



Cerebion Rivet — User Guide

Version 1.1.85 | Post-Quantum Security Analysis Platform

Table of Contents

1. Overview
 2. Installation
 3. License Activation
 4. Dashboard
 5. Code Analyzer
 6. Binary Analyzer
 7. Certificate Analyzer
 8. Network Analyzer
 9. Risk Scoring
 10. AI-Powered Fixes
 11. Reporting & Export
 12. CI/CD Integration
 13. Settings Reference
 14. Troubleshooting
 15. FAQ
-

1. Overview

Cerebion Rivet is a post-quantum security analysis platform that identifies cryptographic vulnerabilities in source code, compiled binaries, TLS certificates, and live network infrastructure. It assesses how exposed your systems are to quantum computing threats and provides actionable migration guidance.

The Quantum Threat

Quantum computers running Shor's algorithm will break RSA, ECC (all curves), and Diffie-Hellman key exchange. Grover's algorithm halves the effective security of symmetric algorithms, making AES-128 and SHA-256 significantly weaker. The current best estimates put a cryptographically relevant quantum computer 10–20 years away — but **harvest-now-decrypt-later attacks mean sensitive data encrypted today with RSA or ECC is already at risk.**

What Rivet Does

Analyzer	What it scans	Output
Code Analyzer	Source code (47+ languages)	Findings by severity, AI-generated fixes
Binary Analyzer	Compiled binaries (PE, ELF, Mach-O, .NET, Java)	Quantum Risk Score 0–100

Analyzer	What it scans	Output
Certificate Analyzer	TLS/SSL certificates and live HTTPS endpoints	Quantum Risk Score 0–100
Network Analyzer	Live hosts — open ports, TLS configuration	Quantum Risk Score 0–100

All Analysis is Local

All analysis runs entirely on your machine. No files, source code, or binaries are ever sent to Cerebion servers. Rivet is designed to work in air-gapped environments with zero network connectivity.

2. Best Practices Before You Scan

Important: Always work on a copy or a dedicated branch — never run scans or apply AI-generated fixes directly on production systems or your main repository branch.

- **Use a branch or backup.** Before applying any AI-generated fixes, create a Git branch or backup copy of your files. This lets you review, test, and roll back changes safely.
- **Review fixes before applying.** AI-suggested fixes are provided as guidance only. Always review and test them in a non-production environment before committing to production.
- **Non-production first.** Run analyses in a development or staging environment where possible, especially when using the AI Fix feature.

3. Installation

System Requirements

Platform	Requirement
Windows	Windows 10 or later (64-bit)
macOS	macOS 12 Monterey or later
Linux	Ubuntu 22.04+ recommended
RAM	4 GB minimum, 8 GB recommended
Disk	1 GB for application, additional space for scan history

Installing

1. Download the installer for your platform from cerebion.com/download or from your organization's internal distribution point.
2. Run the installer and follow the on-screen prompts.
3. Launch Cerebion Rivet from the Start Menu (Windows), Applications folder (macOS), or application launcher (Linux).

Linux note: If your distro is missing required runtime libraries, install them using your package manager before launch. On Ubuntu-based systems, the release notes may include a ready-to-run `apt-get` command. If the archive includes a sandbox helper, you may also need to apply the recommended permissions once during first setup.

Windows note: Windows may display a SmartScreen warning during install. Click **More info** → **Run anyway** to proceed. This is expected for newly released software.

On first launch, Rivet will prompt you to accept the End User License Agreement and activate your license. If you do not have a license yet, Rivet can still open in limited mode so you can start a free trial from the app.

4. License Activation

Online Activation

1. Open Cerebion Rivet and go to **Settings** → **License**.
2. Under the **Online License** tab, enter your license key from your trial signup or purchase confirmation email.
3. Click **Activate License** — Rivet validates against `license.cerebion.com` and saves the key automatically.

To move your license to a different machine, click **Detach License** on the current machine first, then activate on the new one. If you no longer have access to the old machine, contact `support@cerebion.com` to free the seat.

Headless / CI Activation

Set your license key as an environment variable — no GUI required:

```
# Linux / macOS
export RIVET_LICENSE_KEY=your-license-key-here
```

```
# Windows (cmd)
set RIVET_LICENSE_KEY=your-license-key-here
```

Rivet validates online on first run and caches the result for 7 days at `~/.rivet/license_cache.json`. Subsequent runs use the cache — no internet needed until it expires. Store the key in your CI/CD platform's secret store (GitHub Actions Secrets, GitLab CI Variables, Jenkins Credentials) — never hardcode it in pipeline files.

