TLP:CLEAR

Product Security Bad Practices

Version 2
Publication: January 2025

Cybersecurity and Infrastructure Security Agency
Federal Bureau of Investigation

TLP:CLEAR

Change Record

Version	Date	Revision/Change Description	Section/Page Affected
1.0	October 2024	Initial Version	
2.0	January 2025	Feedback incorporated from the 78 public comments CISA received in response to our Request for Information. This includes: ▪ Three new bad practices on use of known insecure or outdated cryptographic functions, hardcoded credentials, and product support periods. ▪ Additional context added to the memory safety section. ▪ Added additional examples of recommended actions to prevent SQL injection vulnerabilities. ▪ Added additional examples of recommended actions to prevent command injection vulnerabilities. ▪ Clarified timelines for patching Known Exploited Vulnerabilities (KEVs). ▪ Added language for multi-factor authentication (MFA) specific to operational technology products. ▪ Added that software manufacturers should support phishing-resistant MFA. ▪ Other updates to phrasing throughout.	Updates throughout.

Overview

As outlined in the Cybersecurity and Infrastructure Security Agency's (CISA's) Secure by Design initiative, software manufacturers should ensure that security is a core consideration from the onset of software development and throughout the entirety of the development lifecycle. This voluntary guidance provides an overview of product security bad practices that are considered exceptionally risky, particularly for software manufacturers who produce software used in service of critical infrastructure or national critical functions (NCFs). This guidance also provides recommendations for software manufacturers to mitigate these risks.

CISA and the Federal Bureau of Investigation (FBI)—hereafter referred to as the authoring organizations—developed this guidance to urge software manufacturers to reduce customer risk by prioritizing security throughout the product lifecycle. This document is intended for software manufacturers who develop software products and services, including on-premises software, cloud services, and software as a service (SaaS). This also applies to software products that run on operational technology (OT) products or embedded systems. The authoring organizations strongly encourage all software manufacturers to avoid these product security bad practices. By following the recommendations in this guidance, manufacturers will signal to customers that they are taking ownership of customer security outcomes, a key secure by design principle. The guidance contained in this document is non-binding, and while the authoring organizations encourage avoiding these bad practices, this document imposes no requirement to do so.

The bad practices are divided into three categories.

1. Product properties, which describe the observable, security-related qualities of a software product.
2. Security features, which describe the security functionalities that a product supports.
3. Organizational processes and policies, which describe the actions taken by a software manufacturer to ensure strong transparency in its approach to security.

This list is focused and does not include every possible inadvisable cybersecurity practice. The lack of inclusion of any particular cybersecurity practice does not indicate that the authoring organizations endorse or deem such a practice to present acceptable levels of risk. Items present in this list were chosen based on the threat landscape as representing the most dangerous and pressing bad practices that software manufacturers should avoid.

Product Properties

1) The development of new product lines for use in service of critical infrastructure or NCFs in a memory-unsafe language (e.g., C or C++) where readily available alternative memory-safe languages could be used is dangerous and significantly elevates risk to national security, national economic security, and national public health and safety.

   For existing products written in memory-unsafe languages, not having a published memory safety roadmap is dangerous and significantly elevates risk to national security, national economic security, and national public health and safety. The memory safety roadmap should outline the

manufacturer's prioritized approach to eliminating memory safety vulnerabilities in priority code components written in memory unsafe languages (e.g., network-facing code or code that handles sensitive functions like cryptographic operations). Manufacturers should demonstrate that the memory safety roadmap will lead to a significant, prioritized reduction of memory safety vulnerabilities in the manufacturer's products and demonstrate they are making a reasonable effort to follow the memory safety roadmap. Publication of a memory safety roadmap does not apply to products that have an announced end-of-support date that is prior to Jan. 1, 2030.

Note: The authoring organizations understand that significant time and resources must be invested to migrate to memory safe languages. Recognizing this, we encourage software manufacturers to plan for both mitigating memory safety vulnerabilities in the short term and eliminating them in the long term. For instance, a company might begin by writing new code components in memory safe languages and concurrently implement hardware or compiler controls to mitigate memory safety vulnerabilities. Over time, the company could rewrite parts of high-risk components (such as those performing cryptographic operations) using memory safe languages to incrementally improve memory safety over time. For additional guidance, see The Case for Memory Safe Roadmaps.

