CodeSprinter
Innovating today, shaping tomorrow

Vulnerability Management Policy

Version 1.0 · 2026

www.codesprinter.net

DOCUMENT

Type · Policy document
Version · 1.0 (2026)
Prepared by · CodeSprinter

CODESPRINTER

Kwikstaartlaan 42
3704 GS Zeist
The Netherlands

CONTACT

info@codesprinter.nl
www.codesprinter.net
Emergency number: 085 - 369 - 7644

COMPANY DETAILS

Chamber of Commerce (KVK) 94394768
VAT NL005081056B55

CHAPTER 1

Introduction and scope

Purpose of this policy

This Vulnerability Management Policy describes how CodeSprinter deals with security vulnerabilities in the systems, applications and software components that it manages or develops for clients. The policy sets out how vulnerabilities are identified, assessed, prioritised and remedied, and how third parties can report a vulnerability they have found in a responsible manner.

A vulnerability is a weakness in software, configuration or design that an attacker can exploit to gain unauthorised access, manipulate data or disrupt the operation of a system. Timely identification and control of vulnerabilities is an essential part of CodeSprinter's security approach.

Scope

This policy applies to:

- operating systems and server configurations that CodeSprinter manages on behalf of clients;
- web applications and software solutions that CodeSprinter develops or manages;
- software dependencies (third-party libraries, frameworks, packages) that form part of the managed applications;
- the server hardware, the operating system and the infrastructure components that CodeSprinter manages itself, as well as the services of engaged suppliers.

This policy does not apply to systems or software that fall entirely outside CodeSprinter's management scope, unless additional arrangements have been made in this respect in the Software Maintenance Agreement or the Managed Hosting SLA.

Relationship with other documents

This policy should be read in conjunction with the following documents from the CodeSprinter Document Suite:

- **Technical and Organisational Measures (TOM):** describes the broader security measures of which vulnerability management forms part (see chapter 8 of the TOM).
- **Software Maintenance Agreement:** contains the contractual arrangements for carrying out security updates and patch management.

- **Managed Hosting SLA:** governs the availability commitments and maintenance windows within which patches can be applied.
- **Security & Privacy Statement:** describes CodeSprinter's broader security and privacy approach at a public level.
- **Incident and Data Breach Procedure:** applies as soon as a vulnerability has been exploited and a security incident or data breach has occurred.

CHAPTER 2

Identification of vulnerabilities

Automated dependency scans

CodeSprinter uses automated tools to detect vulnerabilities in software dependencies (libraries, packages and frameworks). The scans deployed include Dependabot, which continuously checks the repositories for vulnerable dependencies, supplemented by npm audit as part of the CI pipeline. These scans are run:

- automatically on every new build or deploy, insofar as the tooling used supports this;
- manually when updating a dependency or integrating a new package;
- periodically on the existing codebase, at a minimum frequency of once a month.

The results of the scans are assessed for relevance and severity and, where necessary, converted into a remediation action.

Security notices from suppliers and projects

CodeSprinter actively follows the security notices of suppliers and open-source projects on which the managed applications depend. This includes:

- security advisories from the suppliers of the operating systems, databases and runtime environments used;
- notices via the public vulnerability databases, including the National Vulnerability Database (NVD) and the Common Vulnerabilities and Exposures (CVE) list;
- security notices via the official channels of the open-source frameworks and libraries used;
- relevant bulletins from the National Cyber Security Centre (NCSC) of the Dutch government.

Monitoring updates and patch notices

For every system and every software component that CodeSprinter manages, CodeSprinter keeps track of which version is in use and whether newer, more secure versions are available. Depending on the environment, the following methods are used for this purpose:

- automatic installation of, and notifications about, operating system security updates via unattended-upgrades;
- periodic manual review of the release notes and changelogs of critical components;

- notifications via subscribed security mailing lists or RSS feeds from relevant suppliers.

Own security testing

In addition to passive monitoring, CodeSprinter periodically carries out active checks on managed systems and applications, including:

- periodic vulnerability scans at server level, at a minimum frequency of once a month;
- manual review of security-critical configurations (access, TLS, firewall rules) upon material changes to the environment;
- review of application code for security risks as part of the development process, in accordance with the Secure Development Lifecycle.