Offline / Air-Gapped Activation

Enterprise plan required. Offline licensing is not available on Trial or Professional tiers.

Offline licenses are cryptographically signed with Ed25519 and hardware-fingerprinted to a specific machine. Each machine requires its own license file.

Step 1 — Get your device fingerprint

In Rivet, go to **Settings** → **License** → **Online License** tab and click **Show Device Fingerprint**. Copy the fingerprint string or save it as `device-fingerprint.json`.

Email the fingerprint to `support@cerebion.com` with your license key or order reference. Support will send back a signed `license-[uuid].json` file.

The fingerprint contains only hardware identifiers (MAC address hash, architecture, hostname) — no personal data or file contents.

Step 2 — Upload the license file

1. Go to **Settings** → **License** → **Offline License** tab.
2. Drop the `license-[uuid].json` file onto the upload area or click to browse.
3. Click **Upload License File** — Rivet verifies the signature locally, no internet needed.

Deploying to headless machines:

Set the path to the license file as an environment variable:

```
# Linux / macOS
export OFFLINE_LICENSE_FILE=/path/to/license-[uuid].json

# Windows (cmd)
set OFFLINE_LICENSE_FILE=C:\path\to\license-[uuid].json
```

Common deployment approaches: - **Bake into machine image** — include the file in your AMI, Docker image, or VM template at a fixed path (e.g. `/etc/rivet/license.json`). One-time setup, survives reboots. - **Deploy via secrets manager** — store the file contents in AWS Secrets Manager, HashiCorp Vault, or similar. Add a provisioning script that writes the secret to disk before Rivet runs. - **Copy manually** — for a small number of fixed machines: `scp license.json user@host:/etc/rivet/license.json`

License Type	Trial Period	Air-Gap Support
Trial	14 days	No
Professional	—	No
Enterprise	—	Yes

Command-line scanner tools accept the same `--offline-license-file` flag (or `OFFLINE_LICENSE_FILE` environment variable) and share this machine's activation — once the desktop app or a CLI tool is activated, the others recognize it automatically.

5. Dashboard

The Dashboard provides a snapshot of your security posture across all analyzers.

Overview Panel

Metric	What it shows
Overall Risk Score	Weighted aggregate of all recent scan results
Critical Findings	Count of CRITICAL severity items requiring immediate action
High Risk Assets	Assets (domains, binaries, code repos) scoring above 60
PQC Ready	Percentage of scanned assets already using post-quantum algorithms

Recent Activity

The Recent Activity panel shows the last 10 scans across all analyzers, with timestamp, scan target, score, and status. Click any entry to open the full results.

Quantum Threat Timeline

A visual timeline showing estimated years until quantum computers can break each algorithm class:

- **RSA / ECC / DH** — 10–20 years (Shor's algorithm)
- **AES-128 / SHA-256** — effective security halved by Grover's (plan migration)
- **AES-256 / SHA-384+** — quantum-resistant (no action needed)

6. Code Analyzer

Static analysis for quantum-vulnerable cryptography in source code, combining a rule-based scanning engine with custom scanners for languages requiring specialized detection.

Supported Languages

Language	Extensions	Scanner
Python	.py, .pyw, .pyi	Built-in
JavaScript	.js, .jsx, .mjs, .cjs	Built-in
TypeScript	.ts, .tsx	Built-in
Java	.java	Built-in
C / C++	.c, .cpp, .cc, .cxx, .h, .hpp	Built-in
C#	.cs	Built-in
Go	.go	Built-in
Rust	.rs	Built-in
Ruby	.rb	Built-in
PHP	.php	Built-in
Kotlin	.kt, .kts	Built-in
Scala	.scala	Built-in
Swift	.swift	Built-in
Solidity	.sol	Built-in
COBOL	.cob, .cbl	Built-in
Elixir	.ex, .exs	Custom scanner
Apex (Salesforce)	.cls, .trigger	Custom scanner
Dart	.dart	Custom scanner
Lua	.lua	Custom scanner
Vue	.vue	Custom scanner
SQL	.sql	Custom scanner
Config / Data	.json, .yaml, .yml, .toml, .xml, .env	Built-in
Jsonnet	.jsonnet, .libsonnet	Built-in
Certificates	.pem, .crt, .cer	Custom scanner

