Professional Certificate in Post-Quantum Cryptography

Security Proofs and Reductions

Security proof is the formal argument that a cryptographic scheme meets a defined notion of security under specified assumptions. In the context of post-quantum cryptography, a security proof must consider adversaries equipped with quantum computers, which can query oracles in superposition and exploit algorithms such as Shor's and Grover's. A proof typically proceeds by a sequence of reductions that transform an attack on the scheme into an algorithm that solves a hard mathematical problem, for example the Learning With Errors (LWE) problem or the Short Integer Solution (SIS) problem.

Reduction is the core technique used to relate the security of a construction to the difficulty of an underlying problem. A reduction is an algorithm that, given an adversary that breaks the target scheme, produces a solver for the assumed hard problem. The reduction may be black-box, meaning it treats the adversary as an oracle, or it may be non-black-box, using the internal code of the adversary. Reductions are evaluated by their tightness, which measures how closely the advantage of the solver matches the advantage of the adversary. A tight reduction incurs only a small loss, while a non-tight reduction may require the adversary to be substantially more powerful in order to yield a useful solver.

Advantage quantifies the success probability of an adversary beyond random guessing. For a decision problem, the advantage is defined as |Pr[adversary outputs 1 on a real instance] – Pr[adversary outputs 1 on a fake instance]|. In the quantum setting, advantage is measured with respect to the best possible quantum algorithm that respects the allowed queries. The advantage is a function of the security parameter n, and a proof typically shows that the advantage is negligible in n under the hardness assumption.

Security parameter is an integer that determines the size of the problem instance and the key material. In lattice-based schemes, the security parameter may be the dimension of the lattice or the bit-length of the modulus. A larger security parameter yields a larger problem and, under the assumption, a smaller advantage for any efficient adversary. The definition of "efficient" is crucial: In post-quantum proofs, "efficient" refers to algorithms running in polynomial time on a quantum Turing machine.

Hardness assumption is a conjecture that a specific computational problem cannot be solved in polynomial time by any quantum algorithm. Common assumptions in post-quantum cryptography include the hardness of the LWE problem, the SIS problem, the NTRU problem, and the decoding problem for random linear codes. A reduction demonstrates that breaking a scheme would imply solving one of these problems, thereby inheriting the presumed difficulty.

Game-based proof is a standard framework for security arguments. The proof defines a series of games, each differing slightly from the previous one, and shows that the adversary's advantage changes by only a negligible amount between games. The final game is constructed so that the adversary's success would solve the underlying hard problem. The technique is called a hybrid argument. For example, to prove IND-CPA security of a lattice encryption scheme, one may start with Game 0 (the real encryption

experiment) and gradually replace the public key with a uniformly random matrix, then replace the ciphertext with an encryption of a random message, and finally replace the ciphertext with a sample from the error distribution. Each transition is justified by a reduction that uses the adversary to distinguish the two games, which in turn would break the LWE assumption.

Indistinguishability (IND) is a central security notion. IND-CPA (indistinguishability under chosen-plaintext attack) requires that no efficient adversary can distinguish encryptions of two chosen messages. IND-CCA (indistinguishability under chosen-ciphertext attack) strengthens this by allowing the adversary to query a decryption oracle on ciphertexts other than the challenge. In the quantum realm, definitions are refined to allow the adversary to make superposition queries to the encryption and decryption oracles. The Quantum Random Oracle Model (QROM) captures this capability by treating the random oracle as a quantum-accessible unitary transformation.

Random oracle model (ROM) is an idealized setting where a hash function is modeled as a truly random function that parties can query. In classical proofs, the ROM simplifies reductions by allowing the simulator to program the oracle's responses. In the QROM, the adversary may query the oracle on a superposition of inputs, and the simulator must respond consistently with a unitary that preserves quantum interference. Proving security in the QROM is significantly more challenging, often requiring sophisticated techniques such as the "measure-and-program" method or the use of quantum-secure extraction lemmas.

Hybrid argument is a method for bounding the difference in advantage between two games. The argument constructs a chain of intermediate games G0, G1, …, Gk, where each adjacent pair differs by a small, well-understood change. By the triangle inequality, the total difference between G0 and Gk is at most the sum of the differences between each pair. Each pairwise difference is bounded by a reduction to a known hard problem. The hybrid technique is widely used in lattice-based proofs, for instance to argue that replacing a LWE matrix with a uniform matrix does not noticeably affect the adversary's view.

