there are too many disagreements, abort.
- Discard every bit they revealed.

Checking some of the bits

Alice and Bob publicly compare a **random sample** of their kept bits.

- If there are too many disagreements, abort.
- Discard every bit they revealed.

In our example, suppose they check positions 3 and 7.

$$\underbrace{0\,1\,0\,0\,1}_{\text{positions } 1,3,4,7,8} \longrightarrow \underbrace{0\,0\,1}_{\text{positions } 1,4,8} .$$

Checking some of the bits

Alice and Bob publicly compare a **random sample** of their kept bits.

- If there are too many disagreements, abort.
- Discard every bit they revealed.

In our example, suppose they check positions 3 and 7.

$$\underbrace{0\,1\,0\,0\,1}_{\text{positions } 1,3,4,7,8} \longrightarrow \underbrace{0\,0\,1}_{\text{positions } 1,4,8} .$$

Eve did not know in advance which positions would be checked.

For the attack we just analyzed, t checked bits all agree with probability $(3/4)^t$.

Why not just copy the qubits?

Eve could try to copy each qubit, send one copy to Bob, and save the other until the bases are announced.

Why not just copy the qubits?

Eve could try to copy each qubit, send one copy to Bob, and save the other until the bases are announced.

The **no-cloning theorem** rules out making perfect copies of arbitrary BB84 states.

Why not just copy the qubits?

Eve could try to copy each qubit, send one copy to Bob, and save the other until the bases are announced.

The **no-cloning theorem** rules out making perfect copies of arbitrary BB84 states.

Trying to obtain information can disturb the states. The sample checks help Alice and Bob detect this.

This is the intuition. A full security proof also has to handle attacks much more general than measuring one qubit at a time.

Finishing the protocol

Passing the sample check is not the last step.

Finishing the protocol

Passing the sample check is not the last step.

Error correction: Alice and Bob communicate to correct remaining disagreements, so they hold the same string.

Finishing the protocol

Passing the sample check is not the last step.

Error correction: Alice and Bob communicate to correct remaining disagreements, so they hold the same string.

Privacy amplification: They shorten that string to produce a key about which Eve has very little information.

Finishing the protocol

Passing the sample check is not the last step.

Error correction: Alice and Bob communicate to correct remaining disagreements, so they hold the same string.

Privacy amplification: They shorten that string to produce a key about which Eve has very little information.

The final key must be short enough to account for the observed errors and the information revealed during the protocol.

We will not cover the algorithms or the full security proof here.

What are we still assuming?

QKD does not need a computational-hardness assumption for the secrecy of the final key. But it still needs:

- **Authenticated classical messages**. Eve must not be able to impersonate Alice or Bob.
- **Devices that satisfy the security model**, and secure handling of the keys.

What are we still assuming?

QKD does not need a computational-hardness assumption for the secrecy of the final key. But it still needs:

- **Authenticated classical messages**. Eve must not be able to impersonate Alice or Bob.
- **Devices that satisfy the security model**, and secure handling of the keys.

Authentication can start from a small pre-shared secret. QKD does not establish identity from nothing.

What are we still assuming?

QKD does not need a computational-hardness assumption for the secrecy of the final key. But it still needs:

- **Authenticated classical messages**. Eve must not be able to impersonate Alice or Bob.
- **Devices that satisfy the security model**, and secure handling of the keys.

Authentication can start from a small pre-shared secret. QKD does not establish identity from nothing.

And using a QKD key with AES still leaves message encryption dependent on AES. The one-time pad gives the stronger confidentiality guarantee.

Spring 2022 QKD notes; Portmann–Renner, Security in quantum cryptography (2022).

Post-quantum cryptography and QKD

	Post-quantum crypto	**QKD**
Basic idea	Choose different hard problems	Send quantum states to establish a key
Honest parties	Classical computers	Quantum communication devices
Key secrecy	Computational assumptions	Can hold against unlimited computation
Still needed	Authentication and secure implementations	Authentication and devices satisfying the security model

QKD lets us avoid a computational-hardness assumption. It does not eliminate the need for authentication or secure devices.

NIST PQC project; Spring 2022 QKD notes.

Source lectures

- November 6, 2026: Phase Estimation and Factoring. Recap, notation, and the $x = 7$, $N = 15$ example.
- October 23 and 30, 2025 Beamer sources. Phase estimation, order finding, resource estimates, and post-quantum cryptography. The lattice figure is reused from these slides.
- December 4, 2023: Quantum Crypto, slides 3, 6, 10–27. QKD framing, conjugate coding, and BB84. Re-typeset in Beamer.
- Spring 2022, Lecture 8: Quantum Key Distribution. Protocol details and the eavesdropping example.
- Fall 2018, Lecture 9. One-time-pad motivation.

Additional references

- Gidney and Ekerå (2021; preprint 2019), *How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits*.
- Gidney (2025), *How to factor 2048 bit RSA integers with less than a million noisy qubits*.
- NIST Post-Quantum Cryptography project; What Is Post-Quantum Cryptography?
- Anthropic (July 28, 2026), *Discovering cryptographic weaknesses with Claude*; HAWK designers' withdrawal announcement (July 29, 2026).
- Portmann and Renner (2022), *Security in quantum cryptography*.

Current factual additions checked November 12, 2026. The resource table compares two specified estimates; it is not a survey of every estimate.