What Gets Detected

Category	Examples
Asymmetric encryption / key exchange	RSA key generation, RSA-OAEP, ECDH, ECDHE, DH, DHE
Digital signatures	ECDSA, DSA, RSA signatures
Weak hash functions	MD5, SHA-1, SHA-224, SHA-256 (Grover risk)
Weak symmetric encryption	AES-128, DES, 3DES, RC4, Blowfish
Deprecated TLS	TLS 1.0, TLS 1.1, weak cipher suites, SSLv3
Hardcoded secrets	Private keys, API tokens, cryptographic material in source
Insecure random	Math.random(), rand() in cryptographic contexts
Key size issues	RSA < 4096, DH < 3072, ECC below P-384

Running a Scan

1. Click **Code Analysis** in the sidebar.
2. Click **Select Folder** and choose your project directory.
3. Configure file extensions and excluded directories in **Settings** → **Code Analysis** if needed.

4. Click **Scan**.

Findings stream to the UI in real time as batches complete. The progress bar shows files scanned vs total.

Cancellation: Click **Cancel** during a scan to stop it. Partial results are saved to history and marked as *Partial* — findings collected up to that point are preserved.

Understanding Results

Field	What it means
Severity	CRITICAL / HIGH / MEDIUM / LOW / INFO — set by the matching rule
Category	PQC badge shown when the finding involves a quantum-vulnerable algorithm
Line	Line number of the vulnerable code
Message	Description of the issue and why it is quantum-vulnerable, including the recommended migration
Actions	Fix generation buttons — see Section 10

Severity	Quantum meaning
CRITICAL / ERROR	Directly broken by Shor's algorithm (RSA, ECDSA, DH) — immediate migration required
HIGH / WARNING	Significant quantum risk — weak key sizes, AES-128, SHA-256 in critical contexts
MEDIUM	Deprecated but not immediately broken algorithms, weak TLS configurations
LOW / INFO	Best practice violations, lifecycle warnings

Scan History

Every scan is saved to history. Cancelled scans are stored as *Partial*. Clean files (no findings) are stored separately and lazy-loaded on demand.

7. Binary Analyzer

Scans compiled binaries for quantum-vulnerable cryptographic patterns without requiring source code. Combines YARA pattern matching, binary structure analysis, and disassembly.

Cross-platform analysis: Rivet can analyze binaries from any platform regardless of your OS. A Windows user can analyze Linux ELF binaries, macOS Mach-O libraries, or Docker image layers.

Supported File Formats

Format	Platform	Notes
PE (.exe, .dll, .sys)	Windows	Includes .NET assemblies
ELF	Linux	Shared libraries (.so) and executables
Mach-O (.dylib, .app, .framework)	macOS	Universal binaries supported

Format	Platform	Notes
Java .class / JAR	Cross-platform	JAR contents scanned individually
DEX	Android	Basic string extraction

Scan Options

Analysis Type:

Option	What it does
Crypto (default)	Cryptographic algorithm detection — imports, symbols, YARA patterns, API calls
Full	Crypto + hardcoded secrets, security flags, binary metadata

Analysis Depth:

Option	What it does
Quick	YARA pattern matching and import table scan only. Fastest — suitable for triage.
Standard (default)	YARA + full binary structure analysis + disassembly of code sections
Deep	All of the above plus extended disassembly and crypto constant detection

How Analysis Works

1. **YARA pattern matching** — scans raw binary bytes against quantum-specific YARA rules. Fast and format-agnostic. Runs on every scan.
2. **Binary structure parsing** — parses PE/ELF/Mach-O, extracts imported symbols, exported functions, and section metadata.
3. **Disassembly** — disassembles code sections to identify cryptographic operations at the instruction level.
4. **.NET metadata extraction** — for PE binaries, extracts managed code metadata to detect .NET crypto API usage.
5. **Risk scoring** — findings from all sources feed into the unified 4-component Quantum Risk Score.