Rewinding is a technique used in many classical reductions, especially those involving zero-knowledge proofs or signatures. The simulator runs the adversary, records its queries, and then "rewinds" the adversary to a previous state to extract information. In the quantum setting, rewinding is problematic because quantum states cannot be copied due to the no-cloning theorem. Consequently, many quantum-secure reductions avoid rewinding, or they employ specialized quantum rewinding lemmas that work under restricted conditions, such as when the adversary's measurement outcomes are classical.

Extractor is an algorithm that, given access to an adversary, extracts a witness (for example, a secret key) from the adversary's behavior. In proofs of knowledge, an extractor demonstrates that any prover that convinces a verifier must "know" a secret. In the context of reductions, an extractor may be used to recover a solution to the underlying hard problem from the adversary's responses. The extraction may be black-box, relying only on the adversary's input-output behavior, or may require more invasive techniques.

Simulation is the process by which a reduction mimics the environment of the target scheme for the adversary while internally using the hard-problem oracle. The simulator must answer all queries of the adversary in a way that is indistinguishable from the real scheme. In the ROM, the simulator can program the random oracle; in the QROM, it must ensure that the quantum oracle's unitary operation is consistent

with the programmed values. The quality of the simulation directly impacts the tightness of the reduction.

Black-box reduction treats the adversary as an oracle, without examining its internal code. The reduction can only query the adversary's inputs and observe its outputs. Black-box reductions are often easier to construct and analyze, but they may incur a loss in tightness because the reduction cannot exploit structural properties of the adversary. In post-quantum proofs, black-box reductions are preferred when possible, since they avoid the need for quantum rewinding.

Non-black-box reduction can inspect the adversary's source code or internal state. This additional power can lead to tighter reductions, but it is more difficult to implement in the quantum setting because the adversary's state may be entangled with the environment. Non-black-box techniques are occasionally used in lattice-based signatures where the reduction extracts a short vector from the adversary's signing algorithm.

Worst-case to average-case reduction is a cornerstone of lattice-based cryptography. It shows that solving a random instance of a lattice problem (average case) is at least as hard as solving the hardest instance (worst case). For LWE, the reduction demonstrates that an algorithm that solves LWE on average can be used to solve the GapSVP (Shortest Vector Problem) on any lattice. This type of reduction provides confidence that the problem does not have hidden easy instances, and it forms the basis for many security proofs.

Tight reduction is a reduction where the advantage of the solver is within a small multiplicative factor of the adversary's advantage. Formally, if $\mathrm{Adv}_A \le \varepsilon$, then the reduction produces a solver with advantage at least ε / c, where c is a constant close to 1. Tightness is desirable because it allows the designer to set parameters that directly reflect the assumed hardness. In contrast, a non-tight reduction with a large loss factor forces the designer to increase key sizes or error rates to compensate.

Non-tight reduction incurs a larger loss, often polynomial in the security parameter. For example, a reduction that introduces a factor of n (the lattice dimension) may require the scheme to use a lattice of size $n \approx 2 \times$ the desired security level to achieve the same security guarantee. Non-tightness is a practical challenge because it can lead to inefficient parameters, especially in resource-constrained environments.

Hybrid argument for indistinguishability is a concrete instantiation of the hybrid technique. Suppose an encryption scheme encrypts a message by adding an LWE error term. The proof may define hybrids where the error term's distribution is gradually altered from the true error distribution to a uniform distribution. Each hybrid step is justified by the LWE assumption: If an adversary could distinguish the two neighboring hybrids, it would break LWE. By chaining the steps, the proof shows that the adversary's overall advantage is bounded by the sum of the LWE distinguishing advantages.

Quantum query model specifies how an adversary may interact with oracles. In the classical model, queries are made one at a time and the oracle returns a deterministic answer. In the quantum model, the adversary may prepare a superposition of inputs, apply the oracle as a unitary, and receive a superposition of outputs. The model is essential for defining security in the QROM, and it influences the design of reductions. For instance, a reduction that relies on measuring the adversary's query must ensure that the measurement does not collapse the superposition prematurely.