Recommended action: Software manufacturers should build products in a manner that systematically prevents the introduction of memory safety vulnerabilities. Software manufacturers should develop new product lines in memory safe languages. For existing products, software manufacturers should publish a memory safety roadmap by the end of 2025, outlining their prioritized approach to eliminating memory safety vulnerabilities in priority code components written in memory unsafe languages.

CWE (Common Weakness Enumeration): CWE-119

Resources: The Case for Memory Safe Roadmaps, CISA Secure by Design Pledge (Reducing Classes of Vulnerability), Back to The Building Blocks, NIST Secure Software Development Framework (SSDF) PW 6.1.

2) The inclusion of user-provided input directly in the raw contents of a SQL database query string in products used in service of critical infrastructure or NCFs is dangerous and significantly elevates risk to national security, national economic security, and national public health and safety.

   Recommended action: Software manufacturers should build products in a manner that systematically prevents the introduction of SQL injection vulnerabilities, such as by consistently enforcing the use of parametrized queries, prepared statements, or consistent use of an object-relational mapping (ORM) library that automatically generates parametrized queries.

   CWE: CWE-89

   Resources: CISA Secure by Design Pledge (Reducing Classes of Vulnerability), SSDF PW.5.1, CISA SQL Injection Secure by Design Alert.

3) The inclusion of user-provided input directly in the raw contents of an operating system command string in products used in service of critical infrastructure or NCFs is dangerous and significantly elevates risk to national security, national economic security, and national public health and safety.

Recommended action: Software manufacturers should build products in a manner that systematically prevents command injection vulnerabilities. Example approaches include consistently ensuring that command inputs are clearly delineated from the contents of a command itself; using built-in library functions instead of running a command, when available; and using restrictive allowlists that only allow alphanumeric characters and underscores to sanitize user input.

Note: This is intended to cover cases in which an application might be vulnerable to a command injection vulnerability, such as if user input on a website is invoked in an operating system command. This does not apply to cases where software intentionally allows users to execute commands, such as a terminal program that exposes a shell to users.

CWE: CWE-78

Resources: CISA Secure by Design Alert: Eliminating OS Command Injection Vulnerabilities, CISA Secure by Design Pledge (Reducing Classes of Vulnerability), SSDF PW.5.1.

4) The release of a product used in service of critical infrastructure or NCFs with default passwords, which CISA defines as universally-shared passwords that are present by default across a product with no requirement to be changed upon initialization, is dangerous and significantly elevates risk to national security, national economic security, and national public health and safety.

Recommended action: Software manufacturers should ensure that default passwords are not present in a product, such as by:

- Providing random, instance-unique initial passwords for the product.
- Requiring the user installing the product to create a strong password at the start of the installation process.
- Providing time-limited setup passwords that disable themselves when a setup process is complete and require configuration of a secure password (or more secure authentication approaches, such as phishing-resistant MFA).
- Requiring physical access for initial setup and the specification of instance-unique credentials.
- Conducting campaigns or offering updates that transition existing deployments from default passwords to more secure authentication mechanisms.

CWEs: CWE-1392 and CWE-1393

Resources: CISA Secure by Design Pledge (Default Passwords), SSDF PW.9.1, CISA Default Passwords Secure by Design Alert.

5) The release of a product used in service of critical infrastructure or NCFs that, at time of release, includes a component that contains an exploitable vulnerability present on CISA's Known Exploited Vulnerabilities (KEV) Catalog is dangerous and significantly elevates risk to national security, national economic security, and national public health and safety.[1] Additionally, if a new KEV affecting the product is published in CISA's KEV catalog, and the KEV is exploitable in the product, software

[1] For KEVs published within 30 days prior to a product's planned release, the manufacturer should issue a patch within 30 days from the date of which a patch for the component containing the KEV is made available.

manufacturers should issue a patch at no cost to its users in a timely manner to address the KEV. If the KEV is not exploitable in the product, software manufacturers should publicly document the presence of the vulnerability. Failure to take such actions is dangerous and significantly elevates risk to national security, national economic security, and national public health and safety.

Recommended action: Software manufacturers should patch all known exploited vulnerabilities within software components prior to release. In the case of the publication of a new KEV on CISA's catalog, the manufacturer should issue a patch at no cost to its users in a timely manner (no longer than 30 days from the date of which a patch for the component containing the KEV is made available) and clearly warn users of the associated risks of not installing the patch.