Q-Score Reference

Each YARA rule match carries a Q-Score (1–5) indicating quantum severity:

Q-Score	Severity	Examples
5	CRITICAL	RSA (any key size), ECDSA, ECDH, DH, AES-128, MD5, SHA-1
4	HIGH	Kyber-512, Dilithium-2, FALCON-512, PRNG in quantum contexts
3	MEDIUM	RSA-1024/2048 key size constants, legacy crypto in quantum-aware systems

Q-Score	Severity	Examples
2	LOW	PQC performance-over-security modes, debug logging of quantum state
1	INFO	Migration recommendations, algorithm lifecycle warnings

Post-Quantum Migration Reference

Vulnerable Algorithm	Post-Quantum Alternative	NIST Standard
RSA (signatures)	ML-DSA (CRYSTALS-Dilithium)	FIPS 204
ECDSA	ML-DSA or SLH-DSA (SPHINCS+)	FIPS 204 / 205
RSA-OAEP / ECDH (encryption)	ML-KEM (CRYSTALS-Kyber)	FIPS 203
DH / DHE	ML-KEM (CRYSTALS-Kyber)	FIPS 203
AES-128	AES-256, ChaCha20-Poly1305	—
SHA-1 / MD5	SHA-384, SHA-512, SHAKE-256	—

Resource Limits

All configurable in **Settings** → **Binary Analysis**:

Limit	Default	Range
Max file size	500 MB	1–5000 MB
Analysis timeout	600 seconds	30–3600 s
Max memory	2048 MB	512–8192 MB
Max concurrent scans	3	1–10
Max archive size	500 MB	1–2000 MB
Max files per archive	100	1–1000

Stripped and Packed Binaries

- **Stripped binaries** — YARA pattern matching still works on raw bytes. Binary analysis may find fewer named crypto functions but can still detect crypto constants and instruction patterns. Confidence level will be *YARA Primary* or *Medium*. The score may be understated — a low score on a stripped binary should be interpreted with caution.
- **Packed binaries** — the outer packer layer may prevent crypto detection in packed sections.
- **Very small binaries** (< 100 KB) — may return UNKNOWN if no analyzable functions are found.

Library-Linked Algorithms

When a binary links against a crypto library (e.g. OpenSSL), the library’s symbols for all supported algorithms appear in the binary even if the application only calls a subset of them. Rivet cross-references YARA matches against confirmed algorithm usage detected by the binary parser. Matches that are present in the linked library but not confirmed as actually called by the application are shown in a separate **Library-Linked Algorithms** section rather than as active vulnerabilities. This prevents false-positive CRITICAL scores on binaries that link OpenSSL but only use AES.

Limitations

The Binary Analyzer performs **static analysis only** — it does not execute code. The following scenarios are outside its detection scope:

- **Runtime-only crypto** — algorithms loaded or assembled at runtime are not visible to static analysis.
- **Custom or proprietary implementations** — crypto that does not use known constants, OIDs, or library function names will not be detected.
- **Encrypted or heavily obfuscated payloads** — if the code section is encrypted and unpacked only at runtime, detection coverage is reduced.
- **Key size inference** — when a key size cannot be extracted, conservative worst-case assumptions are applied (e.g. RSA without a detected key size is treated as RSA-2048).
- **Unsupported formats** — file types other than those listed above return no findings, not a clean result.

A clean scan result means no known vulnerable cryptographic patterns were detected. It does not constitute a guarantee of quantum safety.

8. Certificate Analyzer

Analyzes SSL/TLS certificates and live HTTPS endpoints for quantum cryptography vulnerabilities.

Input Methods

- **Hostname scan** — connects to a live host on port 443 (or custom port) and retrieves the full certificate chain.
- **File upload** — analyze .pem, .crt, .cer, or .p7b files directly.
- **Bulk scan** — scan a list of hostnames from a CSV or newline-delimited text file.

Running a Scan

1. Click **Certificate Analysis** in the sidebar.
2. Enter a hostname (e.g. `example.com`) or upload a certificate file.
3. Click **Scan**.

For bulk scans, click **Bulk Scan**, upload your CSV/text file, and click **Start Bulk Scan**. Results for each host are saved individually to history.