Quantum indistinguishability extends the classical notion by requiring that no quantum polynomial-time algorithm can distinguish two distributions with non-negligible advantage, even when allowed to make quantum queries. In practice, this often translates to proving that the statistical distance between the two quantum states is negligible. Many lattice-based schemes achieve quantum indistinguishability by showing that the ciphertext distribution is statistically close to a uniform distribution, which remains true even under quantum observation.

Statistical distance (also called total variation distance) measures how far two probability distributions are from each other. In the quantum setting, the analogous quantity is the trace distance between two density matrices. A reduction may bound the statistical distance between the real ciphertext distribution and an ideal distribution, thereby limiting the adversary's advantage. For LWE-based encryption, the error term is drawn from a discrete Gaussian; the proof often shows that the resulting distribution is statistically close to uniform, up to a negligible term.

Trace distance is the quantum analogue of statistical distance. If ρ and σ are density matrices, the trace distance is $(1/2)\cdot\|\rho - \sigma\|_1$, where $\|\cdot\|_1$ denotes the Schatten 1-norm. Bounding the trace distance is a key step in many QROM proofs, because it directly limits the distinguishing advantage of a quantum adversary. Techniques such as the "measure-and-program" lemma exploit the fact that a quantum-accessible random oracle can be programmed without significantly increasing the trace distance.

Measure-and-program lemma is a tool for constructing quantum-secure random oracles. The lemma states that a simulator can measure the adversary's query in a suitable basis, program the oracle's response accordingly, and still maintain a low trace distance between the simulated and real oracles. This approach avoids the need for full quantum rewinding and enables tighter reductions in the QROM. The lemma is applied, for example, in the analysis of hash-based signatures that rely on the Fiat-Shamir transform.

Fiat-Shamir transform converts an interactive proof into a non-interactive signature scheme by replacing the verifier's random challenge with a hash of the transcript. In the quantum setting, the transform must be analyzed in the QROM because the adversary can query the hash in superposition. Security proofs for post-quantum signatures such as Dilithium or Falcon often use a variant of the Fiat-Shamir transform and require a careful reduction that respects quantum queries. The proof typically shows that any forgery leads to a solution of the underlying lattice problem.

Forking lemma is a classical tool used to analyze signature schemes in the ROM. It states that if an adversary can produce a valid signature with non-negligible probability, then by rewinding and providing a different random oracle response, the simulator can obtain two forgeries that share most of the transcript. In the quantum setting, the forking lemma does not directly apply because quantum rewinding is problematic. Researchers have developed quantum-aware forking techniques, often at the cost of a loss in tightness.

Quantum-aware forking adapts the classical forking lemma to the QROM by employing a "measure-and-reprogram" strategy. The adversary's quantum queries are measured in a basis that reveals the hash input without destroying the quantum state needed for the rest of the execution. After measurement, the simulator can reprogram the hash value for a second run. The resulting reduction may incur an additional factor related to the number of queries, which must be accounted for in the security

parameters.

Adaptive security refers to the ability of an adversary to choose its queries based on previous answers. In IND-CCA security, the adversary may adaptively request decryptions of ciphertexts that are related to the challenge ciphertext. Security proofs must handle adaptive queries, often by using a sequence of games where each game limits the adversary's ability to make certain queries. In the quantum setting, adaptivity is more subtle because the adversary's queries may be entangled with its internal state.

Non-adaptive security is a weaker notion where the adversary must fix all its queries before seeing any answers. Many lattice-based encryption schemes achieve IND-CPA security non-adaptively, but extending the proof to adaptive security typically requires additional techniques such as a "dual-mode" encryption or the use of trapdoors.

Trapdoor is a piece of secret information that enables efficient sampling from a lattice distribution that would otherwise be hard. Trapdoors are central to many lattice-based constructions, including the GPV signature scheme and the NIST-standardized KEMs. In a security proof, the trapdoor is often used by the simulator to generate public keys that are indistinguishable from real keys while still allowing the reduction to embed a hard problem instance. The existence of a trapdoor is justified by algorithms such as the "sampling from a discrete Gaussian over a lattice with a trapdoor" procedure.

Gaussian sampling is the process of drawing lattice vectors according to a discrete Gaussian distribution centered at a target point. The parameter σ (sigma) controls the width of the distribution. Properly chosen σ ensures that the sampled vectors are short enough for security while still being statistically close to uniform. Security reductions frequently require that the sampling algorithm be efficient and that its output distribution be close to the ideal Gaussian. In the quantum setting, it is crucial that the sampling algorithm be implementable with quantum-compatible primitives, otherwise the reduction could be invalid.