CHAPTER 3

Assessment and classification by severity

Classification method

Every identified vulnerability is assessed for severity before a remediation action is determined. CodeSprinter applies a classification based on the Common Vulnerability Scoring System (CVSS), supplemented by a contextual assessment that takes into account:

- the actual exposure of the affected system or component (internally versus externally reachable);
- the existence of a known, publicly available exploit;
- the sensitivity of the data that the affected component processes or has access to;
- the likelihood of exploitation given the current threat landscape.

Classification levels

VULNERABILITY SEVERITY LEVELS

CodeSprinter applies four classification levels. Where there is doubt about the correct classification, the higher class is applied until further investigation provides certainty.

Critical	A vulnerability with a CVSS score of 9.0 or higher, or a vulnerability with a lower score that is designated as critical by the contextual assessment. An immediate, realistic threat of exploitation with serious consequences: full compromise of a system, unauthorised access to personal data or services, or complete failure of a production environment.
High	A vulnerability with a CVSS score of 7.0 to 8.9, or a vulnerability with a lower score that is designated as high by the contextual assessment. A considerable risk of exploitation with a significant impact on the confidentiality, integrity or availability of systems or data.
Medium	A vulnerability with a CVSS score of 4.0 to 6.9. A limited or indirect risk; exploitation generally requires additional conditions, such as pre-existing (limited) access, or the impact is limited in scale.
Low	A vulnerability with a CVSS score of 0.1 to 3.9. A minimal direct threat; possibly informational in nature, or exploitation is merely theoretical. Remediation is included in regular maintenance.

Contextual assessment and adjustment

A CVSS base score is a generic measure that does not take the specific deployment environment into account. CodeSprinter adjusts the classification on the basis of the contextual factors described above. A vulnerability with a medium score may be upgraded to high through the contextual assessment when:

- the affected component processes personal data of the client;
- the vulnerability is present in an externally reachable part of the application;
- a publicly known exploit is available.

Adjusted classifications are documented in the vulnerability register (see chapter 7), including the reasons for the adjustment.

CHAPTER 4

Remediation timeframes by severity

Principles

CodeSprinter sets concrete timeframes for remedying vulnerabilities. These timeframes are realistically calibrated: they take into account the available capacity, the need to test before deploying to production, and the avoidance of ill-considered rushed patches that introduce new problems.

The timeframes start at the moment a vulnerability has been identified and classified, not at the moment of first discovery via an external source.

Remediation timeframes by severity level

- **Critical:** remediation or an adequate mitigating measure within 48 hours of classification. If a definitive fix is technically not achievable within this timeframe (for example because the supplier has not yet released a patch), CodeSprinter implements a temporary mitigating measure as quickly as possible (such as disabling or isolating the affected component) and CodeSprinter informs the client accordingly.
- **High:** remediation within 7 calendar days of classification.
- **Medium:** remediation within 30 calendar days of classification, or included in the next scheduled maintenance window that falls within this timeframe.
- **Low:** remediation within 90 calendar days of classification, or included in a regular maintenance round.

Deviation from timeframes

In the following situations the remediation deadline cannot be met:

- the supplier of the affected component has not yet released a security patch;
- applying the patch requires extensive testing or a scheduled maintenance period in order to avoid unwanted downtime;
- the vulnerability is located in a component that falls technically or contractually outside CodeSprinter's management scope.

In all these cases, CodeSprinter documents the reason for the delay and the mitigating measures taken in the vulnerability register. For critical and high vulnerabilities, CodeSprinter actively informs the client about the delay and the temporary measures, without waiting for a request from the client.

Availability of the client

For some patches (in particular at application level or for updates that may affect functions or interface elements), the cooperation or approval of the client is required. In those cases CodeSprinter informs the client as quickly as possible and provides a clear explanation of the urgency. If the client delays or refuses a necessary patch, the client bears the consequences of exceeding the timeframe (see also chapter 8).

CHAPTER 5

Patching and updating

Test procedure for patches

Unless the situation requires acute, immediate rollout (in the case of critical vulnerabilities without an available mitigation), security updates are first installed in a test or acceptance environment. This verifies that:

- the patch works correctly and demonstrably remedies the vulnerability;
- the patch does not introduce any unforeseen breakage of functionality in the application;
- the patch is compatible with the other components of the environment.