Result Fields

Field	Range	What it means
Quantum Risk Score	0–100	Higher = more vulnerable
Security Grade	A–F / ?	A (< 20), B (20–39), C (40–59), D (60–79), F (80+). Shows ? when certificate details could not be retrieved.
Threat Level	minimal / low / medium / high / critical	Highest quantum threat within the certificate's remaining lifetime
Migration	monitor / plan / prepare / urgent / immediate	Recommended urgency
PQC Ready	Ready / Not Ready	Whether the certificate uses a NIST-approved PQC algorithm
Days to Expiry	integer / —	Days until notAfter. Short-lived certificates carry lower quantum risk. Shows — when the expiry date is unavailable.

Risk Score Components

Component	Weight	What drives it
Algorithm Risk	40%	Key algorithm and size vulnerability to Shor's algorithm
Timeline Risk	25%	Whether a quantum computer will exist before the certificate expires
Business Impact	20%	Criticality and compliance requirements
PQC Readiness	15%	Whether PQC algorithms are already in use

9. Network Analyzer

Scans live hosts for open ports and assesses the quantum security of detected TLS services.

Running a Scan

1. Click **Network Analysis** in the sidebar.
2. Enter a hostname or IP address.
3. Optionally configure port range and scan depth in Settings.
4. Click **Scan**.

Note: Port scanning without authorization is illegal in many jurisdictions. Only scan hosts you own or have explicit written permission to test.

What Gets Assessed

- Open ports and detected services
- TLS protocol version (TLS 1.0/1.1/1.2/1.3)
- Cipher suite quantum vulnerability
- Certificate chain quantum risk (same as Certificate Analyzer)
- Weak key exchange (DHE < 2048, ECDHE with weak curves)

Port Restrictions

System ports (below 1024) are restricted by default. Use **Settings** → **Network Analysis** to configure allowed port ranges.

10. Risk Scoring

All Binary, Certificate, and Network scans produce a unified **Quantum Risk Score from 0 to 100** using the same 4-component engine.

Score Ranges

Range	Level	Recommended Action
0–25	Low	Monitor; no immediate action needed
26–50	Medium	Plan migration within 3–5 years
51–75	High	Prioritize migration
76–100	Critical	Begin migration immediately

Score Components

Component	Weight	Description
Algorithm Risk	40%	Vulnerability of detected algorithms to Shor's/Grover's. RSA/ECC/DH score highest.
Timeline Risk	25%	Whether a quantum computer capable of breaking the detected algorithms is expected before the asset's replacement cycle.
Business Impact	20%	Business criticality and compliance requirements (FedRAMP, HIPAA, PCI, SOX).
PQC Readiness	15%	Whether post-quantum algorithms (ML-KEM, ML-DSA, SLH-DSA) are already present.

Algorithm Risk Reference

Algorithm	Risk Score	Notes
RSA-1024	95	Broken today by classical computers
RSA-2048	85	Broken by quantum; most common vulnerable RSA
RSA-4096	55	Longer key provides slight timeline mitigation
ECDSA P-256	85	Broken by Shor's algorithm
ECDSA P-521	45	Larger curve, longer timeline
DSA / DH	85	Same quantum risk as RSA-2048
AES-128	60	Grover halves effective key size to 64 bits
MD5 / SHA-1	90	Classically broken; quantum accelerates further
AES-256	0	Quantum-resistant
SHA-384 / SHA-512	0	Quantum-resistant

11. AI-Powered Fixes

The Code Analyzer can generate and apply fixes for detected vulnerabilities using an LLM.

Requirements

- An LLM API key configured in **Settings** → **AI Configuration** (cloud provider by default; Enterprise air-gapped licenses can point to a custom or on-premise LLM endpoint instead)
- The file must be inside a Git repository for Apply and Rollback to work

Fix Modes

Two modes are available, selectable in **Settings** → **AI Configuration**:

Mode	Button	What it generates
AI Recommendation (default)	Generate AI Recommendation	Inserts a single-line <code>@TODO-pq</code> comment above the vulnerable code describing the issue and the recommended PQC migration. Does not modify existing code.

Mode	Button	What it generates
AI Code Fix	Generate AI Code Fix	Replaces the vulnerable code with a working PQC-safe implementation and adds a <code>@TODO-pq REPLACED:</code> comment. Review carefully — the LLM works from ~30 lines of context.

If AI is disabled or no LLM key is configured, a **Generate Recommendation** button appears instead. This uses the rule autofix field where available, or a rule-based recommender — no LLM required.