Rejection sampling is a technique used to obtain a desired distribution by sampling from a simpler distribution and rejecting samples that do not meet a certain criterion. In lattice signatures, rejection sampling is used to hide the secret key and achieve statistical zero-knowledge. The security proof must argue that the rejection probability is bounded and that the resulting distribution is close to the target distribution. In the QROM, the analysis must account for the fact that the adversary may query the sampling routine in superposition, which can affect the rejection probability.

Zero-knowledge is a property of interactive proofs where the verifier learns nothing beyond the validity of the statement. In the context of signatures, zero-knowledge often appears as "statistical zero-knowledge" or "computational zero-knowledge". A security proof that a signature is zero-knowledge typically constructs a simulator that can produce a transcript indistinguishable from a real interaction without knowledge of the secret key. The simulator's ability to program the random oracle is essential, and in the quantum setting the simulator must respect the QROM constraints.

Statistical zero-knowledge requires that the simulated transcript be statistically close to the real transcript, independent of computational assumptions. This stronger notion is valuable for post-quantum schemes because it does not rely on unproven hardness assumptions for the zero-knowledge property itself. Proving

statistical zero-knowledge often involves bounding the statistical distance between the real and simulated distributions, which may be done using coupling arguments or the leftover hash lemma.

Computational zero-knowledge relaxes the requirement to indistinguishability by any efficient adversary. The proof typically relies on a hardness assumption such as LWE. In the quantum setting, the assumption must hold against quantum adversaries, and the reduction must be quantum-secure. The distinction between statistical and computational zero-knowledge is important when selecting schemes for applications that demand strong privacy guarantees.

Leftover hash lemma is a tool that guarantees that hashing a high-entropy source with a universal hash function yields a distribution that is close to uniform. In lattice-based cryptography, the lemma is used to argue that the output of a Gaussian sampler, when hashed, hides the secret. The proof relies on the min-entropy of the source and the universality of the hash family. In the QROM, the lemma must be adapted to account for quantum side information, leading to the “quantum leftover hash lemma”.

Quantum leftover hash lemma extends the classical lemma by considering an adversary that holds a quantum state correlated with the source. The lemma states that if the source has sufficient conditional min-entropy given the quantum side information, then the hashed output is close to uniform from the adversary’s perspective. This result is instrumental in proving security of schemes that combine lattice sampling with hash-based masking, such as the Kyber KEM.

Conditional min-entropy measures the amount of randomness in a classical random variable X conditioned on a quantum system E. Formally, $H_{min}(X|E) = -\log \max_{\sigma_E} F(\rho_{XE}, I_X \otimes \sigma_E)^2$, where F denotes the fidelity. In security proofs, a high conditional min-entropy guarantees that the adversary cannot predict X even with quantum side information. The reduction often shows that breaking the scheme would lead to a violation of the assumed min-entropy bound.

Fidelity quantifies the similarity between two quantum states. For density matrices ρ and σ, the fidelity is defined as $F(\rho,\sigma) = \|\sqrt{\rho}\sqrt{\sigma}\|_1^2$. In reductions, fidelity is used to relate the trace distance to the advantage of a quantum adversary. A small fidelity loss indicates that the simulated oracle remains close to the ideal one, preserving the adversary’s success probability.

Hybrid argument in the QROM must account for the fact that the adversary can query the random oracle in superposition. The proof typically uses a series of “oracle-programming” steps where the simulator changes the oracle’s behavior on a small set of inputs. The analysis shows that each change introduces at most a negligible increase in trace distance, and therefore the overall advantage remains bounded. This approach is used in the security analysis of lattice-based KEMs such as Kyber, where the decryption oracle is programmed to embed an LWE instance.

Embedding is the act of inserting a hard problem instance into the public parameters of a scheme. For example, in a reduction from LWE to a KEM, the simulator may set the public matrix A to be the LWE matrix provided by the challenger. The rest of the scheme’s parameters are generated normally, and the reduction ensures that any successful attack on the KEM yields a solution to the original LWE instance. Embedding is a central concept in many reductions, and its correctness depends on the indistinguishability of the modified

distribution from the original.