The test procedure is described in the Software Maintenance Agreement. For critical vulnerabilities where immediate deployment to production is necessary, the test is carried out as concisely yet as thoroughly as possible, followed by enhanced monitoring after rollout.

Maintenance windows

Patches are preferably rolled out within the agreed maintenance windows as laid down in the Managed Hosting SLA or the Software Maintenance Agreement. For critical and high vulnerabilities, CodeSprinter may, after consultation with the client or in the event of an immediate threat, deviate from the regular maintenance window and roll out an urgent patch outside the scheduled time slots.

End-of-life components

Software, libraries and operating systems that no longer receive security updates (end-of-life or end-of-support) are not deployed in production environments that process personal data. When a component reaches end-of-life status, CodeSprinter advises the client in good time about the need for replacement or upgrade. The planning and costs of such a replacement fall outside regular patch management and are treated as additional work in accordance with the Software Maintenance Agreement.

Documentation and feedback

After a security update or patch has been carried out, the following details are recorded in the vulnerability register (see chapter 7):

- the identified vulnerability (including the CVE reference where available);
- the affected component and the version number before and after the patch;
- the date of rollout in the test and production environment;
- the name of the person who carried out the patch on behalf of CodeSprinter;
- any particulars or deviations from the standard procedure.

On request, the client receives an overview of the security updates carried out over the past period.

CHAPTER 6

Coordinated disclosure of vulnerabilities by third parties

Coordinated Vulnerability Disclosure

CodeSprinter values the efforts of security researchers, ethical hackers and other third parties who discover vulnerabilities in the systems and applications of CodeSprinter or its clients and report them in a responsible manner. For this purpose CodeSprinter operates a Coordinated Vulnerability Disclosure (CVD) scheme, also known as responsible disclosure.

The purpose of this scheme is to ensure that vulnerabilities are remedied as quickly as possible without being made public prematurely in a way that facilitates exploitation.

Reporting point and how to contact us

Vulnerabilities can be reported via:

- **E-mail:** security@codesprinter.net (state "CVD" or "Responsible Disclosure" in the subject line)
- **Encryption:** if the reporter wishes to use PGP encryption for the report, the public key block can be requested via the same e-mail address.

CodeSprinter confirms receipt of a report within 3 business days and aims to inform the reporter of the initial assessment of the report within 10 business days.

What constitutes a responsible report

A report is considered responsible if the reporter:

- has investigated the vulnerability only to the extent necessary to demonstrate its existence;
- has not exploited the vulnerability, such as by viewing, copying, modifying or deleting data, or by disrupting the service;
- has not shared the vulnerability with third parties until CodeSprinter has had the opportunity to remedy the problem;
- has not used social engineering, phishing, brute-force attacks or other attack methods that are not strictly necessary to demonstrate the vulnerability;
- has made the report as soon as possible after discovering the vulnerability;

- submits the report in Dutch or English via the reporting point named in this chapter.

What is not permitted

The following actions fall outside the scope of the responsible disclosure scheme and may give rise to legal action, regardless of the reporter's intentions:

- installing malware, backdoors or other malicious software;
- modifying or destroying data;
- forwarding, publishing or otherwise using data of clients or data subjects that has been encountered;
- carrying out (D)DoS attacks or comparable disruptions;
- manipulating employees or contact persons through social engineering;
- reporting vulnerabilities in third-party systems that do not form part of the environment managed by CodeSprinter.

CodeSprinter's commitment upon a responsible report

If a reporter complies with the conditions described above, CodeSprinter commits to the following:

- CodeSprinter will not take legal action on the basis of the mere discovery and responsible reporting of the vulnerability.
- CodeSprinter treats the report confidentially and does not share the reporter's personal data with third parties without their consent, unless CodeSprinter is legally obliged to do so.
- CodeSprinter aims to remedy the vulnerability in accordance with the remediation timeframes in chapter 4 and keeps the reporter informed of progress.
- If the reporter so wishes and CodeSprinter agrees, the reporter may be publicly acknowledged as the person who made the report (responsible disclosure credit) once the vulnerability has been remedied, unless anonymity is preferred.

CodeSprinter does not offer financial rewards (bug bounties) for vulnerability reports, unless expressly agreed otherwise in writing.