Applying a Fix

1. Click the generate button on any finding — a loading indicator appears while the fix is generated.
2. Click **View AI Recommendation** or **View AI Code Fix** to review the diff.
3. Click **Apply Fix** — Rivet applies the patch as an atomic Git commit.
4. If the fix causes issues, click **Rollback** — the commit is reverted.

Safety Notes

- Fixes are applied via Git patch operations — no changes are made without a reviewable diff.
- Apply and Rollback are atomic — either the full patch applies or nothing changes.
- Rivet detects staleness: if the file has changed since the fix was generated, it prompts you to re-generate before applying.

12. Reporting & Export

Exporting Individual Scan Results

From any scan result view, click **Export** to download results in your preferred format:

Format	Contents	Best for
CSV	Findings table with severity, algorithm, file location, and recommendations	Spreadsheet analysis, ticketing system import
JSON	Full structured output including all metadata and raw findings	Programmatic processing, custom tooling
SARIF	Static Analysis Results Interchange Format — standard findings format	GitHub Advanced Security, Azure DevOps, IDEs

AI Executive Report

From the **Reports** page, select one or more scans, add optional context, and click **Generate Report** to produce an AI-powered executive summary. Generated reports are saved to the Reports history and can be downloaded at any time.

Results are based on static pattern analysis. A clean result does not constitute a guarantee of quantum safety.

Bulk Export

From the **Reports** page, click **Export CSV** or **Export JSON** to download all scan records in one file. Records are sorted by date, newest first (up to 500 most recent).

Code Analysis Export

From the Code Analysis results view, the **Export** button offers CSV, JSON, and SARIF formats for the current scan's findings.

13. CI/CD Integration

Rivet can be integrated into CI/CD pipelines for automated quantum security scanning on every build.

GitHub Actions

```
- name: Quantum Security Scan
  uses: cerebion/rivet-action@v1
  with:
    license_key: ${ secrets.RIVET_LICENSE_KEY }
    scan_path: ./src
    fail_on: high    # fail build if any HIGH or CRITICAL findings
```

GitLab CI

```
quantum-scan:
  image: cerebion/rivet-ci:latest
  script:
    - rivet scan --path ./src --format json --output rivet-report.json
  artifacts:
    reports:
      quantum: rivet-report.json
  variables:
    RIVET_LICENSE_KEY: $RIVET_LICENSE_KEY
```

Jenkins

```
stage('Quantum Security') {
  steps {
    sh 'rivet scan --path ./src --format json --output rivet-report.json'
    archiveArtifacts artifacts: 'rivet-report.json'
  }
  environment {
    RIVET_LICENSE_KEY = credentials('rivet-license-key')
  }
}
```

Exit Codes

Code	Meaning
0	Scan completed, no findings above threshold
1	Scan completed, findings above threshold detected
2	Scan failed (configuration error, license error, timeout)

14. Settings Reference

License

Setting	Description
License Key	Your Cerebion Rivet license key
Device Fingerprint	Hardware fingerprint for offline license generation
Offline License File	Path to air-gapped license file

Code Analysis

Setting	Description
File Extensions	Which file types to include in scans
Excluded Directories	Directories to skip (default: node_modules, .git, venv, dist, build)

Binary Analysis

Setting	Default	Description
Max file size	500 MB	Maximum binary size to accept
Analysis timeout	600 s	Per-scan time limit
Max memory	2048 MB	Memory limit per scan
Max concurrent scans	3	Parallel scan limit

AI Configuration

Setting	Description
LLM Provider	Cloud (default) or custom/on-premise endpoint (Enterprise air-gapped)
API Key	Your AI provider API key
Model	Model to use for fix generation

Network Analysis

Setting	Description
Default port range	Ports to scan when no range is specified
Scan timeout	Per-host timeout in seconds
Allowed ports	Restrict scanning to specific ports or ranges

Application

Setting	Default	Description
Log Level	Info	Controls verbosity of application logs written to <code>app.log</code> . Debug — detailed diagnostics; Info — normal operation (default); Warning — warnings and errors only; Error — errors only. Changes apply immediately and persist across restarts.