Simulation-soundness is a property of proof systems where a proof that appears valid to a verifier (even after many simulated interactions) does not enable a cheating prover to create a false proof. In the context of signature schemes, simulation-soundness ensures that even after the attacker has seen many simulated signatures, it cannot forge a new signature. Proving simulation-soundness often involves a reduction that uses the forger to solve a hard problem, and the reduction must be careful to preserve the quantum oracle's consistency.

Commit-then-open is a technique used in zero-knowledge protocols where the prover first commits to a value and later opens the commitment. In lattice-based protocols, commitments are often built from Gaussian samples. The security proof must show that the commitment is binding (the prover cannot open it to two different values) and hiding (the verifier learns nothing before the opening). The binding property is typically reduced to the hardness of finding short vectors, while the hiding property may rely on the statistical closeness of the commitment distribution to uniform.

Binding property guarantees that a commitment cannot be opened to two distinct values. In lattice commitments, binding is often proven by showing that two different openings would yield a short lattice vector, contradicting the hardness of SIS. The reduction constructs an SIS solver that receives the two openings as input and outputs a short vector. The tightness of this reduction influences the commitment's parameters, such as the modulus size and the Gaussian width.

Hiding property ensures that a commitment reveals no information about the committed value. For lattice commitments, the hiding proof typically uses the leftover hash lemma to argue that the commitment distribution is close to uniform. In the quantum setting, the proof must also consider quantum side information, invoking the quantum leftover hash lemma. The reduction may involve an adversary that distinguishes the commitment from uniform, which would break the LWE assumption.

Key-indistinguishability (KI) is a security notion for public-key encryption where the public key should be indistinguishable from a random string. This property is useful for constructing hybrid encryption schemes and for proving security under key-replacement attacks. A reduction from KI to LWE shows that an adversary that distinguishes a real public key from random can be turned into an LWE distinguisher. The proof often uses a hybrid where the public key matrix is replaced by a uniform matrix, and the statistical distance between the two is bounded by the LWE advantage.

Key-privacy is a stronger property that requires ciphertexts to hide the identity of the recipient's public key. In broadcast encryption and anonymous KEMs, key-privacy prevents an adversary from learning which user a ciphertext is intended for. The proof typically combines IND-CPA security with a key-indistinguishability argument, and may involve a reduction that uses a forgery to solve a decision variant of the underlying lattice problem.

Decision vs. Search problems are two flavors of computational problems. A decision problem asks for a yes/no answer (e.G., "Does a short vector exist?"), While a search problem asks to produce an explicit solution (e.G., "Find a short vector"). In many lattice assumptions, the decision version (e.G., Decisional LWE) is

believed to be as hard as the search version. Security proofs sometimes require a reduction from a decision problem to the scheme's security, while other times a search-to-decision reduction is employed to translate an attack on the scheme into a concrete solution.

Search-to-decision reduction shows that solving the search version of a problem would also solve the decision version. For LWE, the reduction uses the fact that an oracle that distinguishes LWE samples can be turned into an algorithm that recovers the secret vector. The proof constructs a series of hybrid samples where the secret is gradually revealed, and each step uses the decision oracle to test consistency. This reduction is essential when the security definition is based on a decision problem, but the scheme's hardness is expressed in terms of a search problem.

Decision-to-search reduction works in the opposite direction, often required when a scheme's security is tied to a decision problem but the underlying hardness assumption is expressed as a search problem. The reduction may involve a "guessing" technique, where the algorithm tries multiple candidates for the secret and uses the decision oracle to verify each guess. The efficiency of this reduction determines the tightness of the overall proof.

Hybrid encryption combines a public-key encryption (or KEM) with a symmetric-key encryption to achieve efficiency. In a post-quantum setting, the KEM is usually lattice-based, while the symmetric encryption uses classical algorithms that are believed to be quantum-resistant (e.G., AES). Security proofs for hybrid encryption must show that the composition remains IND-CCA secure. The proof typically uses a sequence of games: First replace the KEM ciphertext with an encryption of a random key, then replace the symmetric ciphertext with an encryption of a random message, and finally argue that any difference would break the IND-CCA security of either component.

Encapsulation-decapsulation (KEM) is a primitive where a sender encapsulates a secret key using a public key, producing a ciphertext, and a receiver decapsulates using the private key to recover the same secret. KEMs are a central building block for post-quantum key exchange. Security proofs for KEMs often adopt the "IND-CPA" notion for the encapsulation step, and then extend to "IND-CCA" by adding a decryption oracle for the decapsulation. The reduction may embed a hard LWE instance into the public key, and a successful attack on the KEM yields an LWE solver.

