

Every term explained, in plain English

You kept meeting words like CBOM, X25519MLKEM768, Shor, Grover and HelloRetryRequest, and none of them were explained. This is a dictionary for you. Every entry is two or three sentences in ordinary language, plus a line on why it matters in this world when that helps.

Read Part A first, because everything else leans on it. After that you can jump straight to whatever word someone has just used in front of you. Nothing here assumes you have studied cryptography.

Contents

The whole picture in one minute	3
Part A: The basic building blocks	3
The two kinds of locks	3
How information is fingerprinted and protected	5
How two systems talk	5
Where keys live	6
Easy mistakes	7
Part B: Every algorithm name you will see	7
A to C	7
D to G	8
H to M	9
P to R	10
S to X	10
The libraries a report will name	11
Part C: The quantum words	12
Part D: The scanning words	14
What a scanner is	14
Where the scanner looks	15
The endpoint scan	16
Running it yourself	17
Part E: What is inside a report	17
Part F: The other tools we compare against	19
Part G: Plans, accounts and the runner	20
Part H: Singapore, the regulators and the Handbook	21
Part I: The workbench and migration words	23
The twelve words that matter most	24

The whole picture in one minute

There are only two kinds of locks on the internet, and the quantum story is entirely about the difference between them.

The first kind lets two strangers agree on a secret and prove who they are. It is called public key cryptography, and its famous names are RSA, Diffie-Hellman, ECDSA and Ed25519. A powerful quantum computer breaks this kind completely, and no bigger key size saves it.

The second kind scrambles the actual data once everyone has agreed on a secret. It is called symmetric cryptography, and its famous names are AES and ChaCha20. A quantum computer only weakens this kind, so a longer key is usually enough.

That single difference explains everything else. It explains why governments have set deadlines, why the replacement algorithms are already written down, and why the very first job is not buying anything. The first job is finding out which locks, in which version, are already inside your code and your systems.

That last job, finding the locks, is what this product does. The rest of this document explains every word you will see while it does that job.

Part A: The basic building blocks

The two kinds of locks

Cryptography

The mathematics of keeping secrets and proving who sent something. It runs quietly behind every padlock in your browser, every login and every payment. Why it matters here: every other term in this document is a piece of cryptography.

Encryption

Scrambling information so that only someone with the right key can read it. The scrambled result looks like random noise.

Plaintext and ciphertext

Plaintext is the readable original. Ciphertext is the scrambled version. Encrypting turns one into the other, and decrypting turns it back.

Key

The secret value that makes encryption work, like the key to a physical lock. The same lock can be opened by different keys, and losing the key loses the data. Why it matters here: the type and length of a key decides whether a quantum computer can break it, so most findings in a report are really statements about keys.

Algorithm

A named recipe for a cryptographic job, such as RSA or AES. The recipe is public and the key is secret. Security comes from the key, never from hiding the recipe.

Cipher

Another word for an encryption algorithm. You will also meet "cipher suite", which means a whole bundle of algorithms used together in one connection.

Symmetric cryptography

One shared secret does both the locking and the unlocking, which makes it fast. It protects the bulk of your data. Why it matters here: quantum computers do not break it, they only weaken it, so the usual fix is simply a longer key such as AES-256.

Asymmetric cryptography, also called public key cryptography

Two related keys instead of one: a public key anyone may have, and a private key only you hold. This is how strangers agree on a secret and how a website proves it is genuine. Why it matters here: a quantum computer breaks this kind completely, so this is the half of cryptography that must be replaced.

Public key and private key

The public key is shareable and is used to encrypt something for you or to check your signature. The private key is the secret half and must never be shared.

Digital signature

A mathematical stamp proving that a message came from the holder of a particular private key and was not altered. It is used for certificates, software updates, code signing and login tokens. Why it matters here: signatures are one of the two things a quantum computer breaks, so they move to ML-DSA or SLH-DSA.

Certificate

A signed electronic ID card for a website or service. It carries the name, a public key, the organisation that issued it and an expiry date. Your browser checks it before trusting a site.

Certificate authority, or CA

The organisation that signs certificates, such as Let's Encrypt or DigiCert. Why it matters here: if your certificate is built on a quantum-vulnerable key, you cannot fix it alone. The CA must support the new algorithm too, which is why migration is a supply chain problem as well as a technical one.