Coordination for vulnerabilities in client systems

When a report concerns an application that CodeSprinter manages on behalf of a client, CodeSprinter informs the client concerned as soon as possible about the nature of the vulnerability and the intended remediation action. Coordination with the reporter takes place through CodeSprinter as the point of contact, unless the client expressly requests to take over the communication itself.

CHAPTER 7

Registration

Vulnerability register

CodeSprinter maintains an internal vulnerability register of all identified vulnerabilities assessed as medium or higher, as well as of responsible reports made by third parties. The register serves as demonstrable evidence of the vulnerability management carried out and as an accountability instrument towards clients and, where applicable, supervisory authorities.

For each vulnerability, the register records at least the following details:

- **Identification:** unique registration ID, date of identification and source (automated scan, supplier bulletin, own test or report by a third party).
- **Description:** description of the vulnerability, the affected component(s) and version(s), and the CVE reference or equivalent reference where available.
- **Classification:** severity level (critical/high/medium/low), CVSS score where available, and the reasons for any adjusted classification.
- **Status:** current remediation status (open, in progress, mitigated, closed).
- **Remediation action:** description of the measure taken or planned, the date of rollout in test and production, and the person who carried it out.
- **Client:** where applicable, the client whose system the vulnerability concerns.
- **Notification to client:** date and manner in which the client was informed, where applicable.

Retention of the register

The vulnerability register is retained for a period of at least 3 years after the relevant entry is closed. The register is available for inspection by the client on request, insofar as the entries relate to the client's systems. On request, CodeSprinter provides an overview of the vulnerabilities handled over a specified period.

Relationship with the incident register

If a vulnerability is exploited and leads to a security incident or data breach, an entry is also recorded in the incident register in accordance with the Incident and Data Breach Procedure, in addition to the entry in the vulnerability register. Both entries refer to each other via the registration ID.

CHAPTER 8

Client responsibilities

Cooperation in patch management

Effective vulnerability management requires the cooperation of the client. The client is responsible for:

- granting permission in good time for the execution of security updates that may affect the availability or behaviour of the application;
- making available a test environment or acceptance environment in which CodeSprinter can validate patches before deploying to production, where the nature of the application requires this;
- responding in good time to urgent requests for information from CodeSprinter in the case of critical or high vulnerabilities;
- being available for consultation outside regular office hours in the case of acute security incidents arising from a serious vulnerability.

Security outside the management scope

The client is itself responsible for the security of systems, applications and data that fall outside CodeSprinter's management scope, such as:

- systems, accounts or third-party integrations managed by the client itself;
- passwords and access credentials that the client manages for access to the application or the administration panel;
- changes that the client, or a third party engaged by the client, has made to the application without involving CodeSprinter.

Vulnerabilities that have arisen as a result of actions by the client or engaged third parties outside CodeSprinter's management scope are not covered by the remediation timeframes of this policy and, if remediation is desired, are treated as additional work in accordance with the Software Maintenance Agreement.

Client's duty to inform

The client is obliged to inform CodeSprinter without delay as soon as it becomes aware of a vulnerability, security incident or suspicious activity relating to the systems or applications managed

by CodeSprinter. Early notification by the client can considerably increase the speed of remediation.

Instructions for secure use

The client is responsible for the secure use of the application by its own employees and users, including the use of strong passwords, the use of multi-factor authentication where offered, and the timely reporting of suspicious activity to CodeSprinter.

CHAPTER 9

Review

Periodic review of the policy

CodeSprinter periodically reviews this Vulnerability Management Policy for currency, completeness and effectiveness. The review takes place at least once a year and in any event after:

- a security incident in which an unpatched or late-patched vulnerability played a role;
- a material change in the technology stack used, the deployment environment or the management scope;
- a significant change in the threat landscape or the state of the art in the field of vulnerability management;
- a relevant change in legislation or regulations affecting the requirements for vulnerability management.

Lessons learned from incidents and reports

After handling a critical or high vulnerability, and after processing a responsible report from a third party, CodeSprinter evaluates whether the current identification, classification and remediation procedures have functioned adequately. Lessons learned are recorded and incorporated into the policy or the supporting procedures.

Versioning and communication

Each material change to this policy is accompanied by a new version number and a revision date. Clients who have entered into a Software Maintenance Agreement or Managed Hosting SLA with CodeSprinter are informed of new versions that may affect the agreed way of working. The most recent version of this policy is available via info@codesprinter.nl.