IND-CPA security for KEM asserts that an adversary cannot distinguish the encapsulated key from a random key, even after seeing the public key. The proof constructs a hybrid where the encapsulated key is replaced by a uniform key, and argues that any distinguishing advantage would break the underlying LWE assumption. The reduction often requires the simulator to program the random oracle in the QROM, ensuring that the decapsulation algorithm behaves consistently with the programmed values.

IND-CCA security for KEM strengthens the guarantee by allowing the adversary to query a decapsulation oracle on any ciphertext except the challenge ciphertext. The proof typically uses a "simulation-based" technique where the simulator answers decapsulation queries by re-encrypting the queried ciphertext with a simulated secret, or by using the trapdoor to compute the correct shared secret. The reduction must ensure that the simulation does not reveal the embedded LWE instance, which is achieved by careful programming of the random oracle and by bounding the number of queries.

Replay attacks are a practical threat where an adversary reuses a previously captured ciphertext to gain information or cause denial of service. Security proofs often assume that the scheme includes mechanisms such as nonce or session identifiers to prevent replay. In lattice-based KEMs, the encapsulation process typically includes randomness that makes each ciphertext fresh, and the proof shows that this randomness is indistinguishable from uniform, thus thwarting replay.

Chosen-plaintext attack (CPA) allows the adversary to request encryptions of arbitrary messages. In the lattice setting, CPA security is often achieved by the inherent randomness of the encryption algorithm. The proof shows that the ciphertext distribution does not reveal any information about the plaintext beyond what is allowed by the hardness assumption. In the QROM, the proof must also consider that the adversary can query the encryption oracle in superposition, which may affect the analysis of the randomness.

Chosen-ciphertext attack (CCA) extends CPA by granting the adversary decryption capabilities. To achieve CCA security, lattice-based schemes frequently employ a “Fujisaki-Okamoto” (FO) transformation. The FO transform takes a CPA-secure encryption scheme and adds a hash-based verification step to detect malformed ciphertexts. The security proof for FO in the QROM is intricate: It must bound the adversary’s ability to produce a valid ciphertext that passes the hash check, and it typically uses a hybrid argument that gradually replaces the hash function with a random oracle.

Fujisaki-Okamoto transform works as follows: To encrypt a message m, the sender first hashes m to obtain a random value r, then encrypts r using the underlying CPA scheme, and finally outputs the ciphertext together with a tag derived from the hash of the ciphertext and m. Decryption recomputes the tag and checks consistency before recovering m. The proof of security in the QROM shows that any adversary that can forge a valid ciphertext must either break the underlying CPA security or find a collision in the hash function, both of which are assumed hard.

Collision resistance is a property of hash functions stating that it is computationally infeasible to find distinct inputs that map to the same output. In the QROM, collision resistance is often assumed to hold against quantum adversaries, which may use algorithms like the quantum birthday attack that runs in O(2^{n/2}) time. Security proofs for schemes that rely on collision resistance must account for this quantum speedup, typically by choosing hash output lengths that remain secure against such attacks.

Quantum birthday attack exploits the fact that a quantum algorithm can find collisions in a hash function with roughly the square root of the classical effort. For an n-bit hash, a classical attacker needs about 2^{n/2} queries, while a quantum attacker needs about 2^{n/3} queries using the BHT algorithm. This reduction in complexity influences parameter selection: A hash used in a post-quantum scheme must have enough output bits to remain collision-resistant even against quantum attacks.

Parameter selection is the practical process of choosing dimensions, modulus sizes, error distributions, and hash output lengths to achieve a target security level. The security proof provides formulas that relate the advantage of an adversary to the underlying hardness assumption and the reduction loss. Designers must invert these relationships to determine concrete parameters. For example, to achieve 128-bit security against quantum attacks, a lattice-based KEM may use a dimension $n \approx 768$, a modulus $q \approx 2^{15}$, and a Gaussian width σ chosen to balance correctness and security. The proof ensures that with these parameters,

any quantum adversary's advantage is below 2^{-128}.