How information is fingerprinted and protected

Hash, also called digest or fingerprint

Turns any input into a short fixed-length string. It only works one way, and a one character change produces a completely different result. It is used to check that data has not changed and to store passwords. Why it matters here: hashes are not broken by quantum computers, although the old ones, MD5 and SHA-1, are already broken today for other reasons.

MAC and HMAC

A MAC is a short tag proving a message was not tampered with, using a shared secret. HMAC is the common way to build one from a hash. Both are symmetric, so a quantum computer does not break them.

Key derivation function, or KDF

A recipe that turns something like a password into a proper key, usually by being deliberately slow so that guessing is expensive. PBKDF2, bcrypt, scrypt and Argon2 are the well known ones.

Salt

A random value mixed into a password before hashing, so that two users with the same password do not get the same result and pre-built attack tables stop working.

IV and nonce

A number used once alongside a key. IV stands for initialization vector, and a nonce is simply a number used once. Reusing either with the same key can be catastrophic rather than merely weak, so scanners flag repeated or hardcoded ones.

Randomness, entropy and CSPRNG

Good cryptography needs unpredictable numbers to build keys. A CSPRNG is a cryptographically secure random number generator, such as SecureRandom in Java or crypto.randomBytes in Node. Using an ordinary random function to make a key is a genuine weakness and scanners flag it.

How two systems talk

Protocol

An agreed set of rules for a conversation between two systems. TLS is the protocol that secures web traffic. Protocols have versions, and old versions are usually the problem.

Handshake

The opening conversation in which two systems agree on a protocol version, a cipher suite and a key. Our endpoint scan deliberately performs this conversation to see what a server chooses.

Cipher suite

The exact bundle of algorithms chosen for one connection, for example TLS_AES_256_GCM_SHA384. Reading it tells you what encryption and integrity protection are actually in use.

Key exchange

How two systems that have never met agree on a shared secret across an open network. Diffie-Hellman, ECDH and X25519 are the classical ways. ML-KEM is the post-quantum replacement.

Forward secrecy

A property where each session uses a temporary key, so stealing the long-term key later does not unlock traffic recorded earlier. Static RSA key exchange does not have it, and scanners flag that.

TLS and HTTPS

TLS is the protocol that encrypts traffic between a browser and a server. HTTPS is ordinary web traffic carried inside TLS. The padlock in your browser is TLS doing its job.

SSH

The protocol used to log into servers and move files securely. It has its own keys and algorithms, with their own quantum problem, mostly around host keys and key exchange.

VPN

A private tunnel over the internet, using protocols such as IPsec, IKE or WireGuard. WireGuard uses X25519, so it needs a post-quantum plan too.

Where keys live

Keystore, or key store

A file that holds keys and certificates, such as Java's JKS or the industry standard PKCS#12. Finding one in a repository or inside a container tells you there are keys to inventory and probably to re-issue.

HSM, KMS, TPM and PKCS#11

Ways to keep keys out of ordinary files. HSM is a hardware security module, a dedicated device. KMS is a cloud key management service. TPM is a small chip on a computer. PKCS#11 is the standard way software talks to such devices. Why it matters here: for these, migration may mean a firmware update or a hardware replacement rather than a code change, so they belong early in the plan.

Token and JWT

A JWT is a signed token that proves a login, widely used in web APIs. It is signed with an algorithm such as RS256, ES256 or HS256. The RS and ES families are RSA and elliptic curve based and therefore quantum vulnerable. HS256 is symmetric so quantum computers do not break it, but its strength depends entirely on keeping one secret secret.

Easy mistakes

Encoding, and Base64

Encoding changes how data is written down, it does not hide it. Base64 is reversible by anyone. Treating Base64 as encryption is a classic mistake and scanners flag it.

Authenticated encryption, GCM and ECB mode

Authenticated encryption scrambles data and detects tampering in one step; AES-GCM and ChaCha20-Poly1305 are the usual choices. ECB is an old mode in which identical inputs produce identical outputs, so it leaks patterns and provides no integrity. Scanners flag ECB.

Production code versus tests and examples

Production code is what runs for real users. Test fixtures, sample files and documentation often contain fake keys and old algorithms, which is not a real risk. Why it matters here: our reports separate the two, and only production code counts toward the score.

