

08 Cybersecurity

*As the EU Cyber Resilience Act (CRA) *has already entered into force in 2024, and the first legal requirements must be fulfilled starting in September 2026, we are now explicitly collecting tips, tricks, and best practices related to the topic here.*

Cybersecurity is part of software quality and maintainability. Secure systems should be deterministic, understandable, and explicit in their behavior.

General Principles

- Treat all external input as untrusted.
- Prefer simple and explicit architectures.
- Fail safely and log security-relevant failures.
- Minimize attack surface and unnecessary dependencies.
- Apply the principle of least privilege.

Secrets and Credentials

- Never hardcode:
 - passwords,
 - API keys,
 - certificates,
 - tokens,
 - database credentials.
- 2. Do not store secrets in:
 - block diagram constants,
 - typedef defaults,
 - test VIs,
 - screenshots,
 - source-controlled config files.
- 3. Use external configuration or secure credential storage.
- 4. Never log sensitive information.

Networking

- Prefer encrypted communication:
 - TLS,
 - HTTPS,
 - SSH,
 - SFTP,
 - OPC UA security.
- 2. Avoid plaintext credentials and insecure protocols.
- 3. Validate certificates and remote peers.
- 4. Explicitly document:
 - timeout behavior,

- reconnect strategy,
- retry handling.

LabVIEW-Specific Guidelines

VI Server

- Disable VI Server if not required.
- Restrict network access and permissions.
- Never expose unrestricted VI Server access on production systems.

Dynamic VI Loading

- Only load trusted plugins or VIs.
- Avoid loading VIs from user-writable directories.
- Use strict connector pane contracts and versioning.

System Exec

- Treat all command line input as untrusted.
- Avoid constructing shell commands from unchecked user input.
- Document all external tool dependencies.

File Handling

- Validate file paths and filenames.
- Avoid path traversal vulnerabilities.
- Restrict writable and executable locations.

DQMH and Modular Architectures

- Encapsulate queues, references, and communication resources.
- Avoid exposing internal module resources.
- Validate message payloads and typedef compatibility.
- Keep security-related blocking operations outside the EHL.

Logging and Error Handling

- Log security-relevant events:
 - failed authentication,
 - rejected certificates,
 - malformed packets,
 - permission violations.
- 2. Logs should include:
 - timestamps,
 - module names,
 - severity,
 - connection information.
- 3. Avoid excessive logging in real-time loops.

Deployment

- Remove development tooling from production systems.
- Avoid unnecessary administrator privileges.
- Keep dependencies updated.
- Prefer signed installers and executables.

Recommended

- Encapsulated modules
- Explicit APIs
- Typed message payloads
- Centralized configuration
- Deterministic error handling
- Sanitized logging
- Secure communication protocols

Avoid

- Hardcoded credentials
- Global mutable state
- Disabled certificate validation
- Unvalidated external input
- Arbitrary dynamic VI loading
- Logging secrets
- Silent failure handling

From:
<https://dokuwiki.hampel-soft.com/> - **HAMPEL SOFTWARE ENGINEERING**

Permanent link:
<https://dokuwiki.hampel-soft.com/kb/bestpractices/codingconventions/cybersecurity?rev=1778146301>

Last update: **2026/05/07 09:31**