If the manufacturer deems a KEV cannot be exploited in its product (for instance, the KEV is only exploitable via a function that is never called), the manufacturer should publicly publish written documentation acknowledging the KEV and explaining how it is not exploitable in their product.[2]

Resources: CISA Secure by Design Pledge (Security Patches), SSDF PW.4.4, Binding Operational Directive 22-01.

6) The release of a product used in service of critical infrastructure or NCFs that, at time of release, includes open source software components that have critical vulnerabilities is dangerous and significantly elevates risk to national security, national economic security, and national public health and safety.[3] Additionally, if exploitable vulnerabilities are subsequently disclosed in the included open source components, failure to issue a patch or other mitigation at no cost to the product's users in a timely manner is dangerous and significantly elevates risk.

Recommended action: Software manufacturers should responsibly consume and sustainably contribute to the open source software that they depend on. This includes making a reasonable effort to evaluate and secure their open source software dependencies by taking the following actions:[4]

- Maintaining a software bill of materials (SBOM) in an industry-standard, machine-readable format describing all first- and third-party software dependencies, both open source and proprietary, and providing this to customers.
- Having an established process for managing the incorporation of open source software, including taking reasonable steps to:
  - Run security scanning tools on each open source software component when selected, including its dependencies and transitive dependencies, and each subsequent version when updated.
  - Select open source software projects that are well-maintained, and—when appropriate—contribute to the project's ongoing maintenance to sustain the expected standard of quality.
  - Evaluate alternatives to identify and select the most well-secured and maintained option.

[2] Ideally, the documentation should be published in a machine-processable format through Vulnerability Exploitability eXchange (VEX).
[3] Critical vulnerabilities are defined as those with a Common Vulnerability Scoring System (CVSS) score of 9.0 or greater.
[4] Organizations may choose to establish an open source program office (OSPO) to centralize these activities.

- Download open source software project artifacts from package repositories (or other appropriate sources) that adhere to security best practices.
- Routinely monitor for Common Vulnerabilities and Exposures (CVEs) or other security-relevant alerts, such as end-of-life, in all open source software dependencies and update them as necessary.
- Cache copies of all open-source dependencies within the manufacturer's own build systems and do not update products or customer systems directly from unverified public sources.

- Including the cost of updating to new major versions of third-party open source software dependencies in business planning activities and ensuring that such dependencies continue to receive necessary security fixes for the expected product life.

If the manufacturer deems that a critical vulnerability cannot be exploited in its product (because, for instance, the vulnerability is only exploitable via a function that is never called), the manufacturer should publicly publish written documentation acknowledging the vulnerability and explaining how it is not exploitable in their product.

Resources: SSDF PW.4.4, ESF Recommended Practices for Managing Open Source Software and Software Bill of Materials, TODO Group Open Source Program Office (OSPO) Definition and Guide

7) Information technology products used in service of critical infrastructure or NCFs and that use known insecure or deprecated cryptographic algorithms or lack encryption for the transit or storage of sensitive information are dangerous and significantly elevate risk to national security, national economic security, and national public health and safety.

Recommended action: Software manufacturers should utilize modern cryptographic algorithms to ensure that all sensitive data is protected in transit and at rest. Software manufacturers should avoid known insecure or deprecated algorithms such as Transport Layer Security (TLS) 1.0/1.1, MD5, SHA-1, and Data Encryption Standard (DES). Additionally, software manufacturers should begin supporting standardized post-quantum cryptographic algorithms consistent with NIST guidance. All websites should use modern TLS encryption.

Resources: Let's Encrypt, OWASP Transport Layer Security Cheat Sheet, NIST Post-Quantum Cryptography

8) The presence of hardcoded credentials or secrets in source code for products used in service of critical infrastructure or NCFs is dangerous and significantly elevates risk to national security, national economic security, and national public health and safety.

Recommended action: Software manufacturers should ensure that secrets are not present in source code, such as by using a secret manager that continuous integration / continuous deployment (CI/CD) pipelines and applications can use to securely retrieve secrets. Additionally, software manufacturers should integrate scanning for presence of secrets or credentials in their code into their development processes.