Misuse

Using a correct algorithm the wrong way, such as a hardcoded key, a repeated IV, or switching off certificate checking. Misuse is often more dangerous than an old algorithm, because it creates false confidence.

Part B: Every algorithm name you will see

A to C

AES

Advanced Encryption Standard. The world's default symmetric cipher, fast and correct. Quantum computers do not break it.

AES-128 and AES-256

The number is the key length in bits. AES-128 is still sound today, but Grover's algorithm halves the effective strength, so AES-256 is preferred for anything that must stay secret for a long time.

Argon2, bcrypt, scrypt and PBKDF2

Ways to turn a password into a key, deliberately slow to make guessing expensive. This is not a quantum problem, though the parameters still deserve review.

Blowfish

An older symmetric cipher, now dated. It usually appears in a list with DES, 3DES and RC4 as legacy.

BLAKE, BLAKE2 and BLAKE3

Modern hash families. They are fine, just less common than SHA-2 and SHA-3 in business systems.

Camellia, ARIA, Twofish, Serpent, CAST, SEED, IDEA and GOST

Strong but non-standard block ciphers. They are not broken by quantum computers, but they are flagged as "adequate" because using the mainstream choice, AES-GCM, is better for review and interoperability.

ChaCha20-Poly1305

A modern symmetric cipher with built-in tamper detection, very common on mobile. Quantum resistant, no change needed.

CRYSTALS-Kyber and CRYSTALS-Dilithium

The research names of the algorithms that became ML-KEM and ML-DSA. You will see them in older documents and in library names.

D to G

DES and 3DES

Old symmetric ciphers. DES is fully broken and 3DES is deprecated. Replace with AES-256-GCM or ChaCha20-Poly1305.

Diffie-Hellman, DH and DHE

The classic way two sides agree on a secret. Broken by Shor's algorithm, so it is replaced by ML-KEM or a hybrid.

DSA

An older US signature standard. Broken by Shor's algorithm and deprecated for ordinary reasons too.

ECDH

The elliptic curve version of Diffie-Hellman, smaller and faster. Same quantum problem.

ECDSA

Elliptic curve signatures, widely used in TLS certificates. Same quantum problem, replaced by ML-DSA or SLH-DSA.

Ed25519 and Ed448

Modern, fast signature algorithms. They look new, but they still rest on the same mathematics that Shor's algorithm breaks, so they must also be replaced. Why it matters here: Ed25519 is nevertheless the right interim choice for SSH today, while post-quantum SSH key types mature.

FIPS 203, 204 and 205

The three US standards, finalised in August 2024, that define the main post-quantum algorithms: ML-KEM, ML-DSA and SLH-DSA.

FN-DSA, also called FALCON

A fourth post-quantum signature family. Signatures are compact, but implementation is harder to get right.

H to M

HKDF

A modern way to derive keys from an existing secret. Fine for quantum purposes.

HMAC-SHA256

A tamper-detection tag built from SHA-256. Symmetric, so Shor's algorithm does not break it.

MD5

An old hash, broken today. Replace with SHA-256 or SHA-3.

ML-DSA, from CRYSTALS-Dilithium

The standardised post-quantum digital signature. It replaces RSA and ECDSA signatures.

ML-KEM, from CRYSTALS-Kyber

The standardised post-quantum key exchange. It replaces RSA and ECDH for agreeing on a key.

P to R

P-256, P-384, secp256r1 and secp256k1

Names of specific elliptic curves. All are quantum vulnerable. secp256k1 is the curve used by Bitcoin and most smart contracts, which is why on-chain cryptography is genuinely hard to migrate.

Post-quantum cryptography, or PQC

Cryptography designed to survive a quantum computer. The families are ML-KEM, ML-DSA, SLH-DSA and FN-DSA.

RC2 and RC4

Old ciphers, broken or deprecated. Replace them.

RIPEMD and Whirlpool

Older hashes with little margin. Prefer SHA-256 or SHA-3.

RSA

The workhorse public key algorithm since the 1970s, used for both encryption and signatures. Broken by Shor's algorithm at every key size.