15. Troubleshooting

Application Won't Start

- Verify your system meets the minimum requirements (64-bit OS, 4 GB RAM).
- On macOS: if blocked by Gatekeeper, go to **System Preferences** → **Security & Privacy** and click **Open Anyway**.
- On Linux: after extracting the archive, ensure the launcher is executable: `chmod +x CerebionRivet`.
- Check the application log at:
 - Windows: `%APPDATA%\CerebionRivet\logs\app.log`
 - macOS: `~/Library/Logs/CerebionRivet/logs/app.log`
 - Linux: `~/.config/CerebionRivet/logs/app.log`

License Activation Fails

- Verify you have internet access and that `license.cerebion.com` is reachable.
- Check that your license key was entered correctly (no extra spaces).
- If your license is already active on another machine, click **Detach License** there first, or contact `support@cerebion.com`.

Code Scan Returns No Results

- Verify the selected folder contains files with supported extensions.
- Check **Settings** → **Code Analysis** → **File Extensions** — ensure the relevant languages are enabled.
- Try scanning a single file first to confirm the scanner is working.
- Ensure the bundled rules are present in the installation directory. On a clean install, rules are included automatically. If rules appear missing, reinstall the application.

Binary Scan Returns No Findings

- Confirm the file is a supported format (PE, ELF, Mach-O, Java .class/JAR).
- Increase **Analysis Depth** to **Deep** for a more thorough scan.
- Stripped or heavily packed binaries may return fewer findings — see the Stripped and Packed Binaries section.
- Check file size is within limits (**Settings** → **Binary Analysis** → **Max file size**).

Certificate Scan Fails

- Verify the host is reachable and has a valid TLS certificate on the specified port.
- For certificate file uploads, ensure the file is in PEM, DER, or P7B format.
- Check that port 443 (or your specified port) is not blocked by a firewall.

AI Fix Generation Fails

- Verify your Google API key is configured in **Settings** → **AI Configuration**.
- Ensure the file is inside a Git repository (required for Apply and Rollback).
- If the fix was generated for an older version of the file, click **Re-validate** to regenerate with the current content.

Log Locations

Platform	Log path
Windows	%APPDATA%\CerebionRivet\logs\main.log
macOS	~/Library/Logs/CerebionRivet/main.log
Linux	~/.config/CerebionRivet/logs/main.log
Backend (Windows)	%APPDATA%\CerebionRivet\logs\app.log
Backend (macOS)	~/Library/Logs/CerebionRivet/logs/app.log
Backend (Linux)	~/.config/CerebionRivet/logs/app.log
Backend errors	Same directory, <code>error.log</code>

16. FAQ

When will quantum computers break RSA? Current estimates range from 10–20 years for cryptographically relevant quantum computers. However, harvest-now-decrypt-later attacks mean sensitive data encrypted today is already at risk.

Does Rivet upload my files to the cloud? No. All analysis runs locally on your machine. No files, binaries, or source code are ever sent to Cerebion servers.

Does a clean Binary Analyzer result mean my binary is quantum-safe? No. A clean result means no known vulnerable cryptographic patterns were detected using static analysis. Rivet cannot detect runtime-only crypto, custom implementations that do not match known patterns, or algorithms hidden inside encrypted/packed binary sections.

What is the difference between Desktop and Server editions? Desktop is a single-seat license for individual use. Server supports unlimited seats and includes CI/CD integrations, air-gap support, and priority support.

Can I use Rivet in an air-gapped environment? Yes. Rivet supports fully offline operation with Ed25519-signed offline license files. No internet connection is required after activation. All analysis rules are bundled with the application.

What NIST PQC algorithms does Rivet recommend? ML-KEM (CRYSTALS-Kyber) for key encapsulation — NIST FIPS 203. ML-DSA (CRYSTALS-Dilithium) for digital signatures — NIST FIPS 204. SLH-DSA (SPHINCS+) as a stateless hash-based signature alternative — NIST FIPS 205.

How do I move my license to a new machine? Go to **Settings** → **License** → **Deactivate License** on the old machine. Then activate on the new machine with the same key. If you no longer have access to the old machine, contact support@cerebion.com.

How do I cancel my subscription? Cancel anytime from your account at cerebion.com/portal or by contacting support. Your license remains active until the end of the billing period.

Contact & Support

- **Documentation:** cerebion.com/docs
- **Support:** support@cerebion.com
- **License portal:** cerebion.com/portal
- **Security disclosures:** security@cerebion.com

Cerebion Rivet is © Cerebion. All rights reserved. See cerebion.com/terms for license terms.