Correctness is the guarantee that honest participants can successfully encrypt and decrypt (or encapsulate and decapsulate) with overwhelming probability. In lattice schemes, correctness depends on the error term being small enough that decryption recovers the original message after rounding. The proof of correctness typically uses concentration bounds for discrete Gaussians, such as the Chernoff bound, to show that the probability of decryption failure is negligible. Correctness must be maintained even when the scheme is used in a hybrid construction, where the encapsulated key is later used for symmetric encryption.

Decryption failure occurs when the noise introduced during encryption exceeds the decoding radius, causing the receiver to output an incorrect plaintext. While rare, decryption failures can be exploited in attacks such as "fault injection" or "reaction" attacks. Security proofs often assume that decryption failures are bounded by a negligible probability ε. In practice, designers must set parameters so that ε is far below the target security level, and may employ techniques like "failure-rate reduction" by using a rejection sampling step that discards ciphertexts with high error magnitude.

Failure-rate reduction techniques aim to lower the decryption failure probability without sacrificing security. One approach is to repeat the encryption with independent randomness until the error falls within a safe range, at the cost of increased ciphertext size. Another method is to use a "decoding" algorithm that corrects small errors, such as the nearest-plane algorithm. The security proof must be updated to reflect the modified distribution of ciphertexts and ensure that the reduction still holds.

Nearest-plane algorithm is a lattice decoding method that projects a point onto the nearest hyperplane of a basis, iteratively reducing the error. The algorithm runs in polynomial time and is used in many lattice-based decryption routines. Its correctness analysis relies on the basis being sufficiently orthogonal, which is guaranteed by the trapdoor construction. In security proofs, the algorithm's success probability is incorporated into the bound on decryption failure.

Basis reduction algorithms such as LLL or BKZ improve the quality of a lattice basis, making it more orthogonal and shorter. In the context of reductions, basis reduction is used to construct trapdoors and to sample short vectors. The security proof may assume that the basis reduction algorithm runs in polynomial time and that the resulting basis satisfies certain geometric properties, which are required for the Gaussian sampling to be efficient and for the reduction to be tight.

BKZ reduction is a stronger reduction algorithm that achieves better approximation factors than LLL, at the expense of higher computational cost. Some security proofs for lattice schemes rely on BKZ to argue that the worst-case hardness of GapSVP reduces to average-case hardness of LWE with a certain approximation factor. The reduction's parameters, such as the block size β, affect the tightness and consequently the concrete security estimates.

GapSVP (Shortest Vector Problem) is the decision version of the shortest vector problem: Given a lattice, decide whether its shortest non-zero vector has length ≤ γ or ≥ α·γ for some approximation factor α. The reduction from GapSVP to LWE shows that solving LWE on average is at least as hard as approximating SVP within a factor of roughly √n. This worst-case to average-case link is a cornerstone of the security argument

for LWE-based schemes.

Shortest Vector Problem (SVP) is the search version: Find a non-zero lattice vector of minimal length. The hardness of SVP underlies many lattice-based constructions. Security proofs may need to assume that no quantum algorithm can solve SVP within a certain approximation factor in polynomial time. This assumption is stronger than the decisional LWE assumption, but it provides a more solid foundation for the security of schemes that rely on the existence of short vectors.

Reduction loss quantifies how much the advantage degrades when moving from the adversary's success probability to the solver's success probability. A reduction loss of O(n) means that the solver's advantage is at most the adversary's advantage divided by n. Losses arise from factors such as the number of oracle queries, the need to guess random coins, or the probability of successful rewinding. Tight reductions aim to keep this loss as low as possible, ideally O(1).

Oracle access refers to the ability of an algorithm to query a black-box function. In security proofs, the reduction often receives oracle access to the adversary (to forward its queries) and to the hard problem (to embed the instance). The type of oracle (classical or quantum) determines the techniques that can be used. For example, quantum oracle access enables the adversary to perform amplitude amplification, which can increase its advantage, and the reduction must therefore be robust against such enhancements.

Amplitude amplification is a quantum technique that generalizes Grover's search, allowing an algorithm to increase the probability of finding a marked item. In the context of security proofs, amplitude amplification can be used by a quantum adversary to boost the success probability of an attack that would otherwise be negligible. The reduction must account for this possibility, often by assuming that the underlying hard problem remains hard even when the attacker can amplify its success.

Grover's algorithm provides a quadratic speedup for unstructured search problems.