RSA-2048, RSA-3072 and RSA-4096

RSA key sizes. Bigger is safer against today's computers, but no size survives a quantum computer, so the fix is replacement and not a bigger key.

RSA-OAEP and RSA-PSS

Correct, modern ways to use RSA. The padding is right, but the underlying RSA is still broken by Shor's algorithm, so these are flagged as needing replacement.

S to X

SHA-1

An old hash, broken today. Replace with SHA-256 or SHA-3.

SHA-2, meaning SHA-256, SHA-384 and SHA-512

The standard hash family. Still strong and quantum resistant.

SHA-3

The newer hash standard, built on a different design from SHA-2. Fine as it is.

SHA-512/256 and SHA-512/224

Shortened versions of SHA-512 with adequate margin. No change needed.

SLH-DSA, from SPHINCS+

A post-quantum signature based on hash functions rather than lattices. Slower, with much larger signatures, but it rests on a different mathematical bet, which is valuable for diversity.

SM2, SM3 and SM4

Chinese national algorithms. SM2 is public key and therefore quantum vulnerable. SM3 and SM4 are hashing and symmetric, so they have margin.

SSL, and TLS 1.0, 1.1, 1.2 and 1.3

Versions of the secure web protocol. SSL and TLS 1.0 and 1.1 are deprecated. TLS 1.2 is still common. TLS 1.3 is the modern version and it is required for hybrid post-quantum key exchange.

X25519

A modern, fast key agreement algorithm. Quantum vulnerable, so it is replaced by ML-KEM or paired with it into a hybrid.

X25519MLKEM768 and SecP256r1MLKEM768

Hybrid key exchange groups. The name is the classical algorithm joined to the post-quantum one. Why it matters here: seeing X25519MLKEM768 in a scan is good news, because it means post-quantum key exchange is already active.

The libraries a report will name

OpenSSL, including libssl and libcrypto

The C library behind most of the internet. libcrypto holds the algorithms, and libssl handles the connections. If it is bundled inside your application, its version decides which algorithms you can use.

BouncyCastle

A widely used Java cryptography provider. FIPS validated builds exist, which matters in regulated settings.

libsodium, NSS, GnuTLS, mbed TLS, wolfSSL and libgcrypt

Other common cryptography libraries. libsodium has modern defaults, NSS is Mozilla's, and the rest appear in specific platforms and products.

Apache Commons Crypto, node-forge, node-rsa, crypto-js, jose, jsonwebtoken, pyjwt, PyCryptodome, phpseclib, ring, ed25519-dalek and pqcrypto

Language level cryptography packages you will see named in a report. A report names them because a dependency often decides what an application can do, and because upgrading that dependency can be the fastest route to post-quantum support.

Part C: The quantum words

Quantum computer

A machine that uses quantum physics instead of ordinary bits. For a narrow set of mathematical problems it is dramatically faster than any ordinary computer, and outside those problems it is not.

Qubit

The unit of quantum information, like a bit but able to behave as 0 and 1 in combination until it is measured.

Superposition

A qubit's ability to hold a blend of 0 and 1 at once. It is the raw material, not the source of the speed.

Entanglement

A link between qubits so that their measured results are correlated in ways ordinary bits cannot be.

Interference

The trick that makes quantum algorithms work: the calculation is arranged so that wrong answers cancel out and the right answer is amplified.

Error correction and the logical qubit

Qubits are fragile, so many physical qubits are combined into one reliable logical qubit. The ratio today is hundreds or thousands to one, and it is the main reason nobody can give a dependable arrival date.

Shor's algorithm

The quantum algorithm that breaks RSA, Diffie-Hellman, ECDSA, Ed25519 and elliptic curve cryptography outright. This is the reason the migration exists.

Grover's algorithm

The quantum algorithm that speeds up searching, which halves the effective strength of symmetric ciphers. It does not break AES, but it is why AES-256 is preferred over AES-128 for long-lived data.

CRQC, a cryptographically relevant quantum computer

A quantum computer large enough to break real cryptography. The phrase exists because "quantum computer" alone is ambiguous: many exist and none is relevant yet.

Quantum-safe, quantum-resistant and post-quantum

Three ways of saying "built to survive a quantum computer". Post-quantum cryptography is the standard term for the new algorithms.