Resources: OWASP Hardcoded Password, OWASP Secrets Management

Security Features

9) For use in service of critical infrastructure or national critical functions, Information technology (IT) products that do not support multi-factor authentication (MFA), including phishing-resistant MFA, in the baseline version of the product are dangerous and significantly elevate risk to national security, national economic security, and national public health and safety.

   Additionally, IT products that do not enable MFA by default for administrator accounts are dangerous and significantly elevate risk to national security, national economic security, and national public health and safety. This does not apply to products that have an announced end-of-support date that is prior to Jan. 1, 2028.

   For OT products where MFA use may introduce safety risks (e.g., on medical devices in emergency rooms where delay in physician access could lead to patient harm), manufacturers should employ authentication measures that effectively mitigate the threat of single-factor credential abuse and other authentication threats. Manufacturers should publish a threat model detailing this approach.

   For OT products where MFA use may be safe, such as for vendor/maintenance accounts, remotely accessible user and engineering workstations, and remotely accessible HMIs, the product should support MFA.

   Note: Other phishing-resistant forms of authentication, such as passkeys, meet this definition even if they are the sole form of authentication.

   Recommended action: For all products besides excepted OT products listed above, software manufacturers should either support MFA, including phishing-resistant MFA, natively in the product (if the product itself handles authentication) or support the use of an external identity provider in the baseline version of the product, such as via standards-based single sign on. Software manufacturers should require MFA for administrators and allow administrators to require MFA for users in their organization, if applicable.

   Resources: CISA Secure by Design Pledge (Multi-Factor Authentication), SSDF PW.9.

10) It is dangerous and significantly elevates risk to national security, national economic security, and national public health and safety for products used in service of critical infrastructure or NCFs to not provide customers with current and historical artifacts and capabilities in the product baseline version sufficient to gather evidence of intrusion types that commonly affect the specific product or the class of products to which the product belongs, which at minimum includes:

   - Configuration changes or reading configuration settings;
   - Identity (e.g., sign-in and token creation) and network flows, if applicable; and
   - Data access or creation of business-relevant data.

   Recommended action:

   - As part of the baseline version of a product, software manufacturers should make logs available to customers in an industry-standard, machine-readable format related to, at minimum, the above listed areas.

- For cloud service providers and SaaS products, software manufacturers should retain such logs for a set timeframe (at least 6 months) at no additional charge and make those logs available to customers.

Resources: CISA Secure by Design Pledge (Evidence of Intrusions).

Organizational Processes and Policies

11) It is dangerous and significantly elevates risk to national security, national economic security, and national public health and safety for the software manufacturer of products used in service of critical infrastructure or NCFs to not issue CVEs in a timely manner for, at minimum, all critical or high impact vulnerabilities[5] (whether discovered internally or by a third party) affecting such products. Additionally, it is dangerous and significantly elevates risk to national security, national economic security, and national public health and safety to not include the CWE field in every CVE record.

Recommended action: Software manufacturers should publish complete CVEs, including the appropriate CWE field, in a timely manner for all critical or high impact vulnerabilities.

Resources: CISA Secure by Design Pledge (CVEs), SSDF RV.1.3.

12) Not having a published vulnerability disclosure policy (VDP) that includes the product in its scope is dangerous and significantly elevates risk to national security, national economic security, and national public health and safety.

Recommended actions:

- Software manufacturers should publish a VDP that:
  - Authorizes testing by members of the public on products offered by the manufacturer;
  - Commits to not recommending or pursuing legal action against anyone engaging in good faith efforts to follow the VDP;
  - Provides a clear channel to report vulnerabilities; and
  - Allows for public disclosure of vulnerabilities in line with coordinated vulnerability disclosure (CVD) best practices and international standards.
- Software manufacturers should remediate all valid reported vulnerabilities in a timely and risk-prioritized manner.

Resources: CISA Secure by Design Pledge (Vulnerability Disclosure Policy), SSDF RV.1.3, ISO 29147.

13) For on-premises products, it is dangerous and significantly elevates risk to national security, national economic security, and national public health and safety for the software manufacturer to not clearly communicate the period of support for the product.

Recommended actions: Software manufacturers should clearly communicate the period of support for their products at the time of sale. Software manufacturers should provide security updates through the entire support period.

[5] Critical vulnerabilities are defined as those with a CVSS score of 9.0 or greater. High impact vulnerabilities are defined as those with a CVSS score of 7.0 or higher.