Hybrid key exchange

Using a classical and a post-quantum algorithm together in one handshake, so the connection is no weaker than today even if the new algorithm turns out to have a flaw. X25519MLKEM768 is an example.

Harvest now, decrypt later

Recording encrypted traffic today so it can be decrypted once a quantum computer exists. This is why the risk starts now rather than on the day the machine arrives.

Mosca's inequality

A rule of thumb named after Michele Mosca. If the years your data must stay secret plus the years migration will take is more than the years until a quantum computer exists, your exposure is already live.

Shelf life of data

How long information must stay confidential. A long shelf life plus harvest now equals exposure today, which is why medical and financial records are named as priority targets.

QKD, quantum key distribution

A different technology that uses quantum physics to distribute keys. It needs special hardware and links, and it is not what this product does.

Quantum entropy

Randomness generated from quantum processes, relevant to key generation. Out of scope for this product.

Part D: The scanning words

What a scanner is

Scanner and engine

The software that reads your code, files or endpoints and looks for cryptography. The engine is the shared set of detection rules, and the same engine runs in the cloud and in the offline runner.

Rule and pattern

A rule is a named check, for example "someone is using RSA in Java". A pattern is the exact text or shape being looked for. This engine has 84 rules covering 366 patterns across 22 language groups.

Regular expression, or regex

A compact way to describe a text pattern, used inside rules. You do not need to read one, but it explains how a scanner spots "RSA" in one language and "Aes.Create()" in another.

False positive and false negative

A false positive is a match that is not really a use, such as the word RSA inside a comment. A false negative is a real use the scanner missed, such as an algorithm name assembled at runtime. An honest tool states both risks rather than claiming perfection.

Coverage

How much of your material the scanner actually looked at. Ours reports entries in the archive, files considered, files scanned, bytes scanned and everything skipped, so you can judge the result instead of trusting it.

Evidence

The precise file, line and snippet behind a finding. Evidence is what separates a report you can act on from a claim you must take on faith.

Finding and occurrence

A finding is one place where cryptography was detected. An occurrence is one match. A single finding can occur many times.

Distinct algorithm

The number of different algorithms seen, counting each name once. This is the headline figure of 81 distinct algorithms.

Declared versus observed

Declared is what your source code says it depends on. Observed is what is actually present in the built file or the container image. The gap between them is often the most valuable finding, for example an OpenSSL version that arrived through a base image and appears in no repository at all.

Where the scanner looks

Source scan, or repository scan

Reading the code and configuration inside a Git repository. Public repositories can be scanned by the hosted service. Private ones need a linked account or the offline runner.

Repository, commit, branch and ref

A repository is a project's code with its full history. A commit is a saved snapshot. A branch is a line of work. Ref means whatever point in history was scanned, so a report can always state which version it saw.

Tarball and tar

A single compressed file holding a whole directory tree. Repositories are downloaded as a tarball and unpacked in memory for scanning.

Filtering and the ignore list

Skipping folders that never hold your own decisions, such as `node_modules`, `vendor`, `dist` and `build`. It keeps scans fast and reports honest, and the report lists what was skipped.

Artifact

The built output of an application, such as a JAR, WAR, package or binary. Scanning an artifact shows what actually shipped, which can differ from what the source suggests.

JAR and WAR

Java packages. A JAR is a library or application, and a WAR is a web application. Both are zip files, so a scanner can open them, list contents and even find archives nested inside archives.

Container image, OCI and Docker

A container image is a packaged filesystem holding an application and everything it needs. OCI is the standard format and Docker is the most common tool. Scanning an image exposes base operating system libraries and OpenSSL builds that no repository mentions.

Layer, base image and whiteout

An image is built from layers stacked on a base image. A whiteout is a marker that deletes a file from an earlier layer. A good scanner applies whiteouts so it does not report files that were deleted.

Registry pull and docker save

Two ways to obtain an image for scanning. Registry pull fetches it directly from a registry by tag, with no Docker needed. docker save exports an image from a machine that runs Docker into a tar file.

The endpoint scan

Endpoint scan

Scanning a public service over the network, such as api.company.com, to see what it negotiates. This is what the free domain check does.

Host, domain and port 443

A host is a machine name, a domain is the name you type, and port 443 is the standard secure web port. The endpoint scan connects to port 443.

ClientHello, ServerHello, HelloRetryRequest and key share

ClientHello is the opening offer from the client. ServerHello is the server's reply naming what it chose. HelloRetryRequest means the server preferred a group the client did not send a key for. A key share is the client's contribution to the key exchange. Why it matters here: these messages let our scanner learn the chosen algorithm without ever completing the handshake, which is why a result takes seconds.

Negotiated key exchange group

The algorithm the server actually agreed to use for key exchange in that handshake, such as x25519 or X25519MLKEM768. This is the single most useful fact in an endpoint scan.

Certificate fields: subject, issuer, SAN, validity and chain

Subject is who the certificate is for, issuer is the CA that signed it, SAN lists the names it covers, validity is the start and end dates, and the chain is the sequence linking it to a trusted root. A scanner reports all of it, because expiry and key type both force a decision.

DER and X.509

X.509 is the standard format of certificates and DER is the binary encoding they use. Reading DER is how a scanner turns a certificate into the fields above.

Vantage point

Where a scan was run from. A scan from our cloud sees a different world from a scan inside your network, which is why the report always states it.

DNS inference

When a handshake is impossible, the scanner can fall back to looking at DNS records and say something weaker. We label it clearly so it is never mistaken for a measurement.

The Cloudflare blind spot

Our hosted scanner runs on Cloudflare, and Cloudflare's own network will not let it reach Cloudflare's edge. For hosts behind Cloudflare, the offline runner is the answer.

Running it yourself

Runner, self-hosted and offline

The runner is a small tool you install that does the same scanning inside your own network. Self-hosted and offline mean the same idea: your code never leaves your machine.

Only findings leave

The runner writes a report locally and uploads nothing unless you ask it to. Even then it sends findings, never source code.

Activation key

A short code that links a downloaded runner to your account and unlocks repository and local scanning. It does not give anyone access to your source. Endpoint scans work without it.

Part E: What is inside a report

CBOM, a cryptographic bill of materials

A structured inventory of the cryptographic assets in a piece of software: algorithms, keys, certificates and where each is used. It is the core output of a scan and the thing you hand to a customer or auditor.

SBOM, a software bill of materials

A list of the software components inside something. A CBOM is the cryptographic cousin, and the two are often produced together.

CycloneDX

The open standard format for SBOMs and CBOMs that the industry already speaks. Because we accept CycloneDX from other tools, existing scanners you already run can feed into the same inventory.

Component and asset

A component is an entry in a bill of materials. In our report, an asset is a specific file that holds keys, certificates or credentials.

Readiness score, band and confidence

The score is a number from 0 to 100 summarising exposure. The band is the plain word that goes with it, such as Poor or Good. Confidence says how much evidence the number rests on. The full formula is published.

Insufficient evidence, and not measured

Sometimes the right answer is to refuse to score, for example when a repository holds almost no cryptography or a host does not answer. We say so rather than invent a flattering 100 or a damning zero.

The five risk categories

Every algorithm lands in exactly one. Quantum vulnerable means broken by Shor's algorithm and must be replaced. Deprecated or weak means broken today regardless of quantum. Adequate today means sound now but worth documenting. Quantum resistant means fine, such as AES-256. Post-quantum ready means the new algorithms, such as ML-KEM.

Prioritised actions

The short list of what to do first, with the recommended replacement for each algorithm. This is the part a busy reader actually uses.

Recommended replacement, or recommended action

The specific modern step to take, such as "replace ECDSA with ML-DSA" or "replace 3DES with AES-256-GCM".

Findings table

The full list of every place cryptography was found, with file, line, language and the exact snippet.

Exports: JSON, CSV and print to PDF

JSON is for machines and your own tooling, CSV opens in a spreadsheet, and printing saves a PDF you can attach or send.

Shareable link and retention

A shareable link is a permanent address for a report you can send to a customer or an auditor. Retention is how long it is kept, from 24 hours on the free plan to three years on the top plan.

Coverage limits and limitations

The honest statement of what was and was not seen: folders skipped, files over the size limit, and the fact that pattern matching misses algorithm names built at runtime and encrypted configuration.

Skipped, and truncated

Skipped counts files excluded by the rules, such as ignored folders, non-text files, oversized files and binaries. Truncated means the scan reached a size or time budget, so coverage is partial and the report says so.

Part F: The other tools we compare against

Syft

An open source tool that builds a software bill of materials from a repository, a container image or a filesystem.

Trivy

A widely used open source scanner for vulnerabilities and misconfiguration, which also produces SBOMs. It is excellent at CVEs, which we deliberately do not chase.

Grype

An open source vulnerability scanner, usually paired with Syft.

cdxgen

An open source tool that generates CycloneDX SBOMs and CBOMs across many languages.

CBOMkit

An open source toolkit for generating and analysing cryptographic bills of materials.

Semgrep and CodeQL

Static analysis tools that follow data through a program to find security bugs. Stronger than pattern matching for logic, and complementary to what we do.

gitleaks and trufflehog

Secret scanners that find committed passwords, keys and tokens, including buried in git history. They are better than us at that specific job.

testssl.sh, sslyze and SSL Labs

TLS configuration testers that grade a server's settings in depth. Use them alongside our endpoint scan rather than instead of it.

Part G: Plans, accounts and the runner

Free plan

The real product rather than a trial. Unlimited endpoint scans, unlimited public repository scans, the workbench, CycloneDX import, and one offline repository scan. Reports are free to view and kept for 24 hours.

Starter and Professional

The paid plans, at S\$149 and S\$499 per month. They add private repository scanning through a linked account, longer retention, permanent shareable links, the applications rollup and more runner seats. Professional adds human review of findings.

Entitlement

A single capability a plan turns on or off, such as "private repositories" or "shareable links". The pricing matrix is generated from these, so the page cannot claim something the software does not enforce.

Seat

One installed copy of the runner that a plan allows.

Linked account and OAuth

Connecting your GitHub, GitLab or Bitbucket account so we can read the private repositories you choose. OAuth is the standard, safe way to grant that access without sharing your password.

Managed scan versus offline scan

Managed means we fetch and scan your repository on our infrastructure. Offline means your machine does the work and only findings leave. Both produce the same report, and the report always says which was used.

Applications rollup

Grouping reports from several sources into one view, for example all twelve repositories and eight containers that make up the payments platform. It labels declared and observed inventory so the two stay distinguishable.

Bot protection via Turnstile, and rate limits

Turnstile is an invisible check that a visitor is human, which stops bots hammering the free scanners. Rate limits cap how many scans one visitor can run in a period, so the free tier stays available for real users.

Plan status and past due

Whether a paid plan is active. A failed payment drops the account to free entitlements rather than locking it out.

Part H: Singapore, the regulators and the Handbook

CSA, the Cyber Security Agency of Singapore

The agency that published the Quantum-Safe Migration Handbook and set the requirements for critical infrastructure owners.

GovTech and IMDA

The two other agencies that developed the Handbook and the Quantum Readiness Index with CSA. GovTech builds government technology and IMDA looks after the digital economy.

MAS, the Monetary Authority of Singapore

The financial regulator. Its 2024 circular asked every bank's chief executive to build crypto-agility and a cryptographic inventory.

CII, Critical Information Infrastructure

The systems whose failure would seriously disrupt Singapore, in sectors such as energy, water, banking, healthcare, transport and government. Their owners carry the three published deadlines.

Quantum-Safe Migration Handbook

CSA's guidance document, version 1 published on 16 July 2026. It is organised into five areas and lists no-regret moves for each.

Domain 1 to Domain 5, the Handbook areas

The Handbook groups the work into five areas. Domain 1 is risk assessment, finding your crown jewels and discovering the cryptography in them. Domain 2 is governance, naming owners, setting policy and tracking milestones. Domain 3 is technology, choosing replacement algorithms and staying crypto-agile. Domain 4 is training and capability, briefing leadership and building skills. Domain 5 is external engagement, asking your vendors and suppliers for their post-quantum roadmaps. Why it matters here: the coverage page says plainly what software can and cannot do in each domain, rather than implying that a scan can finish the job.

QRI, the Quantum Readiness Index

CSA's official self-assessment, a government form covering five areas across four maturity levels. We are not the QRI, we do not administer it and we do not reproduce it.

No-regret move

An action worth taking whichever way the quantum timeline turns out, such as finding out what cryptography you already use.

Crown jewels

The systems and data most critical to an organisation. The Handbook says to start discovery there rather than everywhere at once.

Crypto-agility

The ability to change an algorithm without rebuilding a system. It decides whether migration takes weeks or years, and both CSA and MAS treat it as central rather than optional.

Cryptographic inventory

A register of every cryptographic asset, recording the algorithm, key length, owner and the system it is used in. MAS asks for exactly this.

Migration plan and migration waves

The plan is the document a CII owner submits to CSA by 31 March 2027. Waves sequence the work: wave 1 for the highest risk and internet-facing systems, wave 2 for deprecated primitives folded into scheduled refreshes, wave 3 for lower exposure and third-party items.

Evidence versus compliance, and attestation

Evidence is technical proof of what you have and what you changed. Compliance also needs attestation, documentation and human review. No scanner can certify you, and any that claims to should be treated with suspicion.

Procurement questionnaire

The security and vendor questionnaire a bank or government buyer sends a supplier. This is how the deadlines reach companies that are not themselves CII owners.

PDPA

Singapore's Personal Data Protection Act, which governs how personal data must be protected. It matters here because the lifetime of that data is what drives quantum urgency.

FIPS, NIST and RFC 8446

NIST is the US standards body, FIPS are its published standards, with 203, 204 and 205 covering post-quantum algorithms, and RFC 8446 is the technical specification of TLS 1.3.

Part I: The workbench and migration words

Workbench

The planning area where scan findings become an actual migration plan. It produces documents you can edit and adopt.

Crown jewels register

Your list of the most critical systems and data, with owners. This is the starting point the Handbook recommends.

Exposure arithmetic

The calculation of how many years of confidentiality a system needs, how long migration will take, and how soon a quantum computer might exist. It turns "quantum" into a data question with an owner and a deadline.

Cryptographic inventory

Your working register of every asset, with algorithm, key length, owner and status. The scan fills it in, and you own and extend it.

Certificate register

A table of certificates with key algorithm, key size, signature algorithm, expiry and host. The bulk hostname scan builds one directly.

Migration milestones

The three CSA dates tracked as your own project dates, so progress is visible against them.

Crypto-agility checklist

A twelve point list of questions that reveal whether you could swap an algorithm quickly, such as whether algorithm choices are centralised and whether there is a test environment for cryptographic changes.

Cryptographic policy generator

Produces a first draft of an internal policy you can edit and adopt. A starting point, not legal advice.

Algorithm migration map

The per-algorithm plan: what you have, what it becomes, and any performance or interoperability question to settle first.

Vendor register and vendor request letter

A list of your suppliers with their post-quantum roadmap status, plus a letter containing the questions that actually change a timeline. Their answers set your schedule, which is why the Handbook calls this a no-regret move.

Leadership briefing outline

A short structure for briefing senior management, because they are the people who approve the resources.

The twelve words that matter most

If you keep only one page, keep this one.

Term	In one line
Public key cryptography	The kind a quantum computer breaks outright. RSA, ECDH, ECDSA, Ed25519.
Symmetric cryptography	The kind only weakened. AES and ChaCha20. Use AES-256.
Shor's algorithm	What breaks public key cryptography. The reason migration exists.
Grover's algorithm	Halves symmetric key strength. Why AES-256 over AES-128.
Harvest now, decrypt later	Traffic copied today and decrypted later. Why the risk starts now.
ML-KEM, ML-DSA, SLH-DSA	The three replacement algorithms, standardised in August 2024.
Hybrid key exchange	Classical plus post-quantum together. X25519MLKEM768 is an example.
Crypto-agility	How fast you could swap an algorithm. It decides your timeline.
CBOM	The cryptographic bill of materials. The inventory a scan produces.
Declared versus observed	What source says versus what actually shipped. The gap is the insight.
Coverage limits	What the scanner did not see. Stating it is what makes a report trustworthy.
Evidence, not compliance	We produce technical proof. Attestation and review stay with you.

Every explanation here matches the words used in the QuantumReady scanner, its reports and its website, so the vocabulary you read tonight is the vocabulary you will see tomorrow. Nothing in this document is legal or regulatory advice, and it does not reproduce the CSA Handbook or the Quantum Readiness Index. We are not affiliated with CSA, GovTech, IMDA or MAS.